Федеральное агентство по рыболовству
БГАРФ ФГБОУ ВО «КГТУ»
Калининградский морской рыбопромышленный колледж

Утверждаю
Заместитель начальника колледжа
по учебно-методической работе

А.И.Колесниченко

Подписано цифровой подписью:
Колесниченко Андрей Игоревич
Причина: Я утвердил этот документ
Дата: 3 Июнь 2025 г. 15:45:16
DN: E=uc_fk@roskazna.ru, S=77 Москва, ИНН ЮЛ=7710568760, ОГРН=1047797019830, STREET="Большой Златоустинский переулок, д. 6, строение 1", L=г. Москва, C=RU, O=Казначейство России, CN=Федеральное казначейство

ООД.10 ИНФОРМАТИКА

1 семестр

Методические указания для выполнения практических занятий
по специальности

09.02.06 Сетевое и системное администрирование

МО-09 02 06-ООД.10.ПЗ

РАЗРАБОТЧИКИ	Кривонос Е.В., Сукорская А.О., Халина Е.Н.
ЗАВЕДУЮЩИЙ ОТДЕЛЕНИЕМ	А.М.Бакулин
ГОД РАЗРАБОТКИ	2022
ГОД ОБНОВЛЕНИЯ	2025

Содержание

Введение 3
Практическое занятие №1 Роль информационной деятельности в современном обществе, его экономической, социальной, культурной и образовательной сферах 4
Практическое занятие №2 Техника безопасности при работе в компьютерном классе. Безопасность, гигиена, эргономика, ресурсосбережение 13
Практическое занятие № 3 Основные этапы развития информационного общества. Информационные ресурсы общества. Работа с ними 28
Практическое занятие № 4 Образовательные информационные ресурсы КМРК. Электронная библиотека 31
Практическое занятие № 5 Представление информации в различных системах счисления 41
Практическое занятие №6 Арифметические операции над числами, записанными в двоичной, восьмеричной и шестнадцатеричной системе счисления 57
Практическое занятие №7 Измерение информации. Алфавитный и вероятностный подход к измерению информации 65
Практическое занятие №8 Кодирование и декодирование информации. Кодовые таблицы 69
Практическое занятие № 9 Арифметические и логические основы работы компьютера 78
Практическое занятие №10 Составление таблиц истинности по логическим выражениям 83
Практическое занятие № 11 Построение логических схем 87
Практическое занятие № 12 Графы. Введение и понятия 93
Практическое занятие №13 Графы. Способы задания графов 98
Практическое занятие № 14 Решение логических задач с помощью графов 104
Практическое занятие № 15 Файл как единица хранения информации на компьютере. Атрибуты файла и его объем. Учет объемов файлов при их хранении, передаче 110
Практическое занятие №16 Файл и файловая структура 116
Практическое занятие №17 Сжатие информации. Алгоритм Хаффмана. Создание архива данных. Извлечение данных из архива 123
Практическое занятие №18 Алгоритмы и способы их описания. Линейные алгоритмы 128
Практическое занятие №19 Алгоритмы и способы их описания. Условный и циклический алгоритм 136
Практическое занятие № 20 Описание алгоритма с помощью блок-схем, выполненных в программе Dia 141
Практическое занятие №21 Кумир. Знакомство с исполнителем Робот. Простейшие и линейные алгоритмы 144
Практическое занятие №22 Кумир Робот. Циклические и условные алгоритмы 156
Практическое занятие №23 Кумир Робот Составление алгоритмов 160
Практическое занятие №24 Введение в язык программирования Python 162
Практическое занятие №25 Математические операции в Python 176
Практическое занятие №26 Структура ветвление в Python 186
Практическое занятие №27 Работа с циклами в Python 194
Практическое занятие №28 Работа со строками в Python 201
Практическое занятие №29Работа со списками, кортежами, словарями. Операции над списками в Python 206
Практическое занятие №30 Одномерные массивы как списками. Заполнение и обработка одномерных массивов в Python 212
Практическое занятие №31 Работа с двумерными массивами 222
Практическое занятие №32 Функции и процедуры в Python 238
Практическое занятие №33 Дополнительные типы данных в Python 247
Практическое занятие №34 Знакомство с графикой в Python 254
Практическое занятие №35 Итоговая работа по первому семестру 264

Введение

Рабочей программой дисциплины предусмотрено 35 практических занятий в первом семестре и 31 практических занятий во втором семестре. Целью их проведения является приобретение пользовательских навыков работы с ПК. Наряду с закреплением имеющихся умений в процессе практических занятий курсанты получают навыки по применению ПК на старших курсах и в своей профессиональной деятельности.

Выполнение практических занятий направлено на формирование у обучающихся следующих компетенций:

- общие компетенции:

ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам.

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности.

Содержание учебной программы при ограниченном времени, отведенном на изучение дисциплины «Информатика», требует от обучающихся запоминания изучаемого материала и развития умений, навыков самостоятельной работы с учебной литературой и персональным компьютером. Важное место здесь занимают практические занятия по алгоритмизации и программированию, которые развивают логическое мышление обучающихся, творческий подход к решению задач.

Профильная составляющая общеобразовательной дисциплины Информатика осуществляется:

1. Путем дидактических единиц программы по информатике, знание которых будет необходимо при освоении ППССЗ СПО и в будущей профессиональной деятельности.

2. Через межпредметные связи дисциплины с дисциплинами «Физика», «Математика» и с профильными дисциплинами ППССЗ СПО

3. Через организацию внеаудиторной самостоятельной работы, направленной на расширение и углубление знаний, которые будут необходимы при осуществлении профессиональной деятельности

Перед проведением практических занятий обучающиеся обязаны проработать теоретическую часть практического занятия, уяснить цель задания, ознакомиться с

содержанием и последовательностью его выполнения, а преподаватель проверить их готовность к выполнению задания.

Задания практических занятий выполняются на ПК, каждым обучающимся и в конце занятия проверяется преподавателем.

После каждого практического занятия обучающиеся должны подготовить ответы на вопросы в письменной форме (возможна устная форма) и сдать отчет о проделанной работе преподавателю. Только после этого практическое занятие будет оценено преподавателем.

Практическое занятие №1Роль информационной деятельности в современном обществе, его экономической, социальной, культурной и образовательной сферах

Цель занятия: Познакомить обучающихся с понятием информатика, информация, информационное общество; информация культура;

Рассмотреть роль информационной деятельности в современном обществе, его экономической, социальной, культурной, образовательной и других сферах

Исходные данные: раздаточный материал

Содержание и порядок выполнения задания:

Выполнить входное тестирование

Пример входного тестирования

1. **Под информацией понимают:**

Сигналы от органов чувств человека
Характеристику объекта, выраженную в числовых величинах
Разнообразие окружающей действительности

2. **В позиционной системе счисления значение каждой цифры зависит:**

От значения числа
От значений соседних знаков
От позиции, которую занимает знак в записи числа

3. **За единицу измерения информации в теории кодирования принят:**

1 байт
1 бод
1 бит

4. Электронная таблица предназначена для:

Обработки преимущественно числовых данных, структурированных с помощью таблиц,

Визуализации структурных связей между данными, представленными в таблицах,

Хранения и редактирования больших объемов текстовой информации,

5. Производительность работы компьютера (быстрота выполнения операций) зависит от...

размера экрана дисплея
частоты процессора
быстроты, нажатия на клавиши

6. Драйвер – это:

1) устройство компьютера;
2) программа для работы с устройствами компьютера,
3) язык программирования.

7. В целях сохранения информации CD-ROM необходимо оберегать от...

1) холода
2) магнитных полей
3) загрязнения

8. Укажите устройство, не являющееся устройством вывода информации:

1) монитор
2) принтер
3) клавиатура

Часть В.

9. Ниже в табличной форме представлен фрагмент базы данных о результатах тестирования учащихся (используется сто балльная шкала).

Фамилия	Пол	математика	химия	информатика	Биология
Аганян	Ж	52	43	82	74
Воронин	М	92	75	93	55
Григорчук	М	66	69	51	68
Роднина	Ж	73	51	40	92
Сергеевко	Ж	81	83	83	41
Черепанова	ж	94	64	71	20

Сколько записей в данном фрагменте удовлетворяет условию
«Математика > 60 И Информатика > 55»?
В ответе укажите одно число — искомое количество записей.
Ответ: _____

10. Пользователь работал с каталогом С:\Архив\Рисунки\Натюрморты.
Сначала он поднялся на один уровень вверх, затем еще раз поднялся на один уровень вверх и после этого спустился в каталог Фотографии. Запишите полный путь каталога, в котором оказался пользователь.

1) С:\Архив\Рисунки\Фотографии
2) С:\Архив\Фотографии
3) С:\Фотографии\Архив
4) С:\Фотографии

Часть С.

11. Сколько Кбайт информации содержит сообщение объемом 2^{16}бит? В ответе укажите одно число.

Ответ: _____

Теоретический материал

Информатика – наука об информации и технических средствах ее сбора, хранения, обработки, передачи.

Одним из основных понятий дисциплины Информатика, которое очень часто применяется представителями различных научных дисциплин и направлений, является понятие **"информация".**

Поговорка «Кто владеет информацией - тот владеет миром» подчёркивает важную роль информации.

Информация - это сведения об окружающем нас мире и протекающих в нем процессах.

Свойства информации:

1) Объективность и субъективность – зависит от человеческого фактора.

2) Полнота, если ее достаточно для понимания и принятия решений. Неполнота информации сдерживает принятие решений или может повлечь ошибки.

3)Актуальность–актуальную информацию важно иметь при работе в изменившихся условиях.

4) Достоверность, если она отражает истинное положение дел. Недостоверная информация может привести к неправильному пониманию или принятию неправильных решений.

5) Понятность. Если ценная и актуальная информация выражена непонятными словами, она может стать бесполезной. Информация становится понятной, если она выражена языком, на котором говорят те, кому предназначена эта информация.

6)Доступность–возможность получения информации

7) Адекватность – степень соответствия реальному объективному состоянию дела.

Информационный процесс — это процесс сбора (приёма), передачи (обмена), хранения, обработки (преобразования) информации.

На заре цивилизации человеку было достаточно элементарных знаний и первобытных навыков. По мере развития общества участие в информационных процессах требовало уже не только индивидуальных, но и коллективных знаний и

опыта, способствующих правильной переработке информации и принятию необходимых решений. Для этого человеку понадобились различные устройства.

Этапы появления средств и методов обработки информации, вызвавших кардинальные изменения в обществе, определяются как информационные революции. При этом общество переходит на более высокий уровень развития и обретает новое качество. Информационные революции определяют переломные моменты во всемирной истории, после которых начинаются новые этапы развития цивилизации, появляются и развиваются принципиально новые технологии.

Первая информационная революция связана с изобретением письменности, обусловившей гигантский качественный скачок в развитии цивилизации.

Вторая информационная революция (середина XVI века) началась в эпоху Возрождения и связана с изобретением книгопечатания.

Третья информационная революция (конец XIX века) связана с изобретением электричества, благодаря которому появились телеграф, телефон и радио, позволяющие оперативно передавать информацию в любом объеме. Появилась возможность обеспечить более оперативный обмен информацией между людьми.

Четвертая информационная революция (70-е годы XX столетия) связана с изобретением микропроцессорной технологии и появлением персональных компьютеров. Это стимулировало переход от механических и электрических средств преобразования информации к электронным, что привело к миниатюризации узлов, устройств, приборов, машин и появлению программно-управляемых устройств и процессов. На микропроцессорах и интегральных схемах стали создаваться компьютеры, компьютерные сети, системы передачи данных (информационно-коммуникационные системы) и т. д.

Роль информационной деятельности в современном обществе.

Информация всегда являлась немаловажным фактором общественного развития. С ее помощью человечество концентрировало многовековой опыт жизни прежних поколений. Поговорка «Кто владеет информацией - тот владеет миром» подчёркивает социальную роль информации. Владение информацией открывает новые модели управления, формирует новые социальные структуры общества.

Под влиянием быстро развивающейся системы социальных коммуникаций и информации формируется информационный сектор экономики. Возникает новый капитал - знание. Под воздействием этих процессов меняется характер труда:

внедряются безлюдные технологии, т.е. исчезает сам труд в непосредственном производстве, расширяется участие работников в управлении производством.

Информация как орудие продвижения прогрессивных идей способствует всё большей прозрачности границ между странами и народами. Складывается новая система общественного богатства с использованием информационных технологий, где ценятся, прежде всего, умственные способности человека.

По данным ЮНЕСКО, более половины всего занятого населения наиболее развитых стран прямо или косвенно принимает участие в процессе производства, хранения и распространения информации.

Если до середины XX века общество имело ярко выраженный индустриальный характер, то нынешнее его состояние ученые характеризуют как постиндустриальное, рассматривая его в качестве переходного к информационному обществу.

Информационное общество — общество, в котором большинство работающих занято производством, хранением, переработкой и реализацией информации, особенно высшей ее формы — знаний.

Признаки информационного общества

Осознание обществом приоритетности информации перед другим продуктом деятельности человека.

Информация в чистом виде (сама по себе) является предметом купли – продажи.

Равные возможности в доступе к информации всех слоев населения.

Безопасность информационного общества, информации.

Защита интеллектуальной собственности.

Взаимодействие всех структур государства и государств между собой на основе ИКТ.

Негативные тенденции, наблюдаемые в информационном обществе:

информационные технологии могут разрушить частную жизнь людей и организаций;

существует проблема отбора качественной и достоверной информации;

многим людям будет трудно адаптироваться к среде информационного общества.

Ближе всех на пути к информационному обществу стоят страны с развитой информационной индустрией: США, Япония, Англия, Германия, страны Западной Европы.

В период перехода к информационному обществу необходимо подготовить человека к быстрому восприятию и обработке больших объемов информации, овладению им современными средствами, методами и технологией работы. Кроме того, новые условия работы порождают зависимость информированности одного человека от информации, приобретенной другими людьми. Поэтому уже недостаточно уметь самостоятельно осваивать и накапливать информацию, а надо учиться такой технологии работы с информацией, когда подготавливаются и принимаются решения на основе коллективного знания. Это говорит о том, что человек должен иметь определенный уровень культуры по обращению с информацией, так называемую, информационную культуру.

Информационная культура - умение целенаправленно работать с информацией и использовать для её получения, обработки и передачи компьютерные информационные технологии, современные технические средства и методы. Для свободной ориентации в информационном потоке человек должен обладать информационной культурой как одной из составляющих общей культуры, которую нужно формировать с детства.

Компьютерная грамотность предполагает:

знание назначения и пользовательских характеристик основных устройств компьютера;

знание основных видов программного обеспечения;

умение производить поиск, хранение, обработку текстовой, графической, числовой информации с помощью соответствующего программного обеспечения.

Информационная деятельность – деятельность, обеспечивающая сбор, обработку, хранение, поиск и распространение информации, а также формирование информационного ресурса и организацию доступа к нему.

Мы живем в XXI веке и в нашу жизнь прочно вошли компьютеры и информационные технологии, наша работа, отдых, развлечения прочно с ними связаны. Давайте с вами рассмотрим в каких сферах нашей жизни и деятельности мы их используем. Например, в экономической сфере.

Главная цель информационных технологий **экономической сфере** – в результате целенаправленных действий по переработке первичных данных

получить необходимую для пользователя информацию. К примеру, имеются данных о каком - либо производстве: стоимость исходно сырья, затраты на энергию, заработная плата рабочим и др. Нужно подсчитать стоимость полученного товара, прибыль. Можно считать в ручную по известным формулам, а можно использовать уже готовые программы, которые все подсчитают и выдадут необходимую для пользователя информацию.

Экономическая информация используется в качестве ресурса, услуг, товара, источника добавленной стоимости и занятости, получает развитие электронный бизнес. Не нужно командировать представителя к деловому партнеру из другого региона, документы заверяются электронной цифровой подписью. Не нужно тратить время на выбор товара, достаточно просмотреть каталог электронного магазина. Не нужно посещать налоговую инспекцию, чтобы сдать налоговую отчетность. Не нужно тратить время на дорогу, чтобы выполнить свою работу (для некоторых видов профессиональной деятельности). Не нужно ехать в кассу, чтобы купить билет на поезд, его достаточно заказать и оплатить дистанционно.

Если рассматривать информационную деятельность в **социальной сфере**, то видно, что информация стала более доступной для человека. Сейчас большое количество источников информации, это и традиционные газеты, журналы, радио, телевидение, а теперь и компьютер, интернет, сотовый телефон и т.д. Если раньше информация печаталась на бумаге, затем только доходила до человека, то теперь телевидение, радио и интернет в режиме реального времени передают любую информацию «тут же». Даже с помощью сотового телефона вы будете информированы о стихийных бедствиях, о распродажах или скидках. Всё это делает социальную сферу более информационной. А различные социальные сети в сети интернет позволяют общаться людям на большой расстоянии друг от друга, а также передавать любую информацию, будь то фотографии или важные документы.

С помощью информационных технологий общество становится более грамотным, так как можно достаточно быстро найти ответы на нужные вопросы, а также получить правильные советы. Само название "информационное общество" впервые появилось в Японии в середине 60-х годов XX века. Специалисты, предложившие этот термин, разъяснили, что он характеризует общество, в котором в изобилии циркулирует высокая по качеству информация, а также есть все необходимые средства для ее хранения, распределения и использования. Информация легко и быстро распространяется по требованиям заинтересованных

людей и организаций и выдается им в привычной для них форме. Отличительными особенностями информационного общества являются: открытость, технологичность (особенность информатизации), интеллектуальность, доступ к мировым информационным ресурсам, высокая степень обеспечения безопасности, гибкость и самоорганизация выше указанных систем. В таком обществе наблюдается ускоренная автоматизация и роботизация всех отраслей производства и управления, происходят радикальные изменения социальных структур.

Социальная информация выступает в качестве важного стимулятора изменения качества жизни. Чтобы получить консультацию специалиста, пациенту не нужно ехать в медицинский центр, а достаточно будет оставить свои документы на портале и в назначенное время выйти на связь с профильным врачом (телемедицина). Чтобы получить помощь в чрезвычайной ситуации, достаточно воспользоваться единым номером экстренных служб (например, система «Забота», о которой будет подробнее рассказано в одной из следующих лекций). Чтобы собрать ученика в школу, достаточно скачать комплект учебников с регионального образовательного портала и сохранить их в электронной книге.

Если рассматривать информационную деятельность в **культурной сфере**, то тут видно, что с одной стороны с помощью столь многих информационных источников можно поддерживать и развивать культурную деятельность, а с другой стороны можно и разрушить всю культуру человечества. С помощью телевидения, интернета можно с легкостью транслировать различные концерты, представления, учить культуре, этике. С другой стороны, за счет столь обширной информации, внедряется в нашу жизнь и безкультурие, идущее из – за рубежа. С появлением компьютера, многие люди перестали ходить в театры, встречаться друг с другом и т.д. Все это заменяет общение в интернете и телевидение.

Информационная деятельность и культура неразрывно связанно со СМИ. Средства массовой информации кардинально меняет наш быт, меняет нашу моду, говорит нам, как правильно питаться, чему можно верить и т.д. С помощью СМИ можно внедрять в общество ту или иную культурную среду.

Культурная: признание культурной ценности информации (например, проект ЮНЕСКО «Цифровое наследие»). Чтобы подобрать литературу по интересующей тематике, достаточно воспользоваться электронным каталогом любой библиотеки на всей территории страны. Чтобы посетить зарубежный музей, достаточно

побывать на соответствующем сайте. Чтобы получить образование в любом университете мира, нужно обратиться к его ресурсам дистанционного обучения.

Современное **образование** тоже немыслимо без применения компьютеров. Компьютеры помогают моделировать явления и процессы, осваивать теоретический материал, осуществлять эффективный контроль уровня знаний учащихся. Используя сайт образовательного учреждения , можно записаться на лабораторные работы, посмотреть расписание, разместить ответ на вопрос на специальном форуме, записаться на курс дистанционного обучения и многое другое. В чем же состоит причина такой популярности информационных технологий в образовании? Прежде всего в том, что компьютер позволяет в значительной мере индивидуализировать учебный процесс, выбрать темп обучения, степень подробности, дает возможность еще и еще раз вернуться к материалу, проверить уровень понимания его и тд. Компьютерное моделирование дает возможность провести реально невыполнимые в обычных условиях эксперименты. Например, научится кровообращение в организме человека, управлять государством, расщеплять ядерное ядро или путешествовать по миру. Особая роль принадлежит системе дистанционного обучения. Информационные технологии и ресурсы дистанционного обучения в Интернете являются на сегодня самым мощным средством самообразования и саморазвития людей, со школьной скамьи приобщая к этому процессу, без которого в современном информационном обществе невозможен путь к успеху в профессиональной сфере. И сегодня получая образование в СПО вы будете активно использовать ЭВМ и информационного технологии для получения образования по своей специальности.

К образовательным технологиям относятся: видео-лекции; мультимедиа-лекции и лабораторные практикумы; электронные мультимедийные учебники; компьютерные обучающие и тестирующие системы; имитационные модели и компьютерные тренажеры; консультации и тесты с использованием телекоммуникационных средств; видеоконференции.

Медицина- очень сложная наука. И здесь ей на помощь приходит компьютер. Существует множество болезней, каждая из которых имеет только ей присущие симптомы. Кроме того, есть десятки болезней с похожими симптомами. В подобных случаях врачу бывает трудно составить точный диагноз. В настоящее время врачи используют компьютерную диагностику.

Современная армия невозможна без компьютерной техники и тд. Подумайте и приведите примеры, в каких еще сферах нашей жизни мы используем компьютер и информационные технологию.

Задание: Используя теоретический материал письменно ответить на вопросы для самопроверки.

Вопросы для самопроверки:

Что изучает информатика?

Что такое информация?

Что такое информационное общество?

Что входит в состав информационных ресурсов?

Перечислите признаки информационного общества.

Что обеспечивает использование современных информационных технологий?

Как вы считаете- владеете ли вы компьютерной грамотностью и почему?

Можете ли вы назвать себя информационно культурным человеком?

Практическое занятие №2Техника безопасности при работе в компьютерном классе. Безопасность, гигиена, эргономика, ресурсосбережение

Цель занятия: 1. Оценить знания и умения обучающихся, полученные в школе

2. Ознакомить с техникой безопасности при работе в кабинете информатики

Исходные данные: раздаточный материал, журнал инструктажа по технике безопасности

Содержание и порядок выполнения задания:

1. ПРАВИЛА ПОВЕДЕНИЯ УЧАЩИХСЯ В УЧЕБНОМ КАБИНЕТЕ ИНФОРМАТИКИ

1.1. Находясь в кабинете информатики, учащиеся обязаны:

- соблюдать дисциплину и порядок, правила техники безопасности и чистоту;
- занимать рабочие места согласно указаниям преподавателя и не менять их самовольно;

- заниматься только тем видом деятельности, которую определил преподаватель;
- немедленно сообщать преподавателю о любых замеченных неисправностях оборудования или неверной работе программного обеспечения;
- немедленно сообщать преподавателю о любом случае травматизма в кабинете, особенно от электрического тока.

1.2. Находясь в кабинете информатики, учащийся имеет право:

- на помощь и консультацию преподавателя;
- отказаться от продолжения работы с компьютером, если длительность именно его индивидуальной работы превышает допустимые санитарные нормы;
- самостоятельно экстренно отключить электрооборудование, если от этого зависит безопасность его или окружающих.

2 ПРАВИЛА ТЕХНИКИ БЕЗОПАСНОСТИ В КАБИНЕТЕ ИНФОРМАТИКИ

1. ИСТОЧНИКИ ОПАСНОСТИ:

- электроприборы с напряжением питания 220 В, мониторы и телевизоры, которые могут явиться источником электротравматизма;
- наличие электроприборов увеличивает опасность возгорания;
- мониторы компьютеров, телевизоры являются слабыми источниками ионизирующего излучения электромагнитных, электрических и магнитных статических полей.

2. ПРАВИЛА БЕЗОПАСНОСТИ

ЗАПРЕЩАЕТСЯ:

- работать с электроприборами, имеющими повреждения корпуса или изоляции соединительных проводов;
- производить самовольное переключение разъёмов оборудования;
- приносить и самовольно подключать какое-либо оборудование;
- вставлять в отверстие приборов посторонние предметы;
- выключать или включать приборы без разрешения преподавателя.

Если производится выключение/включение, то интервал времени между включением/и выключением/включением должен быть не менее 15 секунд.

В СЛУЧАЕ ПОРАЖЕНИЯ ЭЛЕКТРИЧЕСКИМ ТОКОМ, НЕОБХОДИМО:

- прекратить действие тока (лучше всего экстренным выключением приборов, т.к. попытка оттащить пострадавшего может привести к поражению током спасающего);

- немедленно сообщить о происшедшем преподавателю (даже если на первый взгляд всё обошлось лёгким испугом);
- оказать первую медицинскую помощь, если необходима.

3. ПРАВИЛА ПОЖАРНОЙ БЕЗОПАСНОСТИ

ЗАПРЕЩАЕТСЯ:

- использовать источники открытого огня (спички, зажигалки, петарды и др.);
- приносить на уроки легковоспламеняющиеся вещества (лаки, краски, порох и т.п.);
- пользоваться неисправными электроприборами (в случае появления специфического запаха горящей изоляции, соответствующий прибор необходимо немедленно отключить и сообщить учителю);
- загромождать или закрывать проходы к путям эвакуации и доступ к средствам первичного пожаротушения;
- производить тушение возгорания не отключенных электроприборов водой или обычными огнетушителями;
- привлекать учащихся к тушению пожара.

В СЛУЧАЕ УГРОЗЫ ПОЖАРА (возгорания, задымленность) НЕОБХОДИМО:

- немедленно отключить все электроприборы, определить источники возгорания (задымленности) и ликвидировать его средствами первичного пожаротушения;
- если первичные действия по ликвидации возгорания в течение первых же минут не дали результата, учащиеся эвакуируются согласно плану эвакуации, по школе объявляется тревога, сообщается о пожар.

САНИТАРНО-ГИГИЕНИЧЕСКИЕ НОРМЫ ПРИ РАБОТЕ С КОМПЬЮТЕРОМ

Требования к организации занятий в кабинете информатики

- расстояние от центра экрана до глаз учащихся должно быть не менее 60 см;
- время интенсивной непрерывной работы на компьютере не должно превышать 25 минут, после чего обязателен перерыв с разминкой;
- в кабинете должна быть обеспечена вентиляция и проветривание между уроками.

Комплексы упражнений для глаз

Теоретический материал

Эргономика – наука о том, как люди с их различными физическими данными и особенностями жизнедеятельности взаимодействуют с оборудованием и машинами, которыми они пользуются.

Цель эргономики состоит в том, чтобы обеспечить комфорт, эффективность и безопасность при пользовании компьютерами уже на этапе разработки клавиатур, компьютерных плат, рабочей мебели и др. для устранения физического дискомфорта и проблем со здоровьем на рабочем месте.

В связи с тем, что всё больше людей проводят много времени перед компьютерными мониторами, ученые многих областей, включая анатомию, психологию и охрану окружающей среды, вовлекаются в изучение правильных, с точки зрения эргономики, условий работы.

Так называемые *эргономические заболевания* – быстрорастущий вид профессиональных болезней.

Если в организации рабочего места оператора ПК допускается несоответствие параметров мебели антропометрическим характеристикам человека, то это вызывает необходимость поддержания вынужденной рабочей позы и может привести к нарушениям в костно-мышечной и периферической нервной системе. Длительный дискомфорт в условиях недостаточной физической активности может вызывать развитие общего утомления, снижения работоспособности, боли в области шеи, спины, поясницы. У операторов часто диагностируются заболевания опорно-двигательного аппарата и периферической нервной системы: невриты, радикулиты, остеохондроз и др.

Главной частью профилактических мероприятий в эргономике является правильная посадка.

Негативные последствия работы за монитором возникают из-за того, что:

а) наш глаз предназначен для восприятия отражённого света, а не излучаемого, как в случае с монитором (телевизором),

б) пользователю приходится вглядываться в линии и буквы на экране, что приводит к повышенному напряжению глазных мышц.

Для нормальной работы нужно поместить монитор так, чтобы глаза пользователя располагались на расстоянии, равном полутора диагоналям видимой части монитора:

- не менее 50-60 см для 15" монитора;

- не менее 60-70 см для 17" монитора;
- не менее 70-80 см для 19" монитора;
- не менее 80-100 см для 21" монитора.

Если зрение не позволяет выдерживать это расстояние, тогда уменьшите разрешение изображения и увеличьте шрифты.

Оптимальная диагональ экрана для работ с текстовыми документами - 15"-17" с разрешением 1024x768. Для графических работ необходим монитор 19"-21" при разрешении 1280x1024 и выше. Для игр рекомендуется 17"-19". Мониторы больших диагоналей приобретать не рекомендуется, т.к. от работы за слишком крупными мониторами, по словам пользователей, "глаза становятся квадратными".

От большого монитора необходимо сидеть дальше, чем от маленького. И в итоге угловая площадь монитора остается такой же. Но сфокусировать глаз на мелком изображении, находящемся в 1-1.5 метрах от глаза становится труднее, что ведет к перенапряжению зрительного аппарата. Чем крупнее объект на экране монитора, тем меньше утомляемость. Поэтому компьютерные игры с их рисованными фигурами утомляют меньше, чем цифры и буквы.

Экран монитора должен быть абсолютно чистым. Периодически и при необходимости протирайте его специальными салфетками.

Усталость от работы с монитором тем меньше, чем ниже яркость экрана и чем крупнее объекты на экране. Установите минимальную яркость, при которой можно без напряжения различать символы на экране. Учтите, что лучше увеличить шрифт или изображение, чем пододвинуться поближе к экрану или увеличить яркость. Современные операционные системы имеют для этого специальные средства. Шрифты на экране можно масштабировать, задавать минимальные размеры элементов рисунков и прочее.

Система гигиенических требований

Длительная работа с компьютером может приводить к расстройствам состояния здоровья. Кратковременная работа с компьютером, установленным с грубыми нарушениям гигиенических норм и правил, приводит к повышенному утомлению. Вредное воздействие компьютерной системы на организм человека является комплексным. Параметры монитора оказывают влияние на органы зрения. Оборудование рабочего места влияет на органы опорно-двигательной системы. Характер расположения оборудования в компьютерном классе и режим его

использования влияет как на общее психофизиологическое состояние организма, так и им органы зрения.

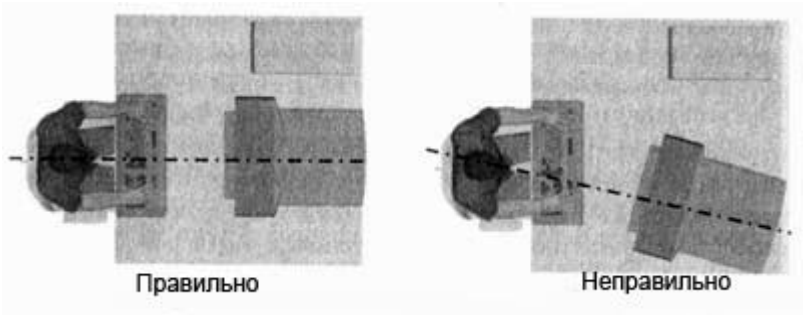
Требования к видеосистеме

В прошлом монитор рассматривали м основном как источник вредных излучений, воздействующих прежде всего на глаза. Сегодня такой подход считается недостаточным. Кроме вредных электромагнитных излучений (которые на современных мониторах понижены до сравнительно безопасного уровня) должны учитываться параметры качества изображения, а они определяются не только монитором, но и видеоадаптером, то есть всей видеосистемы в целом.

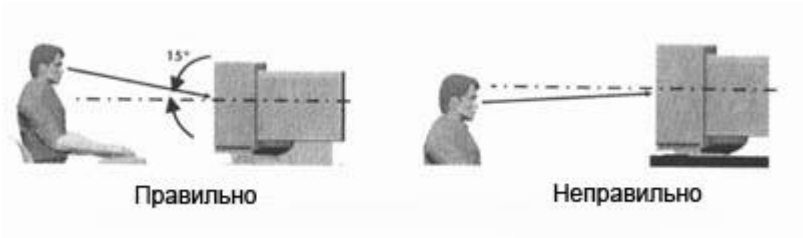
Требования к рабочему месту

В требования к рабочему месту входят требования к рабочему столу, посадочному месту (стулу, креслу), Подставкам для рук и ног. Несмотря на кажущуюся простоту, обеспечить правильное размещение элементов компьютерной системы и правильную посадку пользователя чрезвычайно трудно. Полное решение проблемы требует дополнительных затрат, сопоставимых по величине со стоимостью отдельных узлов компьютерной системы, поэтому и в быту и на производстве этими требованиями часто пренебрегают.

Монитор должен быть установлен прямо перед пользователем и не требовать поворота головы или корпуса тела.

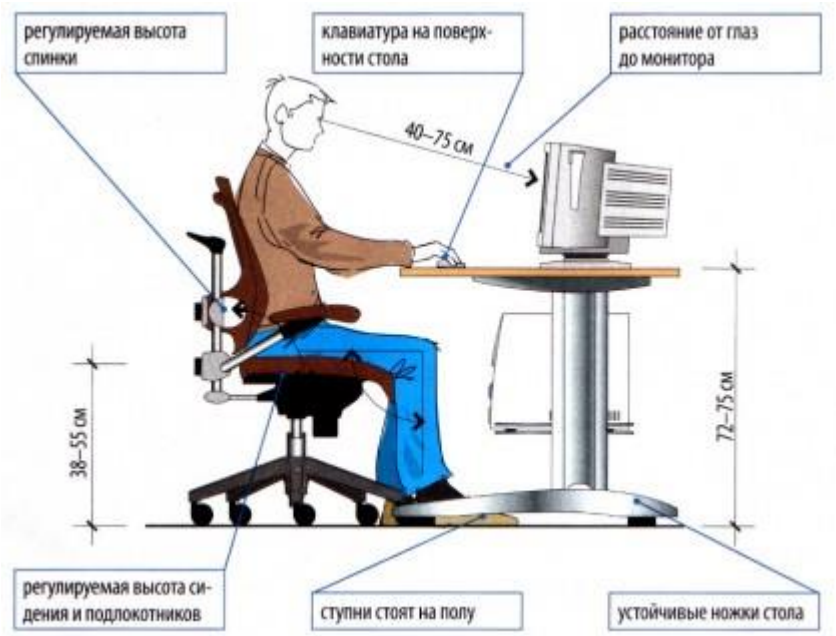


Рабочий стол и посадочное место должны иметь такую высоту, чтобы уровень глаз пользователя находился чуть выше центра монитора. На экран монитора следует смотреть сверху вниз, а не наоборот. Даже кратковременная работа с монитором, установленным слишком высоко, приводит к утомлению шейных отделов позвоночника.



Если при правильной установке монитора относительно уровня глаз выясняется, что ноги пользователя не могут свободно покоиться на полу, следует установить подставку для ног, желательно наклонную. Если ноги не имеют надежной опоры, это непременно ведет к нарушению осанки и утомлению позвоночника. Удобно, когда компьютерная мебель (стол и рабочее кресло) имеют средства для регулировки по высоте. В этом случае проще добиться оптимального положения.

Клавиатура должна быть расположена на такой высоте, чтобы пальцы рук располагались на ней свободно, без напряжения. Для работы рекомендуется использовать специальные компьютерные столы, имеющие выдвижные полочки для клавиатуры.



При длительной работе с клавиатурой возможно утомление сухожилий кистевого сустава. Известно тяжелое профессиональное заболевание — кистевой туннельный синдром, связанное с неправильным положением рук на клавиатуре.

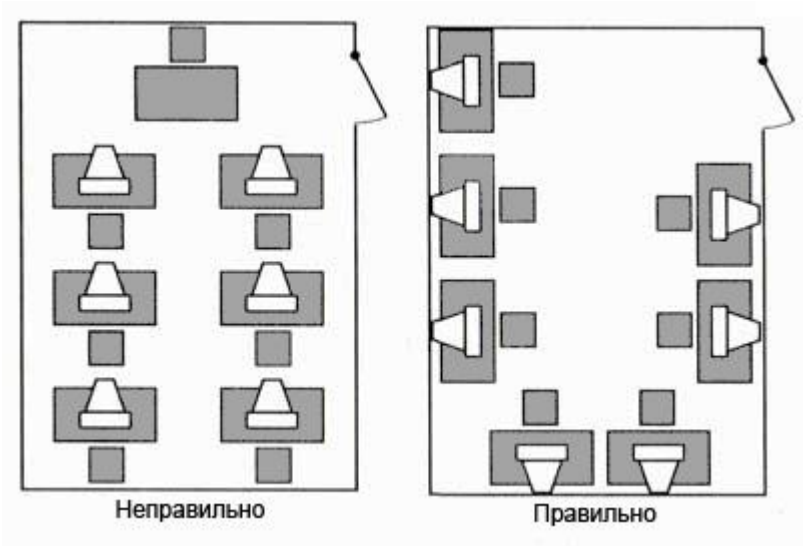
При работе с мышью рука не должна находиться на весу. Локоть руки или хотя бы запястье должны иметь твердую опору. Если предусмотреть необходимое расположение рабочего стола и кресла затруднительно, рекомендуется применить коврик для мыши, имеющий специальный опорный валик. Нередки случаи, когда в поисках опоры для руки (обычно правой) располагают монитор сбоку от пользователя (соответственно, слева), чтобы он работал вполоборота, опирая локоть или запястье правой руки о стол. Этот прием недопустим. Монитор должен обязательно находиться прямо перед пользователем.

Требования к организации занятий.

Экран монитора — не единственный источник вредных электромагнитных излучений.

Монитор компьютера следует располагать так, чтобы задней стенкой он был обращен не к людям, а к стене помещения. В компьютерных классах, имеющих несколько компьютеров, рабочие места должны располагаться по периметру помещения, оставляя свободным центр. При этом дополнительно необходимо проверить каждое из рабочих мест на отсутствие прямого отражения внешних источников освещения. Как правило, добиться этого для всех рабочих мест одновременно достаточно трудно. Возможное решение состоит в использовании штор на окнах и продуманном размещении искусственных источников общего и местного освещения.

Сильными источниками электромагнитных излучений являются устройства бесперебойного питания. Располагать их следует как можно дальше от посадочных мест пользователей.



В организации занятий важную роль играет их продолжительность, от которой зависят психофизиологические нагрузки.

Защита информации, антивирусная защита

Человеку свойственно ошибаться. Любое техническое устройство также подвержено сбоям, поломкам, влиянию помех. Ошибка может произойти при реализации любого информационного процесса. Велика вероятность ошибки при кодировании информации, её обработке и передаче. Результатом ошибки может стать потеря нужных данных, принятие ошибочного решения, аварийная ситуация.

В обществе хранится, передаётся и обрабатывается огромное количество информации и отчасти поэтому современный мир очень хрупок, взаимосвязан и взаимозависим. Информация, циркулирующая в системах управления и связи, способна вызвать крупномасштабные аварии, военные конфликты, дезорганизацию деятельности научных центров и лабораторий, разорение банков и коммерческих организаций. Поэтому информацию нужно уметь защищать от искажения, потери, утечки, нелегального использования.

Пример. В 1983 году произошло наводнение в юго-западной части США. Причиной стал компьютер, в который были введены неверные данные о погоде, в результате чего он дал ошибочный сигнал шлюзам, перекрывающим реку Колорадо.

Пример. В 1971 году на нью-йоркской железной дороге исчезли 352 вагона. Преступник воспользовался информацией вычислительного центра, управляющего работой железной дороги, и изменил адреса назначения вагонов. Нанесённый ущерб составил более миллиона долларов.

Развитие промышленных производств, принесло огромное количество новых знаний, и одновременно возникло желание часть этих знаний хранить от конкурентов, защищать их. Информация давно уже стала продуктом и товаром, который можно купить, продать, обменять на что-то другое. Как и всякий товар, она требует применения специальных методов для обеспечения сохранности.

В информатике в наибольшей степени рассматриваются основные виды защиты информации при работе на компьютере и в телекоммуникационных сетях.

Компьютеры — это технические устройства для быстрой и точной (безошибочной) обработки больших объёмов информации самого разного вида. Но, несмотря на постоянной повышение надёжности их работы, они могут выходить из

строя, ломаться, как и любые другие устройства, созданные человеком. Программное обеспечение также создается людьми, способными ошибаться.

Конструкторы и разработчики аппаратного и программного обеспечения прилагают немало усилий, чтобы обеспечить защиту информации:

- от сбоев оборудования;

- от случайной потери или искажения информации, хранящейся в компьютере;

- от преднамеренного искажения, производимого, например, компьютерными вирусами;

- от несанкционированного (нелегального) доступа к информации (её использования, изменения, распространения).

К многочисленным, далеко не безобидным ошибкам компьютеров добавилась и компьютерная преступность, грозящая перерасти в проблему, экономические, политические и военные последствия которой могут стать катастрофическими.

При защите информации от сбоев оборудования используются *следующие основные методы:*

- периодическое архивирование программ и данных. Причем, под словом «архивирование» понимается как создание простой резервной копии, так и создание копии с предварительным сжатием (компрессией) информации. В последнем случае используются специальные программы-архиваторы (Arj, Rar, Zip и др.);

-автоматическое резервирование файлов. Если об архивировании должен заботиться сам пользователь, то при использовании программ автоматического резервирования команда на сохранение любого файла автоматически дублируется и файл сохраняется на двух автономных носителях (например, на двух винчестерах). Выход из строя одного из них не приводит к потере информации. Резервирование файлов широко используется, в частности, в банковском деле.

Защита от случайной потери или искажения информации, хранящейся в компьютере, сводится к следующим методам:

автоматическому запросу на подтверждение команды, приводящей к изменению содержимого какого-либо файла. Если вы хотите удалить файл или разместить новый файл под именем уже существующего, на экране дисплея появится диалоговое окно с требованием подтверждения команды либо её отмены;

установке специальных атрибутов документов. Например, многие программы-редакторы позволяют сделать документ доступным только для чтения или скрыть файл, сделав недоступным его имя в программах работы с файлами;

возможности отменить последние действия. Если вы редактируете документ, то можете пользоваться функцией отмены последнего действия или группы действий, имеющейся во всех современных редакторах. Если вы ошибочно удалили нужный файл, то специальные программы позволяют его восстановить, правда, только в том случае, когда вы ничего не успели записать поверх удаленного файла;

разграничению доступа пользователей к ресурсам файловой системы, строгому разделению системного и пользовательского режимов работы вычислительной системы.

Защита информации от преднамеренного искажения часто еще называется защитой от ***вандализма.***

Проблема вандализма заключается в появлении таких бедствий, как компьютерные вирусы и компьютерные червяки. Оба этих термина придуманы более для привлечения внимания общественности к проблеме, а не для обозначения некоторых приёмов вандализма.

Компьютерный вирус представляет собой специально написанный небольшой по размерам фрагмент программы, который может присоединяться к другим программам (файлам) в компьютерной системе. Например, вирус может вставить себя в начало некоторой программы, так что каждый раз при выполнении этой программы первым будет активизироваться вирус. Во время выполнения вирус может производить намеренную порчу, которая сейчас же становится заметной, или просто искать другие программы, к которым он может присоединить свои копии. Если «заражённая» программа будет передана на другой компьютер через сеть или дискету, вирус начнёт заражать программы на новой машине, как только будет запущена переданная программа. Таким способом вирус переходит от машины к машине. В некоторых случаях вирусы потихоньку распространяются на другие программы и не проявляют себя, пока не произойдёт определённое событие, например, наступит заданная дата, начиная с которой они будут «разрушать» всё вокруг. Разновидностей компьютерных вирусов очень много. Среди них встречаются и невидимые, и самомодифицирующиеся.

Термин ***«червяк»*** обычно относится к автономной программе, которая копирует себя по всей сети, размещаясь в разных машинах. Как и вирусы, эти программы могут быть спроектированы для самотиражирования и для проведения «диверсий».

Для защиты от вирусов можно использовать:

- общие методы защиты информации, которые полезны также как страховка от физической порчи дисков, неправильно работающих программ или ошибочных действий пользователя;

- профилактические меры, позволяющие уменьшить вероятность заражения вирусом;

- специализированные антивирусные программы.

Многие методы защиты информации от несанкционированного (нелегального) доступа возникли задолго до появления компьютеров.

Одним из таких методов является ***шифрование.***

Проблема защиты информации путем её преобразования, исключающего её прочтение посторонним лицом, волновала человеческий ум с давних времен. История криптологии (kryptos — тайный, logos — наука) — ровесница истории человеческого языка. Более того, письменность сама по себе была вначале криптографической системой, так как в древних обществах ею владели только избранные. Священные книги Древнего Египта, Древней Индии тому примеры. Криптология разделяется на два направления — криптографию и криптоанализ. Цели этих направлений прямо противоположны. Криптография занимается поиском и исследованием методов шифрования информации. Она даёт возможность преобразовывать информацию таким образом, что её прочтение (восстановление) возможно только при знании ключа. Сфера интересов криптоанализа — исследование возможностей расшифровки информации без знания ключей.

Ключ — информация, необходимая для беспрепятственного шифрования и дешифрования текста.

Первые криптографические системы встречаются уже в начале нашей эры. Так, Цезарь в своей переписке уже использовал шифр, получивший его имя. Бурное развитие криптографические системы получили в годы первой и второй мировых войн. Появление вычислительной техники ускорило разработку и совершенствование криптографических методов.

Основные направления использования этих методов — передача конфиденциальной информации по каналам связи (например, по электронной почте), установление подлинности передаваемых сообщений, хранение информации (документов, баз данных) на носителях в зашифрованном виде.

Проблема использования криптографических методов в современных информационных системах становится в настоящее время особенно актуальной. С

одной стороны, расширилось использование телекоммуникационных сетей, по которым передаются большие объёмы информации государственного, коммерческого, военного и частного характера, не допускающего возможность доступа к ней посторонних лиц.С другой стороны, появление новых мощных аппаратных и программных средств, эффективных технологий дешифрования снизило надёжность криптографических систем, ещё недавно считавшихся практически нераскрываемыми.

Другим возможным методом защиты информации от несанкционированного доступа является ***применение паролей.***

Пароли позволяют контролировать доступ как к компьютерам, так и к отдельным программам или файлам. К сожалению, иногда пароль удается угадать, тем более, что многие пользователи в качестве паролей используют свои имена, имена близких, даты рождения.

Существуют программные средства от «вскрытия» паролей. Чтобы противостоять попыткам угадать пароль, операционные системы могут быть спроектированы таким образом, чтобы отслеживать случаи, когда кто-то многократно употребляет неподходящие пароли (первый признак подбора чужого пароля). Кроме того, операционная система может сообщать каждому пользователю в начале его Сеанса, когда в последний раз использовалась его учётная запись. Этот метод позволяет пользователю обнаружить случаи, когда кто-то работал в системе под его именем. Более сложная защита (называемая ловушкой) — это создание у взломщика иллюзии успешного доступа к информации на время, пока идет анализ, откуда появился этот взломщик.

Одной из распространённых форм нарушения информационного права является незаконное копирование программ и данных, в частности находящихся на коммерчески распространяемых носителях информации.

Для предотвращения нелегального копирования файлов используются специальные программно-аппаратные средства, например «электронные замки», позволяющие сделать с дискеты не более установленного числа копий, или дающие возможность работать с программой только при условии, что к специальному разъёму системного блока подключено устройство (обычно микросхема), поставляемое вместе с легальными копиями программ. Существуют и другие методы защиты, в частности, административные и правоохранительные.

Выполнение практической работы не предусматривает выставление оценок

Вопросы для самопроверки

1. Перечислите основные негативные последствия работы за монитором
2. Объясните цель эргономики
3. Что является сильными источниками электромагнитных излучений?
4. Перечислите основные методы, используемые при защите информации от сбоев оборудования.
5. Что такое «Червяк»?
6. Какие методы применяют для защиты от вирусов?

Практическое занятие № 3 Основные этапы развития информационного общества. Информационные ресурсы общества. Работа с ними

Цель занятия:

1. Ознакомиться с основными этапами развития информационного общества, этапами развития технических средств и информационных ресурсов.

2. Научиться работать с документами в локальной сети.

3. Научиться выполнять поиск информации по теме.

Исходные данные:

Папка на РС «Практическое занятие № 3»

Статья «Концепция формирования информационного общества»

Файл «Понятие информационных ресурсов».

Содержание и порядок выполнения задания:

Изучите статью «Концепция формирования информационного общества»

Выполните задания.

Теоретическая часть:

Этапы развития информационного общества.

В развитии человечества существуют четыре этапа, названные информационными революциями, которые внесли изменения в его развитие.

1. **Первый этап – связан с изобретением письменности.** Это обусловило качественный гигантский и количественный скачек в развитии общества. Знания стало возможно накапливать и передавать последующим поколениям, т.е. появились средства и методы накопления информации. В некоторых источниках считается, что содержание первой информационной революции составляет распространение и внедрение в деятельность и сознание человека языка.

2. **Второй этап – изобретение книгопечатания.** Это дало в руки человечеству новый способ хранения информации, а так же сделало более доступным культурные ценности.

3. **Третий этап– изобретение электричества.** Появились телеграф, телефон и радио, позволяющие быстро передавать и накапливать информацию в любом объеме. Появились средства информационных коммуникаций.

4. **Четвертый этап – изобретение микропроцессорной технологии и персональных компьютеров.** Толчком к этой революции послужило создание в середине 40-х годов ЭВМ. Эта последняя революция дала толчок человеческой цивилизации для перехода от индустриального к информационному обществу-обществу, в котором большинство работающих занято производством, хранением, переработкой и реализацией информации, особенно высшей ее формой – знанием. Началом этого послужило внедрение в различные сферы деятельности человека современных средств обработки и передачи информации – этот процесс называется информатизацией

Основные черты информационного общества.

Информационное общество — общество, в котором большинство работающих занято производством, хранением, переработкой и реализацией информации, особенно высшей её формы — знаний.

Некоторые <u>характерные черты информационного общества:</u>

1. Объёмы информации возрастут и человек будет привлекать для её обработки и хранения специальные технические средства.
2. Неизбежно использование компьютеров.
3. Движущей силой общества станет производство информационного продукта.
4. Увеличится доля умственного труда, т.к. продуктом производства в информационном обществе станут знания и интеллект.
5. Произойдёт переоценка ценностей, уклада жизни и изменится культурный досуг.
6. Развиваются компьютерная техника, компьютерные сети, информационные технологии.
7. У людей дома появляются всевозможные электронные приборы и компьютеризированные устройства.
8. Производством энергии и материальных продуктов будут заниматься машины, а человек главным образом обработкой информации.
9. В сфере образования буде создана система непрерывного образования.
10. Дети и взрослые смогут обучаться на дому с помощью компьютерных программ и телекоммуникаций.
11. Появляется и развивается рынок информационных услуг.

Задание 1

Заполните таблицу словами, которые на ваш взгляд, связаны по смыслу с приведенными в первом столбце понятиями (терминами)

Понятие (термины)	Слова, связанные по смыслу
Информация	
Информатика	
Наука	

Задание 2

Заполните таблицу «Информационные революции»

Информационная революция	Период времени	Радикальные изменения в истории человечества	Основные изобретения (место, изобретатели)
Первая			
Вторая			
Третья			
Четвертая			
Пятая			

Задание 3

Заполните таблицу «Развитие информационных технологий»

Этап	Время	Информационная технология	Инструментарий	Коммуникация, связь
1				
2				
3				
4				
5				

Задание 4

Работа с документом «Концепция формирования информационного общества в России»

Выясните особенности формирования информационного общества в России, изучив статью «Концепция формирования информационного общества».

Заполните таблицу особенностей и выявите положительные и негативные особенности.

Положительная особенность	Негативная особенность

Задание 5

Изучите понятие информационных ресурсов, классификацию информационных ресурсов.

Материал находится в файле «Понятие информационных ресурсов». Ответьте на вопросы.

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Определение информационных ресурсов.
2. Запишите свою точку зрения, в чем принципиальное различие между информационными и всякими другими ресурсами?
3. Как называют информационные ресурсы, собираемые со всего мира?
4. Виды мировых информационных ресурсов?
5. Назовите национальные информационные ресурсы.

Практическое занятие № 4 Образовательные информационные ресурсы КМРК. Электронная библиотека

Цель занятия: научиться пользоваться образовательными информационными ресурсами, искать нужную информацию с их помощью.

*Исходные данные:*компьютер с выходом в интернет

Содержание и порядок выполнения задания:

Изучите теоретическую часть

Выполните вход на сайт do.kmrk.ru

Теоретическая часть

Под образовательными информационными ресурсами понимают текстовую, графическую и мультимедийную информацию, а также исполняемые программы (дистрибутивы), то есть электронные ресурсы, созданные специально для использования в процессе обучения на определенной ступени образования и для определённой предметной области.

К образовательным электронным ресурсам можно отнести: – учебные материалы (электронные учебники, учебные пособия, рефераты, дипломы), – учебно-методические материалы (электронные методики, учебные программы), – научно-методические (диссертации, кандидатские работы), – дополнительные текстовые и иллюстративные материалы (лабораторные работы, лекции), – системы тестирования (тесты – электронная проверка знаний), – электронные полнотекстовые библиотеки; – электронные периодические издания сферы образования; – электронные оглавления и аннотации статей периодических изданий сферы образования, – электронные архивы выпусков.

При работе с образовательными ресурсами появляются такие понятия, как субъект и объект этих ресурсов. Субъекты информационной деятельности классифицируются следующим образом:

– субъект, создающий объекты (все пользователи образовательной системы - преподаватель, студент);

– субъект, использующий объекты (все пользователи образовательной системы);

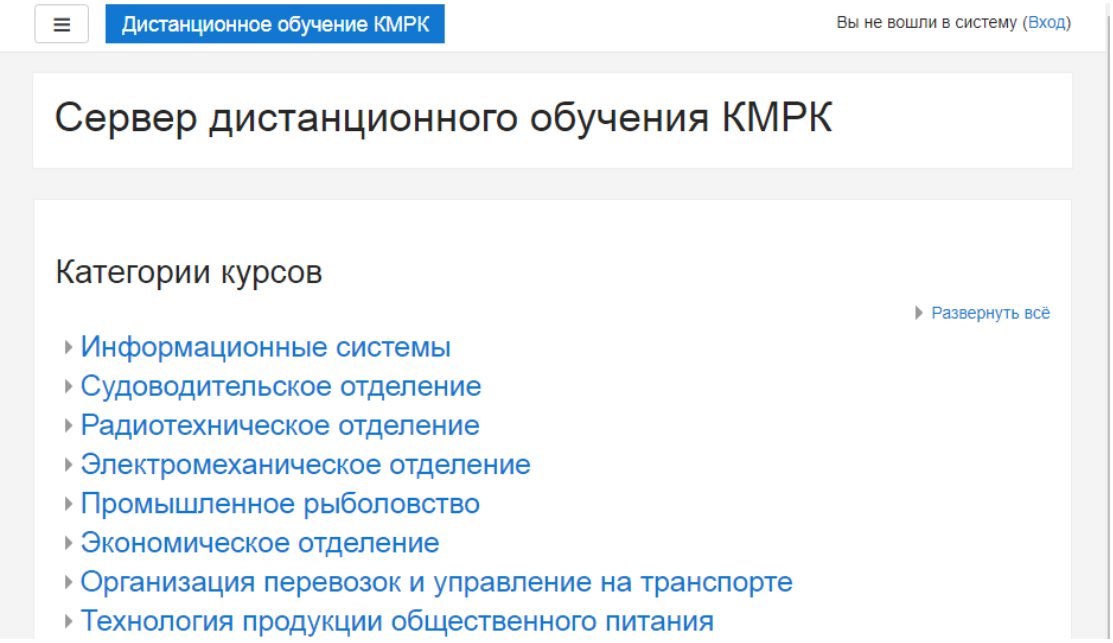
– субъект, администрирующий объекты, то есть обеспечивающий среду работы с объектами других субъектов (администраторы сети);

– субъект, контролирующий использование объектов субъектами (инженеры).

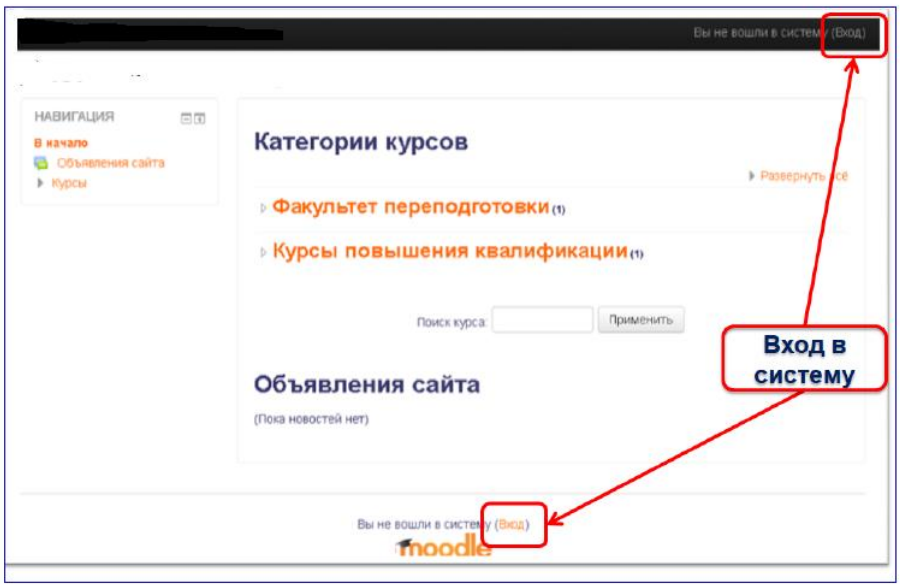
Образовательные ресурсы КМРК – сайт do.kmrk.ru – сервер дистанционного обучения.

Вход на сервер дистанционного обучения КМРК

Введите адрес сервера ДО КМРК http://do.kmrk.ru/



В правом верхнем углу расположена информация Вы не вошли в систему. Нажмите Вход.



В появившиеся поля ввести свои логин и пароль и после аутентификации (проверки, имеет ли пользователем с заявленным логином право на доступ в систему) войти в систему. После того, как Вы зайдете в Систему, в верхнем правом углу появится строка с Вашим именем «Вы зашли под именем: Ваши имя и фамилия».В центре страницы располагаются курсы, к которым вы имеете доступ. Чтобы зайти в курс, необходимо кликнуть мышью по его названию.

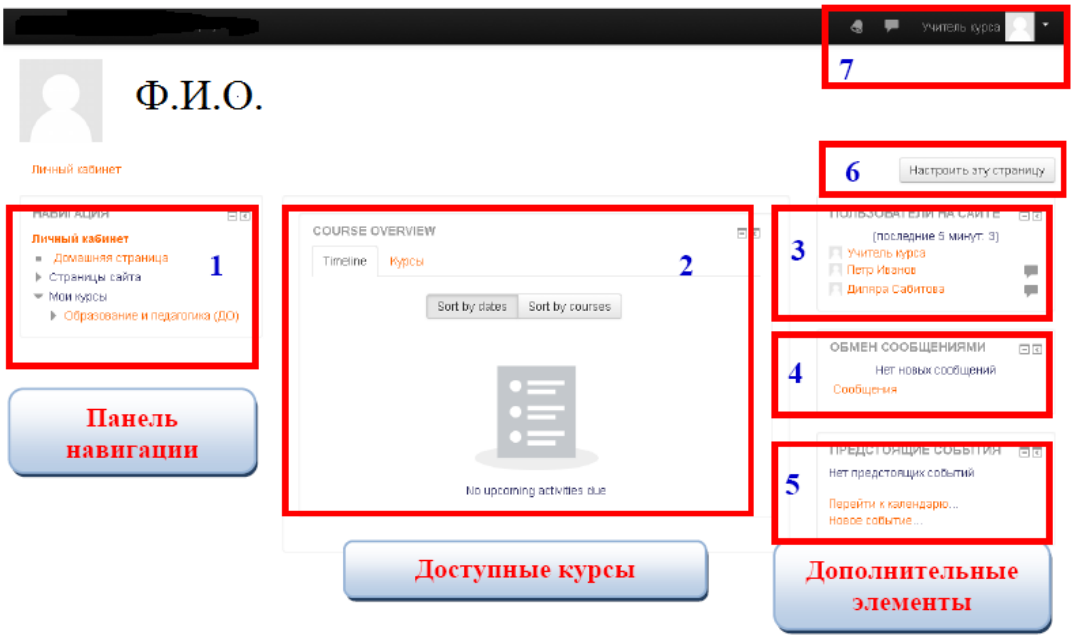
Личный профиль пользователя

После авторизации открывается Личный кабинет пользователя. В домашней странице Личного кабинета пользователю доступен только

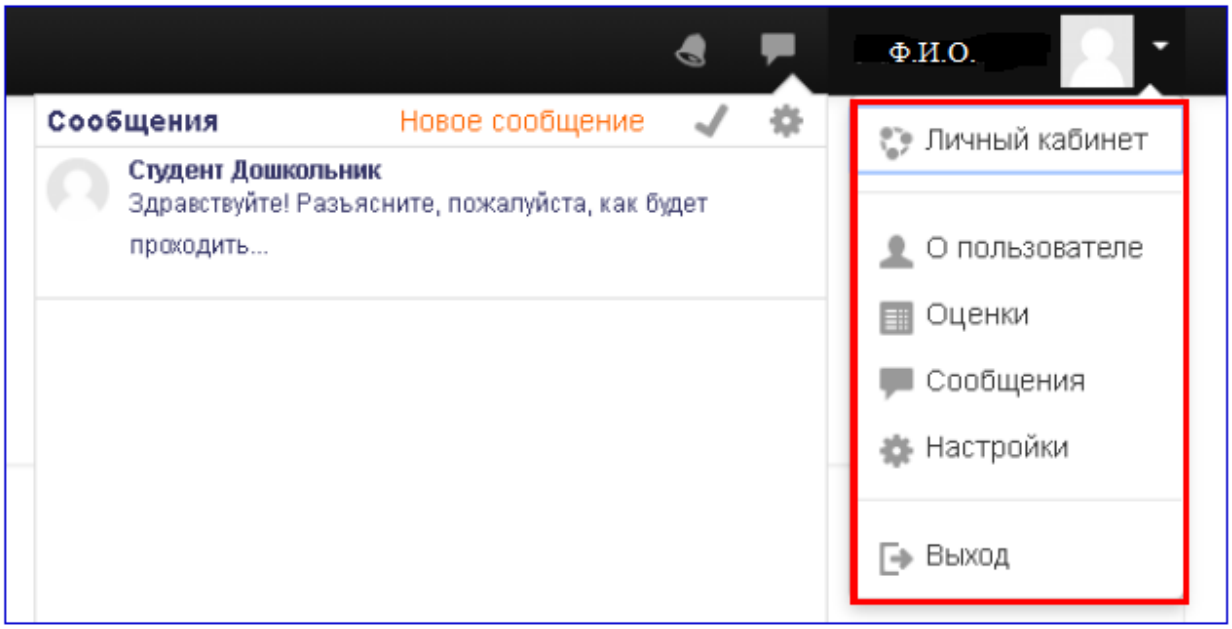
тот курс, на который он записан с ролью слушателя или преподавателя курса**.**

Интерфейс Личного кабинета пользователя интуитивно понятен. Страница разделена на три вертикальные области: панель навигации (на рисунке ниже - цифра 1), панель, в которой отображаются доступные курсы и расписание (цифра 2) и область дополнительных элементов. Дополнительные элементы по умолчанию состоят из блоков: **Пользователи на сайте** (цифра 3), **Обмен сообщениями** (цифра 4), и **Предстоящие события** (цифра 5).

Страницу Личного кабинета пользователь СДО может настроить по собственному усмотрению. Для этого предназначена кнопка **Настроить эту страницу,** расположенная в верхнем правом углу (цифра 6).



В правом верхнем углу строки заголовка системы дистанционного обучения расположены элементы, позволяющие осуществить быстрый переход в личный кабинет, на страницу информации о пользователе, к оценкам, к сообщениям, к уведомлениям и настройкам. Здесь же пользователь осуществляет выход из системы.



Вся информация о пользователе хранится в его профиле. Пользователь с ролью «слушатель» может просматривать только доступную информацию о любом пользователе. Для этого достаточно кликнуть мышью на его фамилии и имени на любой странице сайта.

Редактировать информацию о себе может любой пользователь. Кликнув мышью на ссылку «О пользователе» можно увидеть личную информацию так, как ее видят все посетители сайта.

Изменение личных данных по ссылке «Настройки» - «Редактировать информацию» доступно только самому пользователю и администратору сайта.

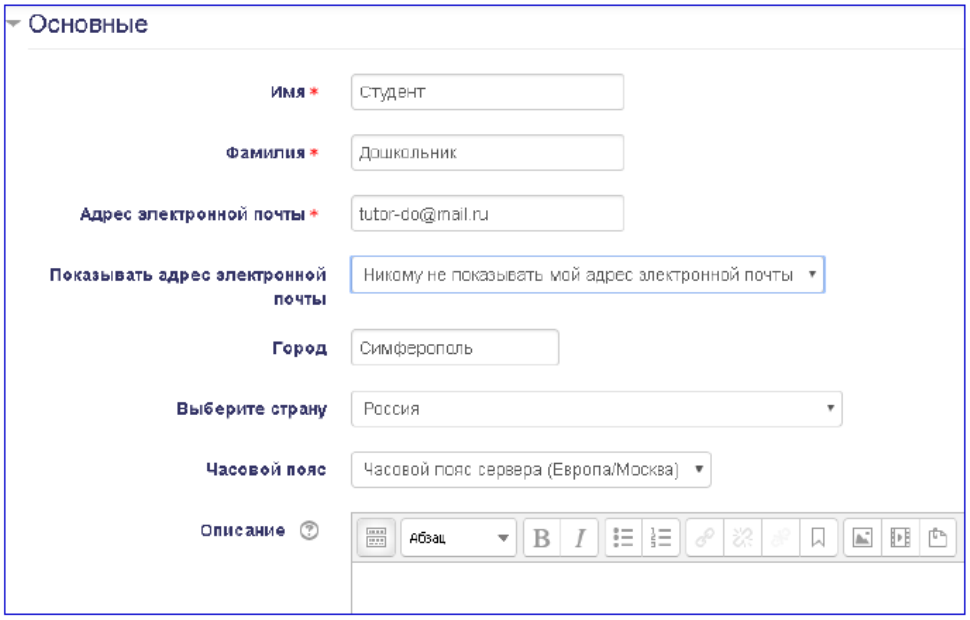
Страница профиля для редактирования информации содержит обязательные поля и поля, которые пользователь может заполнять по собственному усмотрению. Три основных обязательных поля заполняются автоматически при регистрации пользователя администратором сайта. Это поля:

− имя;

− фамилия;

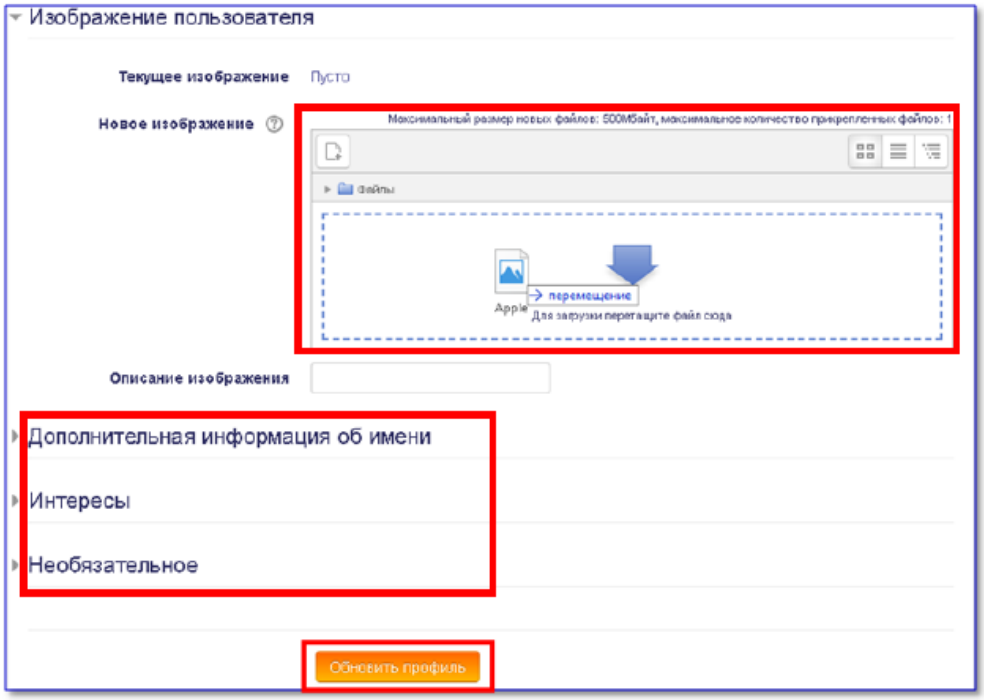
− адрес электронной почты.

Остальные поля можно редактировать. Рассмотрим поля, которые пользователь заполняет и редактирует самостоятельно.

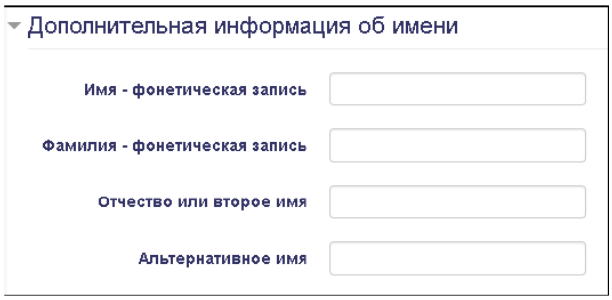


Изображение пользователя: добавляется пользователем по желанию.

Для добавления фотографии кликните по значку в разделе «Изображение пользователя» или перетащите файл с Вашим изображением в поле для загрузки. Формат графического файла JPG или PNG. Изображение будет обрезано до квадрата размером 100x100 пикселей.



При необходимости внесения дополнительной информации в профиль, раскройте соответствующие поля.

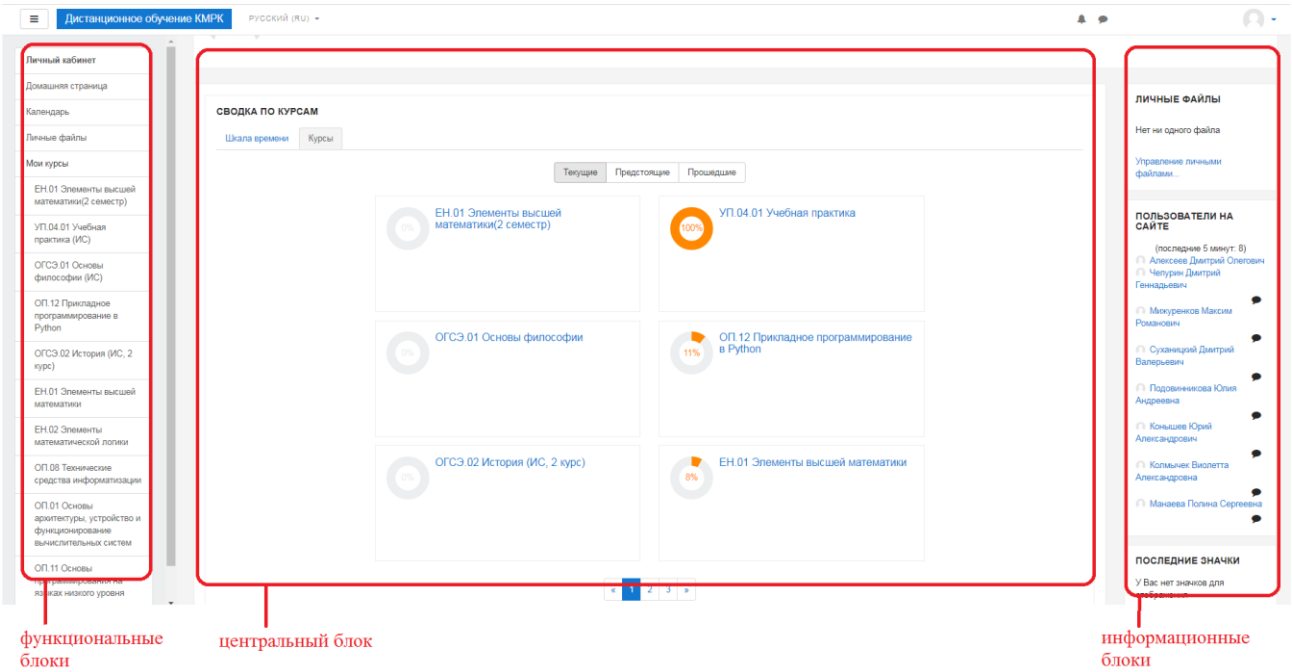


Дополнительная информация об имени: обязательным для любого пользователя является заполнение поля Отчество.

Внешний вид курсов и навигация

Переход по ссылке с названием курса открывает страницу содержимого учебной программы.

В панели навигации отображается структура курса. Переход по любой ссылке в левой панели открывает страницу с содержимым соответствующего модуля. (Пользователь может перейти к любому элементу модуля непосредственно со страницы курса или с панели навигации).



Функциональные блоки:

Элементы курса - блок содержит категории тех элементов, которые доступны в настоящий момент в курсе (форумы, ресурсы, задания, тесты и т.д.);

Мои курсы – отображает список доступных пользователю курсов.

Информационные блоки:

Новостной форум – отображает последние сообщения, которые появились в Новостном форуме.

Календарь – содержит даты, связанные с графиком выполнения учебных заданий, дни отмечены разными цветами.

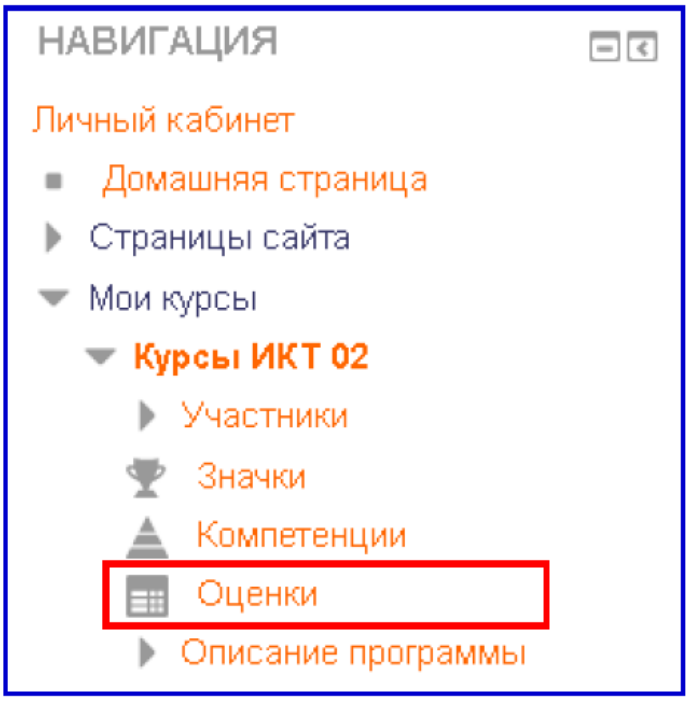
Пользователи на сайте -отображается список участников курса с указанием времени последнего посещения.

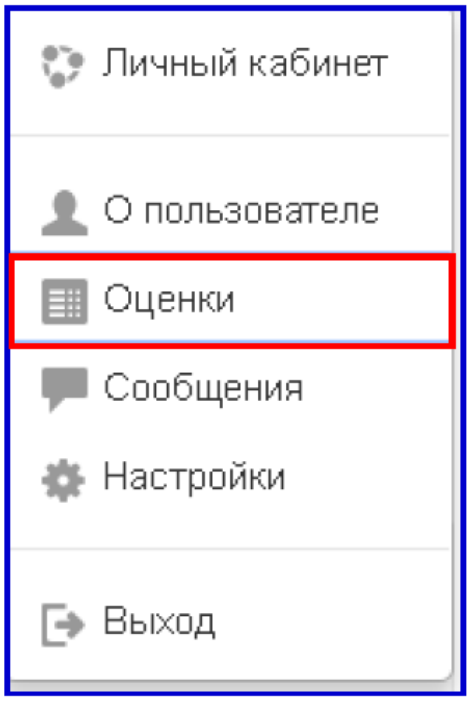
Центральный блок:

Отображается структура курса. Переход по любой ссылке в левой панели открывает страницу с содержимым соответствующего модуля. Пользователь может перейти к любому элементу модуля непосредственно со страницы курса или с панели навигации.

Предметный модуль разбит на темы. Каждая тема может включать материалы лекций, презентационные материалы, задания для практической работы, контрольные вопросы для самопроверки, рекомендуемые информационные источники и итоговый контроль. Оценки за выполненные задания заносятся в таблицу оценок автоматически или преподавателем, курирующим соответствующий модуль.

Информацию об оценках, полученных за освоение программы обучения, каждый пользователь может просмотреть, выбрав пункт «Оценки» в панели навигации курса, или щелкнув на такой же пункт выпадающего меню своего профиля:





Обучение на курсе

Дистанционный курс – это набор тематических модулей, в которых размещены ресурсы и интерактивные элементы курса.

Ресурсы – это представление теоретического материала курса. Ими могут быть: тексты лекций; иллюстративный материал (карты, схемы, диаграммы, формулы, веб-страницы); аудио- и видеофайлы; анимационные ролики, ссылки на ресурсы Интернета и т.п.

Учебные материалы необходимо выполнять последовательно, осваивая их в сроки, указанные преподавателем. Информацию можно прочитать с экрана, распечатать или сохранить ее на свой компьютер.

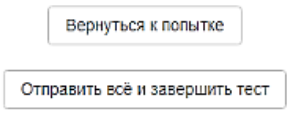
Интерактивные элементы позволяют акцентировать внимание на отдельных фрагментах изучаемого материала, проверить уровень знаний, организовать взаимодействие студентов друг с другом и с преподавателем. К элементам курса относятся: лекции, рабочие тетради, задания различных типов, глоссарии (словари по курсу), форумы, чаты, опросы, тесты.

Работа на сайте:

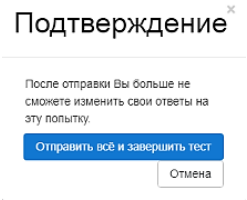
1. Пройдите тест по теме. Для этого:

- Перейдите по ссылке Тест по теме:
- Нажмите кнопку: Начать тестирование.
- Ответьте на вопросы теста

- В конце тестирования нажмите кнопку **Закончить попытку**
- Нажмите кнопку **Отправить все и завершить тест**



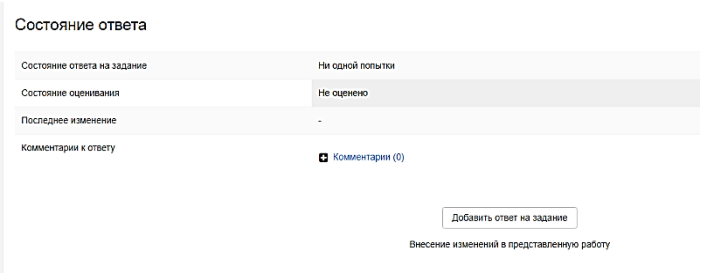
- Подтвердите отправку теста.



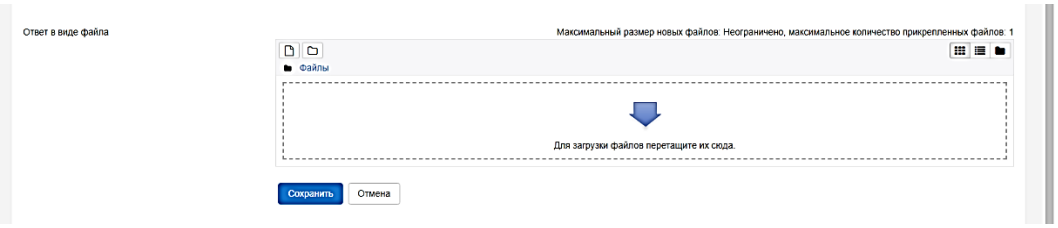
- Вернитесь на курс

2. Практические работы предполагают отправку преподавателю выполненной работы. Для этого:

- Перейдите на Практическая работа № ...
- Нажмите кнопку Добавить ответ на задание.



- Перетащите свой файл в область со стрелкой



- Нажмите кнопку **Сохранить**

Выводы и предложения о проделанной работе

Содержание отчета:

Выполнение практической работы не предусматривает оценивание.

Вопросы для самопроверки:

1 Что такое информационные ресурсы?

2 Что такое образовательные информационные ресурсы?

3 Что относится к образовательным информационным ресурсам?

4 Каковы субъекты и объекты образовательных информационных ресурсов?

Практическое занятие № 5Представление информации в различных системах счисления

Цель занятия:

Изучение систем счисления и получение практических навыков перевода чисел из одной системы счисления в другую.

Исходные данные:

Таблица, правила перевода систем счисления.

Содержание и порядок выполнения задания:

Изучите теоретическую часть

Выполните задания.

Теоретическая часть

Представление числовой информации с помощью систем счисления

Для записи информации о количестве объектов используются числа. Числа записываются с использованием особых знаковых систем, которые называются системами счисления. Алфавит систем счисления состоит из символов, которые называются цифрами. Например, в десятичной системе счисления числа записываются с помощью десяти всем хорошо известных цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Система счисления - это знаковая система, в которой числа записываются по определенным правилам с помощью символов некоторого алфавита, называемых цифрами.

Все системы счисления делятся на две большие группы: *позиционные* и *непозиционные* системы счисления. В позиционных системах счисления значение цифры зависит от ее положения в числе, а в непозиционных - не зависит.

Римская непозиционная система счисления. Самой распространенной из непозиционных систем счисления является римская. В качестве цифр в ней используются **X={I (1), V (5), X (10), L (50), C (100), D (500), M (1000)}**, где в скобках указаны веса символов (не зависящие от позиции символа)

Примеры римских чисел (в скобках – обычные десятичные эквиваленты):

III (3), IV (4), V (5), VI (6), IX (9), XI (11), DCL (650).

Значение цифры не зависит от ее положения в числе. Например, в числе XXX (30) цифра X встречается трижды и в каждом случае обозначает одну и ту же величину - число 10, три числа по 10 в сумме дают 30.

Позиционные системы счисления. Первая позиционная система счисления была придумана еще в Древнем Вавилоне, причем вавилонская нумерация была шестидесятеричной, то есть в ней использовалось шестьдесят цифр! Интересно, что до сих пор при измерении времени мы используем основание, равное 60 (в 1 минуте содержится 60 секунд, а в 1 часе - 60 минут).

В XIX веке довольно широкое распространение получила двенадцатеричная система счисления. До сих пор мы часто употребляем дюжину (число 12): в сутках две дюжины часов, круг содержит тридцать дюжин градусов и так далее.

*В **позиционных системах счисления** количественное значение цифры зависит от ее позиции в числе.*

Наиболее распространенными в настоящее время позиционными системами счисления являются десятичная, двоичная, восьмеричная и шестнадцатеричная. Каждая позиционная система имеет определенный *алфавит цифр* и *основание*.

*В **позиционных системах счисления** основание системы равно количеству цифр (знаков в ее алфавите) и определяет, во сколько раз различаются значения одинаковых цифр, стоящих в соседних позициях числа.*

Десятичная система счисления имеет алфавит цифр, который состоит из десяти всем известных, так называемых арабских, цифр, и основание, равное 10, двоичная - две цифры и основание 2, восьмеричная - восемь цифр и основание 8, шестнадцатеричная - шестнадцать цифр (в качестве цифр используются и буквы латинского алфавита) и основание 16 (табл.).

Таблица – Позиционные системы счисления

Основание	**Система счисления**	**Знаки**
2	*Двоичная*	над алфавитом **X = {0,1}**;
3	*Троичная*	над **X = {0,1,2}**;
8	*Восьмеричная*	над **X = {0, 1, 2, 3, 4, 5, 6, 7}**;
10	Десятеричная	над **X = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};**
16	*Шестнадцатеричная*	над **X = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}**, где символы **A, B, C, D, E, F** имеют, соответственно, десятичные

В позиционных системах чем больше основание системы, тем меньшее количество разрядов (то есть записываемых цифр) требуется при записи числа.

Арабская система счисления, которым мы пользуемся в повседневной жизни, является позиционной. В позиционных системах счисления позиция числа однозначно определяет величину числа. Рассмотрим это на примере числа 6372 в десятичной системе счисления. Пронумеруем это число,справа, налево начиная с нуля:

Число	6	3	7	2
Позиция	3	2	1	0

Тогда число 6372 можно представить в развернутом виде:

$6372=6000+300+70+2 =6\cdot10^3+3\cdot10^2+7\cdot10^1+2\cdot10^0$.

Число 10 определяет систему счисления. В качестве степеней взяты значения позиции данного числа.

Рассмотрим вещественное десятичное число 1287,923. Пронумеруем его, начиная с нуля позиции числа от десятичной точки влево и вправо:

Число	1	2	8	7	.	9	2	3
Позиция	3	2	1	0	.	-1	-2	-3

Тогда число 1287.923 можно представить в виде:

$1287.923=1000+200+80+7+0.9+0.02+0.003=1\cdot10^3+2\cdot10^2+8\cdot10^1+7\cdot10^0+9\cdot10^{-1}+2\cdot10^{-2}+$
$+3\cdot10^{-3}$.

В вычислительных машинах используется двоичная система счисления, её основание - число 2. Для записи чисел в этой системе используют только две цифры - 0 и 1.

Выбор двоичной системы для применения в вычислительной технике объясняется тем, что электронные элементы - триггеры, из которых состоят микросхемы ЭВМ, могут находиться только в двух рабочих состояниях.

Двоичная система удобна для компьютера, но неудобна для человека: числа получаются длинными и их трудно записывать и запоминать. Конечно, можно перевести число в десятичную систему и записывать в таком виде, а потом, когда понадобится перевести обратно, но все эти переводы трудоёмки. Поэтому применяются системы счисления, родственные двоичной - восьмеричная и шестнадцатеричная. Для записи чисел в этих системах требуется соответственно 8 и 16 цифр. В шестнадцатеричной системе счисления первые десять цифр общие, а

дальше используют заглавные латинские буквы. Шестнадцатеричная цифра A соответствует десятеричному числу 10, шестнадцатеричная B – десятичному числу 11 и т. д. Использование этих систем объясняется тем, что переход к записи числа в любой из этих систем от его двоичной записи очень прост. В таблице 1 приведена таблица соответствия чисел, записанных в разных системах.

Таблица 1

Двоичные числа	Восьмеричные числа	Десятичные числа	Шестнадцатеричные числа
0	0	0	0
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	10	A
1011	13	11	B
1100	14	12	C
1101	15	13	D
1110	16	14	E
1111	17	15	F

Перевод чисел из произвольной системы счисления в десятичную систему счисления и обратно

Перевод чисел из одной системы счисления в другую составляет важную часть машинной арифметики. В вычислительной технике применяют позиционные системы счисления с недесятичным основанием: двоичную, восьмеричную, шестнадцатеричную. Для обозначения используемой системы счисления число снабжают верхним или нижним индексом, в котором записывают основание системы счисления.

16-ричная и 8-ричная система счисления используются при составлении программ на языке машинных кодов для более короткой и удобной записи двоичных кодов – команд, данных, адресов и операндов. Задача перевода из одной системы счисления в другую часто встречается при программировании.

Если основание используемой системы счисления больше десяти, то для цифр вводят буквенное обозначение.

В шестнадцатеричной системе счисления основа - это цифры 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 с соответствующими обозначениями 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Примеры чисел: 17D, CA, F12A.

Двоичная система счисления- это система, в которой для записи чисел используются две цифры 0 и 1. Основанием двоичной системы счисления является число 2. Двоичный или бинарный код числа - запись этого числа в двоичной системе счисления. Например,

$0=0_2$

$1=1_2$

$2=10_2$

$3=11_2$

$7=111_2$

$120=1111000_2$.

Для перевода целого числа из десятичной системы счисления в s-ичную необходимо последовательно делить это число и получаемые частные на s (по правилам системы счисления с основанием s) до тех пор, пока частное не станет равным нулю. Старшей цифрой в записи числа с основанием s служит последний остаток, а следующие за ней цифры образуют остатки от предшествующих делений, выписываемые в последовательности, обратной их получению.

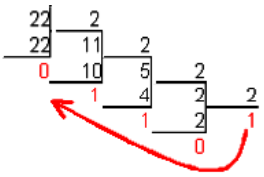
Рассмотрим основные правила перевода:

✓Для перевода десятичного числа в двоичную систему его необходимо последовательно делить на 2 до тех пор, пока не останется остаток, меньший или равный 1. Число в двоичной системе записывается как последовательность последнего результата деления и остатков от деления в обратном порядке.

Пример 1. Число 22_{10} перевести в двоичную систему счисления.

Решение:

I способ:



$22_{10}=10110_2$

II способ:

22_{10} =11·2(0)

11 = 5 ·2(1)

5 = 2·2(1)

2=1·2(0)

$22_{10} = 10110_2$

✓Для перевода двоичного числа в десятичное необходимо его записать в виде многочлена (то есть в развернутом виде), состоящего из произведений цифр числа и соответствующей степени числа 2, и вычислить по правилам десятичной арифметики:

$X_2=a_n \cdot 2^{n-1} + a_{n-1} \cdot 2^{n-2} + a_{n-2} \cdot 2^{n-3} + a_n \cdot 2^{n-1} + \ldots + a_2 \cdot 2^1 + +a_1 \cdot 2^0$

Пример2. Число 11101000_2перевести в десятичную систему счисления.

Решение:

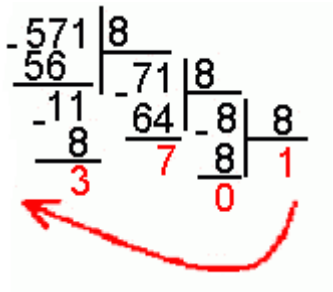
$11101000_2=1 \cdot 2^7+1 \cdot 2^6+1 \cdot 2^5+0 \cdot 2^4+1 \cdot 2^3+0 \cdot 2^2+0 \cdot 2^1+0 \cdot 2^0=232_{10}$

✓Для перевода десятичного числа в восьмеричную систему его необходимо последовательно делить на 8 до тех пор, пока не останется остаток, меньший 8. Число в восьмеричной системе записывается как последовательность цифр последнего результата деления и остатков от деления в обратном порядке.

Пример3. Число 57110 перевести в восьмеричную систему счисления.

Решение:

I способ:



$571_{10} = 1073_8$

II способ:

57110 = 71·8(3)

71 = 8 ·8(7)

8 = 1·8(0)

$571_{10} = 1073_8$

✓Для перевода восьмеричного числа в десятичное число необходимо его записать в виде многочлена, состоящего из произведений цифр числа и соответствующей степени числа 8, и вычислить по правилам десятичной арифметики:

$X_8=a_n \cdot 8^{n-1} + a_{n-1} \cdot 8^{n-2} + a_{n-2} \cdot 8^{n-3} + a_n \cdot 8^{n-1} + \ldots + a_2 \cdot 8^1 + a_1 \cdot 8^0$

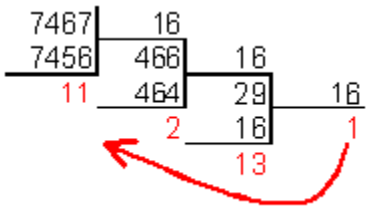
Пример 4. Число 5013_8 перевести в десятичную систему счисления.

Решение: $5013_8 = 5 \cdot 8^3 + 0 \cdot 8^2 + 1 \cdot 8^1 + 3 \cdot 8^0 = 2571_{10}$

Для перевода десятичного числа в шестнадцатеричную систему его необходимо последовательно делить на 16 до тех пор, пока не останется остаток, меньший 16. Число в шестнадцатеричной системе записывается как последовательность цифр последнего результата деления и остатков от деления в обратном порядке.

Пример 5. Число 7467_{10} перевести в шестнадцатеричную систему счисления.

Решение: I способ:



$7467_{10}=1D2B_{16}$

II способ:

$7467_{10} = 466 \cdot 16(11)$

$466 = 29 \cdot 16(1)$

$29=13 \cdot 16(0)$

$7467_{10}=1D2B_{16}$

Для перевода шестнадцатеричного числа в десятичное необходимо его записать в виде многочлена, состоящего из произведений цифр числа и соответствующей степени числа 16, и вычислить по правилам десятичной арифметики:

$X_{16}=a_n \cdot 16^{n-1} + a_{n-1} \cdot 16^{n-2} + a_{n-2} \cdot 16^{n-3} + a_n \cdot 16^{n-1} + \ldots + a_2 \cdot 16^1 + + a_1 \cdot 16^0$

Пример 6. Число $FDA1_{16}$ перевести в десятичную систему счисления.

Решение:

$FDA1_{16}=15\cdot16^3+13\cdot16^2+10\cdot16^1+1\cdot16^0=64929_{10}$

✓Чтобы перевести число из двоичной системы в восьмеричную, его нужно разбить на триады (тройки цифр), начиная с младшего разряда, в случае необходимости дополнив старшую триаду нулями, и каждую триаду заменить соответствующей восьмеричной цифрой (таблица 1).

Пример 7. Число1001011_2 перевести в восьмеричную систему счисления.

Решение: делим число, начиная с конца на триады и заменяем их восьмеричными числами используя таблицу 1

$001\ 001\ 011_2 = 113_8$

✓Чтобы перевести число из двоичной системы в шестнадцатеричную, его нужно разбить на тетрады (четверки цифр), начиная с младшего разряда, в случае необходимости дополнив старшую тетраду нулями, и каждую тетраду заменить соответствующей шестнадцатеричной цифрой (таблица 1).

Пример8. Число1011100011_2 перевести в шестнадцатеричную систему счисления.

Решение: $0010\ 1110\ 0011_2=2E3_{16}$

Для перевода восьмеричного числа в двоичное необходимо каждую цифру заменить эквивалентной ей двоичной триадой (таблица 1).

Пример9. Число531_8 перевести в двоичную систему счисления.

Решение: $527_8= 101\ 010\ 111_2$

Для перевода шестнадцатеричного числа в двоичное необходимо каждую цифру заменить эквивалентной ей двоичной тетрадой (таблица 1).

Пример10. Число $EF8_{16}$ перевести в двоичную систему счисления

Решение: $EE8_{16}=111011101000_2$

При переходе из восьмеричной системы счисления в шестнадцатеричную и обратно, необходим промежуточный перевод чисел в десятичную или двоичную систему.

Пример11. Число FEA_{16} перевести в восьмеричную систему счисления.

Решение:

I способ:

$FEA_{16}=15\cdot16^2+14\cdot16^1+10\cdot16^0=3840+224+10=4074_{10}$

4074:8=509(2)

509:8=63(5)

63:8=7(7)

$FEA_{16}=4074_{10}=7752_8$

II способ:

$FEA_{16}=1111\ 1110\ 1010_2$ (по таблице1)

$111\ 111\ 101\ 010_2=7752_8$

Пример12. Число 635_8 перевести в шестнадцатеричную систему счисления.

Решение:

I способ:

$635_8=6\cdot8^2+3\cdot8^1+5\cdot8^0=384+24+5=413_{10}$

413:16=25(13)

25:16=1(9)

$635_8=413_{10}=19D_{16}$

II способ:

$635_8=110\ 011101_2$ (по таблице1)

$0001\ 1001\ 1101_2=19D_{16}$

Задания:

Варианты заданий.

Вариант 1.

1) Преобразуйте следующие двоичные числа в десятичные:

1101, 110110, 10111, 10010, 1001.

2) Преобразуйте следующие восьмеричные числа в десятичные:

1, 510, 57, 15, 653.

3) Преобразуйте следующие десятичные числа в двоичные:

256, 13, 72, 11, 101.

4) Преобразуйте следующие десятичные числа в восьмеричные:

39, 7, 162, 41, 418.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:

764, 66, 10, 763, 14.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:

69, 8, 2E9, 3FC, 55.

Вариант 2

1) Преобразуйте следующие двоичные числа в десятичные:
   11100, 1110, 100111, 11110, 100101.
2) Преобразуйте следующие восьмеричные числа в десятичные:
   576, 5, 42, 3, 71.
3) Преобразуйте следующие десятичные числа в двоичные:
   296, 13, 44, 62, 2.
4) Преобразуйте следующие десятичные числа в восьмеричные:
   2, 34, 305, 22, 7.
5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
   912, 11, 75, 15, 64.
6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
   61, 268, D, 23B, 67.

Вариант 3.

1) Преобразуйте следующие двоичные числа в десятичные:
   1000, 111001, 10011, 1101, 101110.
2) Преобразуйте следующие восьмеричные числа в десятичные:
   2, 20, 244, 7, 24.
3) Преобразуйте следующие десятичные числа в двоичные:
   52, 21, 342, 6, 112.
4) Преобразуйте следующие десятичные числа в восьмеричные:
   6, 43, 389, 7, 140.
5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
   8, 112, 798, 643, 11.
6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
   248, 41, E, 45, 23C.

Вариант 4.

1) Преобразуйте следующие двоичные числа в десятичные:
   1111, 110001, 11000, 1000, 11011.
2) Преобразуйте следующие восьмеричные числа в десятичные:
   54, 714, 7, 6, 777.
3) Преобразуйте следующие десятичные числа в двоичные:
   40, 12, 199, 91, 29.
4) Преобразуйте следующие десятичные числа в восьмеричные:

288, 3, 19, 336, 2.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
117, 11, 972, 15, 541.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
75, 360, 8, 7A, C.

Вариант 5.

1) Преобразуйте следующие двоичные числа в десятичные:
1111, 10110, 100000, 111001, 10000.

2) Преобразуйте следующие восьмеричные числа в десятичные:
322, 1, 43, 402, 37.

3) Преобразуйте следующие десятичные числа в двоичные:
16, 73, 255, 19, 90.

4) Преобразуйте следующие десятичные числа в восьмеричные:
2, 30, 241, 62, 3.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
949, 89, 9, 120, 12.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
F, 4F, 348, 219, 6C.

Вариант 6.

1) Преобразуйте следующие двоичные числа в десятичные:
1111, 100111, 10100, 1110, 11111.

2) Преобразуйте следующие восьмеричные числа в десятичные:
4, 676, 11, 70, 7.

3) Преобразуйте следующие десятичные числа в двоичные:
97, 7, 479, 155, 41.

4) Преобразуйте следующие десятичные числа в восьмеричные:
294, 1, 20, 197, 7.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
815, 14, 108, 895, 13.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
374, 72, 9, 5F, 2B6.

Вариант 7.

1) Преобразуйте следующие двоичные числа в десятичные:
1011, 10000, 101101, 1100, 100101.

2) Преобразуйте следующие восьмеричные числа в десятичные:
3, 16, 215, 177, 6.

3) Преобразуйте следующие десятичные числа в двоичные:
172, 94, 26, 80, 339.

4) Преобразуйте следующие десятичные числа в восьмеричные:
484, 9, 3, 2, 110.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
639, 66, 10, 11, 77.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
4F, 347, 8, 6E, C.

Вариант 8.

1) Преобразуйте следующие двоичные числа в десятичные:
110000, 1000, 10110, 11000, 111110.

2) Преобразуйте следующие восьмеричные числа в десятичные:
40, 536, 2, 57, 114.

3) Преобразуйте следующие десятичные числа в двоичные:
121, 63, 22, 78, 154.

4) Преобразуйте следующие десятичные числа в восьмеричные:
7, 10, 75, 6, 336.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
76, 14, 898, 646, 70.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
9, 3FC, 7E, 77, 3FD.

Вариант 9.

1) Преобразуйте следующие двоичные числа в десятичные:
111010, 1000, 11011, 10100, 101010.

2) Преобразуйте следующие восьмеричные числа в десятичные:
7, 552, 45, 314, 6.

3) Преобразуйте следующие десятичные числа в двоичные:
97, 149, 19, 5, 90.

4) Преобразуйте следующие десятичные числа в восьмеричные:
4, 437, 13, 7, 47.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
114, 575, 12, 8, 922.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
B, 67, 222, 234, C.

Вариант 10

1) Преобразуйте следующие двоичные числа в десятичные:
10011, 1001, 100010, 10110, 110000.
2) Преобразуйте следующие восьмеричные числа в десятичные:
732, 3, 17, 7, 641.
3) Преобразуйте следующие десятичные числа в двоичные:
27, 341, 93, 40, 5.
4) Преобразуйте следующие десятичные числа в восьмеричные:
31, 1, 403, 117, 29.
5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
620, 10, 100, 66, 760.
6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
73, 233, C, F, 78.

Вариант 11.

1) Преобразуйте следующие двоичные числа в десятичные:
111100, 1110, 11001, 101100, 1111.
2) Преобразуйте следующие восьмеричные числа в десятичные:
7, 21, 521, 730, 67.
3) Преобразуйте следующие десятичные числа в двоичные:
8, 55, 215, 88, 7.
4) Преобразуйте следующие десятичные числа в восьмеричные:
211, 5, 24, 6, 16.
5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
874, 13, 68, 9, 725.
6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
5A, 8, 244, A, 34F.

Вариант 12.

1) Преобразуйте следующие двоичные числа в десятичные:
111000, 1000, 10111, 101101, 10101.
2) Преобразуйте следующие восьмеричные числа в десятичные:
7, 175, 51, 3, 767.
3) Преобразуйте следующие десятичные числа в двоичные:

5, 463, 99, 76, 110.

4) Преобразуйте следующие десятичные числа в восьмеричные:
18, 306, 6, 43, 3.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
111, 584, 9, 671, 117.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
257, 7A, 9, 2FB, 73.

Вариант 13.

1) Преобразуйте следующие двоичные числа в десятичные:
11100, 101100, 1101, 100100, 10011.

2) Преобразуйте следующие восьмеричные числа в десятичные:
230, 16, 5, 607, 27.

3) Преобразуйте следующие десятичные числа в двоичные:
7, 71, 278, 172, 32.

4) Преобразуйте следующие десятичные числа в восьмеричные:
57, 1, 216, 226, 34.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
794, 126, 11, 591, 15.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
D, 350, 4C, F, 389.

Вариант 14.

1) Преобразуйте следующие двоичные числа в десятичные:
101101, 10010, 1000, 110001, 11001.

2) Преобразуйте следующие восьмеричные числа в десятичные:
660, 4, 40, 552, 2.

3) Преобразуйте следующие десятичные числа в двоичные:
128, 4, 45, 5, 297.

4) Преобразуйте следующие десятичные числа в восьмеричные:
104, 7, 61, 2, 223.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
117, 14, 728, 10, 563.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
77, 227, C, F, 204.

Вариант 15.

1) Преобразуйте следующие двоичные числа в десятичные:
   110110, 10010, 1101, 1000, 11010.
2) Преобразуйте следующие восьмеричные числа в десятичные:
   4, 17, 637, 1, 663.
3) Преобразуйте следующие десятичные числа в двоичные:
   32, 373, 64, 67, 154.
4) Преобразуйте следующие десятичные числа в восьмеричные:
   40, 80, 1, 61, 3.
5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
   12, 733, 81, 603, 10.
6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
   7F, 343, 9, 4E, 8.

Вариант 16.

1) Преобразуйте следующие двоичные числа в десятичные:
   1111, 10110, 101100, 1101, 101111.
2) Преобразуйте следующие восьмеричные числа в десятичные:
   577, 14, 2, 631, 11.
3) Преобразуйте следующие десятичные числа в двоичные:
   301, 18, 97, 63, 16.
4) Преобразуйте следующие десятичные числа в восьмеричные:
   4, 444, 22, 294, 7.
5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
   98, 14, 972, 12, 83.
6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
   E, 2EC, 70, 43, 20D.

Вариант 17.

1) Преобразуйте следующие двоичные числа в десятичные:
   1001, 10001, 100110, 10110, 1011.
2) Преобразуйте следующие восьмеричные числа в десятичные:
   20, 445, 6, 5, 607.
3) Преобразуйте следующие десятичные числа в двоичные:
   468, 25, 53, 167, 18.
4) Преобразуйте следующие десятичные числа в восьмеричные:
   196, 1, 40, 486, 5.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
13, 642, 86, 902, 8.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
72, 8, 35B, B, 355.

Вариант 18.

1) Преобразуйте следующие двоичные числа в десятичные:
1110, 111011, 11101, 1100, 100000.

2) Преобразуйте следующие восьмеричные числа в десятичные:
6, 71, 216, 53, 463.

3) Преобразуйте следующие десятичные числа в двоичные:
8, 392, 76, 25, 41.

4) Преобразуйте следующие десятичные числа в восьмеричные:
37, 7, 433, 301, 4.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
12, 991, 124, 11, 67.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
265, 6B, E, 71, C.

Вариант 19.

1) Преобразуйте следующие двоичные числа в десятичные:
10111, 1010, 111010, 1110, 110111.

2) Преобразуйте следующие восьмеричные числа в десятичные:
1, 151, 43, 355, 16.

3) Преобразуйте следующие десятичные числа в двоичные:
13, 91, 466, 75, 4.

4) Преобразуйте следующие десятичные числа в восьмеричные:
5, 497, 14, 45, 128.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:
925, 8, 68, 109, 854.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:
313, 7A, F, 337, A.

Вариант 20.

1) Преобразуйте следующие двоичные числа в десятичные:
100010, 11110, 1011, 1101, 10110.

2) Преобразуйте следующие восьмеричные числа в десятичные:

4, 204, 20, 1, 17.

3) Преобразуйте следующие десятичные числа в двоичные:

66, 207, 3, 90, 279.

4) Преобразуйте следующие десятичные числа в восьмеричные:

50, 1, 114, 277, 13.

5) Преобразуйте следующие десятичные числа в шестнадцатеричные:

11, 528, 126, 89, 9.

6) Преобразуйте следующие шестнадцатеричные числа в десятичные:

A, 29F, 59, 268, 8.

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Вариант задания
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Что называют системой счисления?
2. Какая система счисления используется в компьютерной технике?
3. Почему человеку удобно пользоваться десятичной системой счисления?

Практическое занятие №6Арифметические операции над числами, записанными в двоичной, восьмеричной и шестнадцатеричной системе счисления

Цель занятия:

1. Ознакомиться с арифметические операции над числами, записанными в двоичной, восьмеричной и шестнадцатеричной системе счисления

Исходные данные:

Папка на РС «Практическое занятие № 6»

Содержание и порядок выполнения задания:

Изучите теоретическую часть

Выполните задания.

Теоретическая часть:

Над числами, записанными в любой системе счисления, можно производить различные арифметические операции. Многоразрядные числа складываются, вычитаются, умножаются и делятся по тем же правилам, что и в десятичной системе счисления.

Так, для сложения двоичных чисел необходимо использовать следующие правила:

0 + 0 = 0; 1 + 0 = 1;

0 + 1 = 1; 1 + 1 = 10.

В последнем случае, при сложении двух единиц, происходит переполнение младшего разряда, и единица переносится в старший разряд. Переполнение возникает в случае, если сумма равна основанию системы счисления (в данном случае это число 2) или больше его (для двоичной системы счисления это не актуально).

Пример сложения двоичных чисел:

```
  10110      1001       1111
+   101    + 1010     +    1
 -------    -------    -------
  11011      10011      10000
```

Вычитание одноразрядных двоичных чисел выполняется по следующим правилам:

0 - 0 = 0; 0 - 1 = 1 (заем из старшего разряда);

1 - 0 = 1; 1 - 1 = 0.

*Пример*вычитания двоичных чисел:

```
                (0)
  1011        10110
-  111       -01100
 -----       -------
   100        01010
```

Умножение одноразрядных двоичных чисел выполняется по следующим правилам:

0 · 0 = 0; 1 · 0 = 0;
0 · 1 = 0; 1 · 1 = 1.

Пример умножения двоичных чисел:

```
    1011        10101
   x 101       x  111
 -------     -------
+   1011    +  10101
  1011       10101
 --------   10101
  110111    -----------
            10010011
```

программными средствами 1С: Колледж
сии по оригиналу, хранящемуся в 1С: Колледж

Деление в любой позиционной системе счисления производится по тем же правилам, как и деление углом в десятичной системе. В двоичной системе деление выполняется особенно просто, ведь очередная цифра частного может быть только нулем или единицей.

Выполняя умножение многозначных чисел в различных позиционных системах счисления, можно использовать обычный алгоритм перемножения чисел в столбик, но при этом результаты перемножения и сложения однозначных чисел необходимо заимствовать из соответствующих таблиц умножения и сложения.

Таблица сложения в восьмеричной системе счисления:

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

Таблица умножения в восьмеричной системе счисления:

*	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

Таблица сложения в 16-ричной системе счисления:

+	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

Таблица умножения в 16-ричной системе счисления:

×	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	0	2	4	6	8	A	C	E	10	12	14	16	18	1A	1C	1E
3	0	3	6	9	C	F	12	15	18	1B	1E	21	24	27	2A	2D
4	0	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	0	5	A	F	14	19	1E	23	28	2D	32	37	3C	41	46	4B
6	0	6	C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A
7	0	7	E	15	1C	23	2A	31	38	3F	46	4D	54	5B	62	69
8	0	8	10	18	20	28	30	38	40	48	50	58	60	68	70	78
9	0	9	12	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87
A	0	A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96
B	0	B	16	21	2C	37	42	4D	58	63	6E	79	84	8F	9A	A5
C	0	C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4
D	0	D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A9	B6	C3
E	0	E	1C	2A	38	46	54	62	70	7E	8C	9A	A8	B6	C4	D2
F	0	F	1E	2D	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1

При выполнении действий сложения и вычитания в 8-ной системе счисления необходимо помнить (если не пользоваться таблицей):

- в записи результатов сложения и вычитания могут быть использованы только цифры восьмеричного алфавита;

- основание восьмеричной системы счисления равен 8, т.е. переполнение наступает, когда результат сложения больше или равен 8. В этом случае для записи результата надо вычесть 8, записать остаток, а к старшему разряду прибавить единицу переполнения;

- если при вычитании приходится занимать единицу в старшем разряде, эта единица переносится в младший разряд в виде 8 единиц.

Пример сложения и вычитания восьмеричных чисел:

$$\begin{array}{r} {\scriptstyle 1} \\ 17 \\ +\ \ 6 \\ \hline 25 \end{array}$$

7+6=13=8+5

1+1=2

1) $17_8+6_8=25_8$

кумент управляется программными средствами 1С: Колледж
Проверь актуальность версии по оригиналу, хранящемуся в 1С: Колледж

```
 6 3 5 4
-  7 0 5
 -------
 5 4 4 7 ——— 8+4-5=7
   |  |    ——— 5-1=4
   |  |    ——— 8+3-7=4
   |  ——— 6-1=5
```

2) $6354_8 - 705_8 = 5447_8$

Пример умножения восьмеричных чисел:

```
   17
 × 14
 ----
   74
+ 17
 ----
  264
```

$7 \cdot 4 = 28 = 8 \cdot 3 + \underline{\mathbf{4}}$

$4 \cdot 1 + 3 = \underline{\mathbf{7}}$

$17_8 \cdot 14_8 = 264_8$

✓При выполнении действий сложения и вычитания в 16-ной системе счисления необходимо помнить:

- в записи результатов сложения и вычитания могут быть использованы только цифры шестнадцатеричного алфавита (0-9, A-F)

- основание шестнадцатеричной системы счисления равно 16, т.е. переполнение наступает, когда результат сложения больше или равен 16. В этом случае для записи результата надо вычесть 16, записать остаток, а к старшему разряду прибавить единицу переполнения;

- если при вычитании приходится занимать единицу в старшем разряде, эта единица переносится в младший разряд в виде 16 единиц.

Пример сложения и вычитания 16-ных чисел:

1)

```
 1  1
    9  A
 +  B  7
 -------
 1  5  1
```

A+7 =10+7=17=17-16 =1→1

9+B +1=10+11=21=21- 16=5→1

$9A_{16} + B7_{16} = 151_{16}$

2)

$$\begin{array}{r} BC4_{16} \\ -\ AF_{16} \\ \hline B15_{16} \end{array}$$

Из 4 нельзя вычесть F, значит, из левого разряда мы займем 16. Теперь F надо вычитать из 20. В результате - 5, записываем его под разрядом единиц. Цифра С уменьшилась на 1,теперь это В. Значит надо А вычесть из В, это будет 1.

Записываем этот результат в разряде десятков. Из сотен в этот раз мы ничего не занимали и в вычитаемом только 2 цифры — сотен нет, то есть сносим В из уменьшаемого в результат.

Итак: $BC4_{16}$ - AF_{16} = $B15_{16}$.

Пример умножения 16-ных чисел:

$$\begin{array}{r} \cdot\ \ 7 \\ 5\ \ D \\ \hline +\ \ 5\ \ B \\ 2\ \ 3 \\ \hline 2\ \ 8\ \ B \end{array}$$

7·D = 7· 13 = 91 = 16· 5 +11 =16· **5**+ **B**
7 ·5 = 35 =16 ·**2** +**3**

✓Операция деления выполняется по алгоритму, подобному алгоритму выполнения операции деления в десятичной системе счисления. Следует только грамотно пользоваться теми цифрами, которые входят в алфавит используемой системы счисления.

Наиболее просто деление столбиком осуществить в двоичной системе счисления, так как в двоичной системе счисления придется лишь сравнить два числа и вычитать из большего меньшее.

$$\begin{array}{r|l} -1000011_2 & 101_2 \\ \cline{2-2} 101_2 & 111_2 \\ \cline{1-1} -111_2 & \\ 101_2 & \\ \cline{1-1} -101_2 & \\ 101_2 & \\ \cline{1-1} 0 & \end{array}$$

При делении мы не всегда получим целое число. Разделим число 5 на число 2, предварительно переведя их в двоичную систему счисления:

$$\begin{array}{r|l} -101_2 & 10_2 \\ \cline{2-2} 10_2 & 10,1_2 \\ \cline{1-1} -010_2 & \\ 10_2 & \\ \cline{1-1} 0_2 & \end{array}$$

Деление в восьмеричной системе счисления:

$$\begin{array}{r|l} -330 & 22 \\ \underline{22} & \overline{14_8} \\ -110 & \\ \underline{110} & \\ 0 & \end{array}$$

Процесс деления в шестнадцатеричной системе счисления аналогичен.

✓При выполнении любых арифметических операций над числами, представленными в разных системах счисления, следует предварительно перевести их в одну и ту же систему.

Пример 1. Выполнить арифметические операции с числами в различных системах счисления:

$101110_2 + 1263_8 \rightarrow x_{10}$

$101110_2 = 1\cdot2^5 + 0\cdot2^4 + 1\cdot2^3 + 1\cdot2^2 + 1\cdot2^1 + 0\cdot2^0 = 32 + 0 + 8 + 4 + 2 + 0 = 46_{10}$

$1263_8 = 1\cdot8^3 + 2\cdot8^2 + 6\cdot8^1 + 3\cdot8^0 = 512 + 128 + 48 + 3 = 691_{10}$

$46_{10} + 691_{10} = 737_{10}$

Задание

Выполнить арифметические операции в 2-й СС:

1) $1110_2 + 1001_2 = 10111_2$

2) $1110_2 – 1001_2 = 101_2$

3) $1110_2 * 1001_2 = 1111110_2$

4) $1110_2 / 11_2 = 100_2$

Выполнить арифметические операции в 8-й СС:

5) $67_8 + 23_8 = 112_8$

6) $67_8 – 23_8 = 44_8$

7) $67_8 * 23_8 = 2025_8$

8) $74_8 / 24_8 = 3_8$

Выполнить арифметические операции в 16-й СС:

9) $AF_{16} + 97_{16} = 146_{16}$

10) $AF_{16} – 97_{16} = 18_{16}$

11) $AF_{16} * 97_{16} = 6739_{16}$

12) $5A_{16} / 1E_{16} = 3_{16}$

Сложить числа $5E_{16}$ и 12_8. Сумму представить в десятичной системе счисления.

13) ($94_{10} + 10_{10} = 104_{10}$)

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Вариант задания
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Какие правила применяются для выполнения арифметических операций в позиционных системах счисления?

2. Что необходимо помнить при выполнении операций в различных позиционных системах счисления?

3. Как нужно поступить, если операции производятся над числами, представленными в различных позиционных системах счисления?

Практическое занятие №7 Измерение информации. Алфавитный и вероятностный подход к измерению информации

Цель занятия: изучить алфавитный и вероятностный подход к измерению информации

*Исходные данные:*теоретический материал

Содержание и порядок выполнения задания:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

Различные подходы к измерению количества информации в сообщении определяются различием подходов к определению самого понятия «информация».

Чтобы измерить что-либо, необходимо ввести единицу измерения. Минимальная единица измерения информации — бит. Смысл данной единицы также различен в рамках разных подходов к измерению информации.

Выделяют три подхода.

1. Неизмеримость информации в быту

Если в сообщении содержалось для вас что-то новое, то оно информативно. Но для другого человека в этом же сообщении нет ничего нового, для него оно не информативно. Это происходит оттого, что до получения данного сообщения знания каждого из нас были различны. Фактор субъективного восприятия сообщения делает невозможным количественную оценку информации в сообщении, т. е. если рассматривать количество полученной информации с точки зрения новизны для получателя, то измерить её невозможно.

2. Вероятностный, или содержательный подход

Попытаться объяснить данный подход можно, допустив,что для каждого человека можно условно выделить (например, в виде окружности) область его знания. Всё, что будет находиться за пределами окружности, можно назвать информационной неопределенностью. Постепенно, в процессе обучения или иной деятельности происходит переход от незнания к знанию, т. е. неопределенность уменьшается. Именно такой подход к информации как мере уменьшения неопределенности знания позволяет ее количественно оценить (измерить).

Сообщение, уменьшающее неопределенность знания в 2 раза, несет один бит информации.

Например: при подбрасывании монеты может выпасть либо «орел», либо «решка». Это два возможных события. Они равновероятны. Сообщение о том, что произошло одно из двух равновероятных событий (например, выпала «решка»), уменьшает неопределенность нашего знания (перед броском монеты) в два раза.

Математики рассматривают идеальный вариант, что возможные события равновероятны. Если даже события неравновероятны, то возможен подсчет вероятности выпадения каждого события.

Под неопределенностью знания здесь понимают количество возможных событий, их может быть больше, чем два.

Например, количество оценок, которые может получить студент на экзамене, равно четырем. Сколько информации содержится в сообщении о том, что он получил «4»? Рассуждая, с опорой на приведенное выше определение, можем

сказать, что если сообщение об одном из двух возможных событий несет 1 бит информации, то выбор одного из четырех возможных событии несет 2 бита информации. Можно прийти к такому выводу, пользуясь методом половинного деления. Сколько вопросов необходимо задать, чтобы выяснить необходимое, столько битов и содержит сообщение. Вопросы должны быть сформулированы так, чтобы на них можно было ответить «да» или «нет», тогда каждый из них будет уменьшать количество возможных событий в 2 раза.

Или:

i = log2N.

Это формула Р. Хартли. Если p = 1/N — вероятность наступления каждого из N равновероятных событий, тогда формула Хартли записывается так:

i = log2(1/p) = log2p

Чтобы пользоваться рассмотренным подходом, необходимо вникать в содержание сообщения. Это не позволяет использовать данный подход для кодирования и передачи информации с помощью технических устройств.

3. Алфавитный подход к измерению информации

Подход основан на подсчете числа символов в сообщении. Этот подход не связывает количество информации с содержанием сообщения, позволяет реализовать передачу, хранение и обработку информации с помощью технических устройств, не теряя при этом содержания (смысла) сообщения.

Алфавит любого языка включает в себя конечный набор символов. Исходя из вероятностного подхода к определению количества информации, появление символов алфавита в тексте можно рассматривать как различные возможные события. Количество таких событий (символов) N называют мощностью алфавита. Тогда количество информации, которое несет каждый из N символов, согласно вероятностному подходу определяется из формулы: $2^i = N$.

Количество символов в тексте из k символов:

I=k ? i

Алфавитный подход является объективным способом измерения информации и используется в технических устройствах.

Переход к более крупным единицам измерения

Ограничения на максимальную мощность алфавита не существует, но есть алфавит, который можно считать достаточным (на современном этапе) для работы с информацией, как для человека, так и для технических устройств. Он включает в

себя: латинский алфавит, алфавит языка страны, числа, спецсимволы — всего около 200 знаков. По приведенной выше таблице можно сделать вывод, что 7 битов информации недостаточно, требуется 8 битов, чтобы закодировать любой символ такого алфавита, 256 = 28. 8 бит образуют 1 байт. То есть для кодирования символа компьютерного алфавита используется 1 байт. Укрупнение единиц измерения информации аналогично применяемому в физике — используют приставки «кило», «мега», «гига». При этом следует помнить, что основание не 10, а 2.

1 Кб (килобайт) = 210 байт = 1024 байт,

1 Мб(мегабайт) = 210 Кб = 220 байт и т. д.

Умение оценивать количество информации в сообщении поможет определить скорость информационного потока по каналам связи. Максимальную скорость передачи информации по каналу связи называют пропускной способностью канала связи. Самым совершенным средством связи на сегодня являются оптические световоды. Информация передается в виде световых импульсов, посылаемых лазерным излучателем. У этих средств связи высокая помехоустойчивость и пропускная способность более 100Мбит/с.

Задания.

1. Переведи 40 бит в байты.
2. Решите задачу: Считая, что один символ кодируется одним байтом, подсчитать в байтах количество информации, содержащееся в фразе: «Век живи, век учись.»
3. На экзамене вы берете экзаменационный билет, и преподаватель сообщает, что сообщение о его номере несет 4 бита информации. Определите количество экзаменационных билетов (запишите полное решение)
4. Песня содержит 32 ноты. Какое количество информации несет одна нота этой песни? (запишите полное решение)
5. Считая, что один символ кодируется одним байтом, подсчитать в байтах количество информации, содержащееся в фразе: «Не делай из мухи слона.»
6. Сообщение о том, что ваш друг живет на десятом этаже несет в себе 4 бита информации. Сколько этажей в доме?
7. Какое количество информации о цвете вынутого шарика будет получено, если в непрозрачном пакете хранятся: 10 белых, 20 красных, 30 синих и 40 зеленых шариков?

8. Найти объем информации, содержащейся в тексте из 3000 символов, написанном русскими буквами.

9. Для записи письма был использован алфавит мощностью в 16 символов. Письмо состояло из 25 строк. В каждой строке вместе с пробелами было 64 символа. Сколько байт информации содержало письмо?

10. Черно-белое изображение имеет 8 градаций яркости. Размер изображения 10*15 см. Разрешение 300 точек на дюйм (1 дюйм = 2,5 см). Сколько Кбайт памяти требуется для хранения изображения в несжатом виде?

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Список используемых источников
4. Выводы и предложения
5. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Есть ли связь между алфавитным подходом к измерению информации и содержанием информации?
2. В чем можно измерить объем письменного или печатного текста?
3. Оцените объем одной страницы данного учебника в байтах.
4. Что такое бит с позиции алфавитного подхода к измерению информации?
5. Какие единицы используются для измерения объема информации на компьютерных носителях?

Практическое занятие №8 Кодирование и декодирование информации. Кодовые таблицы

Цель занятия: Изучить кодирование и декодирование информации

Познакомиться с кодовыми таблицами

*Исходные данные:*теоретический материал

Содержание и порядок выполнения задания:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

Кодирование это перевод информации с одного языка на другой.

Декодирование обратный переход. Один символ исходного сообщения может заменяться одним или несколькими символами нового кода, а может быть и наоборот несколько символов исходного сообщения заменяются одним символом в новом коде.

Кодирование может быть Равномерное при равномерном кодировании все символы кодируются кодами равной длины; Неравномерное: при неравномерном кодировании разные символы могут кодироваться кодами разной длины, и это затрудняет однозначное декодирование или делает его невозможным. Условие Фано означает, что никакое кодовое слово не является началом другого кодового слова. Это обеспечивает возможность однозначной расшифровки закодированных сообщений. Обратное условие Фано также является достаточным условием однозначного декодирования неравномерного кода. В нём требуется, чтобы никакой код не был окончанием другого (более длинного) кода. Для возможности однозначного декодирования достаточно выполнения одного из условий или прямого, или обратного. Однако существуют варианты неравномерного кодирования, для которых оба условия нарушены, и тем не менее они однозначно декодируются. Кодовое дерево (дерево кодирования Хаффмана) это двоичное дерево, у которого: листья помечены символами, для которых разрабатывается кодировка; узлы (в том числе корень) помечены суммой вероятностей появления всех символов, соответствующих листьям поддерева, корнем которого является соответствующий узел. Метод Хаффмана на входе получает таблицу частот встречаемости символов в исходном тексте. Далее на основании этой таблицы строится дерево кодирования Хаффмана. Алгоритм построения дерева Хаффмана:

Шаг 1. Символы входного алфавита образуют список свободных узлов. Каждый лист имеет вес, который может быть равен либо вероятности, либо количеству вхождений символа в сжимаемый текст.

Шаг 2. Выбираются два свободных узла дерева с наименьшими весами.

Шаг 3. Создается их родитель с весом, равным их суммарному весу.

Шаг 4. Родитель добавляется в список свободных узлов, а двое его детей удаляются из этого списка.

Шаг 5. Одной дуге, выходящей из родителя, ставится в соответствие бит, другой бит.

Шаг 6. Повторяем шаги, начиная со второго, до тех пор, пока в списке свободных узлов не останется только один свободный узел. Он и будет считаться корнем дерева.

Существует два подхода к построению кодового дерева: от корня к листьям и от листьев к корню.

Пример №1 с решением

Для передачи по каналу связи сообщения, состоящего только из букв А, Б, В, Г, решили использовать неравномерный по длине код: А=1, Б=01, В=001. Как нужно закодировать букву Г, чтобы длина кода была минимальной, и допускалось однозначное разбиение кодированного сообщения на буквы?

1) 001

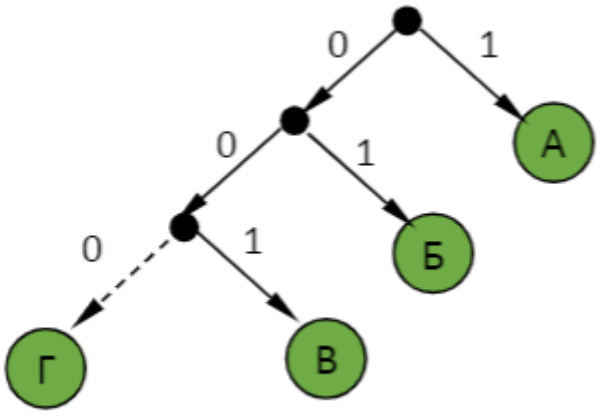
2) 000

3) 11

4) 101

Решение:

Это задание удобнее решать с помощью дерева Хаффмана; условие Фано выполняется тогда, когда все выбранные кодовые слова заканчиваются в листьях дерева. По листьям дерева можно однозначно определить, где может находиться буква Г, чтобы длина кода была минимальной, и допускалось однозначное разбиение кодированного сообщения на буквы.



Штриховой линией отмечена «пустая» ветка, на которой можно «прикрепить» лист для кодового слова буквы Г: 2). 000

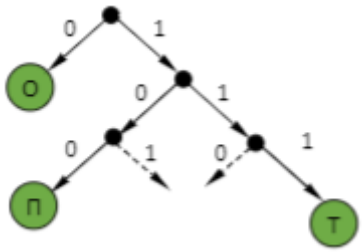
Пример №2 с решением

По каналу связи передаются сообщения, содержащие только 4 буквы П, О, С, Т; для передачи используется двоичный код, допускающий однозначное декодирование. Для букв Т, О, П используются такие кодовые слова: Т - 111, О - 0, П -100.

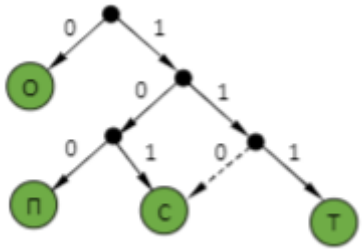
Укажите кратчайшее кодовое слово для буквы С, при котором код будет допускать однозначное декодирование. Если таких кодов несколько, укажите код с наименьшим числовым значением.

Решение:

В дереве кода все кодовые слова должны располагаться в листьях дерева, то есть в узлах, которые не имеют потомков. Построим дерево для заданных кодовых слов О-0, Т - 111 и П - 100:



Штриховыми линиями отмечены две «пустые» ветви, на которые можно «прикрепить» лист для кодового слова буквы С: 101 или 110; из них минимальное значение имеет код 101



Пример 3 с решением

Для кодирования некоторой последовательности, состоящей из букв К, Л, М, Н, решили использовать неравномерный двоичный код, удовлетворяющий условию Фано. Для буквы Л использовали кодовое слово 1, для буквы М - кодовое слово 011. Какова наименьшая возможная суммарная длина всех четырёх кодовых слов?

1) 10

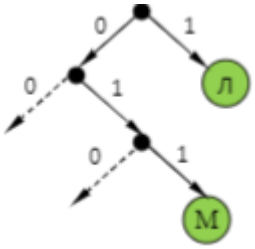
2) 9

3) 8

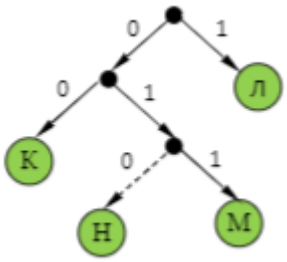
4) 7

Решение:

Построим дерево для заданных кодовых слов Л – 1 и М -011



Штриховыми линиями отмечены две «пустые» ветви, на которые можно «прикрепить» листья для кодовых слов букв К (00) и Н (010)



Таким образом, выбрав кодовые слова Л -1, М -011, К - 00, Н -010, получаем суммарную длину кодовых слов 9 символов

Ответ:2.

Пример 4 с решением

Для кодирования букв Д, Х, Р, О, В решили использовать двоичное представление чисел 0, 1, 2, 3 и 4 соответственно (с сохранением одного незначащего нуля в случае одноразрядного представления). Закодируйте последовательность букв ХОРОВОД таким способом и результат запишите восьмеричным кодом.

Решение:

Сначала следует представить данные в условии числа в двоичном коде:

Д	Х	Р	О	В
0	1	2	3	4
00	01	10	11	100

Затем закодировать последовательность букв: ХОРОВОД - 011110111001100. Теперь разобьём это представление на тройки справа налево и переведём полученный набор чисел в десятичный код, затем в восьмеричный (восьмеричное представление совпадает с десятичным при разбиении тройками)

011 110 111 001 100 – 36714

Пример 5 с решением

Для кодирования букв А, Б, В, Г решили использовать двухразрядные последовательные двоичные числа (от 00 до 11, соответственно). Закодируйте таким образом последовательность символов ББГА и запишите полученное двоичное число в шестнадцатеричной системе счисления.

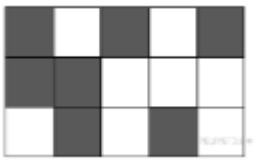
Решение:

Закодируем последовательность букв: ББГА - 01011100. Теперь разобьём это представление на четвёрки справа налево и переведём полученный набор чисел сначала в десятичный код, затем в шестнадцатеричный:

0101 1100 - 5 12 - 5С.

Пример 6 с решением

Черно-белое растровое изображение кодируется построчно, начиная с левого верхнего угла и заканчивая в правом нижнем углу. При кодировании 1 обозначает черный цвет, а 0 - белый. Закодируйте, таким образом, изображение и запишите результат в восьмеричной системе счисления.



Код первой строки: 10101

Код второй строки: 11000

Код третьей строки: 01010.

Запишем коды по порядку в одну строку: 1010111000001010. Теперь разобьём это представление на тройки справа налево и переведём полученный набор чисел в десятичный код (восьмеричное представление совпадает с десятичным при разбиении тройками)

101 011 100 001 010 – 53412

Задания

1) Для кодирования некоторой последовательности, состоящей из букв А, Б, В, Г и Д, решили использовать неравномерный двоичный код, позволяющий однозначно декодировать двоичную последовательность, появляющуюся на приёмной стороне канала связи. Для букв А, Б, В и Г использовали такие кодовые слова: А - 111, Б - 110, В - 100, Г - 101.

Укажите, каким кодовым словом может быть закодирована буква Д. Код должен удовлетворять свойству однозначного декодирования. Если можно использовать более одного кодового слова, укажите кратчайшее из них.

1)0

2) 01

3) 00

4) 000

2) Для кодирования некоторой последовательности, состоящей из букв А, Б, В, Г и Д, используется неравномерный двоичный код, позволяющий однозначно декодировать полученную двоичную последовательность. Вот этот код: А - 00, Б - 01, В - 100, Г - 101, Д - 110. Можно ли сократить для одной из букв длину кодового слова так, чтобы код по-прежнему можно было декодировать однозначно? Коды остальных букв меняться не должны. Выберите правильный вариант ответа.

1) для буквы Д – 11

2) это невозможно

3) для буквы Г – 10

4) для буквы Д – 10

3) По каналу связи передаются сообщения, содержащие только 4 буквы К, О, Р, А; для передачи используется двоичный код, допускающий однозначное декодирование. Для букв Р, А, К используются такие кодовые слова: Р: 000, А: 10, К: 01.

Укажите такое кодовое слово для буквы О, при котором код будет допускать однозначное декодирование. Если таких кодовых слов несколько, укажите то, у которого меньшая длина.

1) 1

2)0

3) 11

4) 001

4) Для кодирования некоторой последовательности, состоящей из букв У, Ч, Е, Н, И и К, используется неравномерный двоичный префиксный код. Вот этот код: У - 000, Ч - 001, Е - 010, Н - 100, И - 011, К - 11. Можно ли сократить для одной из букв длину кодового слова так, чтобы код по-прежнему остался префиксным? Коды остальных букв меняться не должны.

Выберите правильный вариант ответа.

Примечание. Префиксный код это код, в котором ни одно кодовое слово не является началом другого; такие коды позволяют однозначно декодировать полученную двоичную последовательность.

1) кодовое слово для буквы Е можно сократить до 01;

2) кодовое слово для буквы К можно сократить до 1;

3) кодовое слово для буквы Н можно сократить до 10;

4) это невозможно.

5) Для кодирования некоторой последовательности, состоящей из букв А, Б, В, Г и Д, решили использовать неравномерный двоичный код, позволяющий однозначно декодировать двоичную последовательность, появляющуюся на приёмной стороне канала связи. Для букв А, Б, В и Г использовали такие кодовые слова: А - 111, Б - 110, В - 101, Г - 100. Укажите, каким кодовым словом из перечисленных ниже может быть закодирована буква Д. Код должен удовлетворять свойству однозначного декодирования. Если можно использовать более одного кодового слова, укажите кратчайшее из них.

1)1

2) 0

3) 01

4) 10

6) По каналу связи передаются сообщения, содержащие только 4 буквы: Е, Н, О, Т. В любом сообщении больше всего букв О, следующая по частоте буква Е, затем Н. Буква Т встречается реже, чем любая другая. Для передачи сообщений нужно использовать неравномерный двоичный код, допускающий однозначное декодирование; при этом сообщения должны быть как можно короче. Шифровальщик может использовать один из перечисленных ниже кодов. Какой код ему следует выбрать?

1) Е - 0, Н - 1, О - 00, Т – 11

2) О -1, Н -0, Е- 01,Т-10

3) Е - 1, Н -01, О -001, Т -000

4) О -0, Н -11, Е - 101, Т -100

7) По каналу связи передаются сообщения, содержащие только 4 буквы П, О, С, Т; для передачи используется двоичный код, допускающий однозначное декодирование. Для букв Т, О, П используются такие кодовые слова: Т: 111, О: 10, П:

01. Укажите такое кодовое слово для буквы С, при котором код будет допускать однозначное декодирование. Если таких кодовых слов несколько, укажите тот, у которого меньшая длина.

1)1

2) 0

3) 00

4) 110

8) Для кодирования сообщения, состоящего только из букв A, B, C, D и E, используется неравномерный по длине двоичный код:

A	B	C	D	E
000	11	01	001	10

Какое (только одно!) из четырех полученных сообщений было передано без ошибок и может быть раскодировано:

1) 110000010011110

2) 110000011011110

3)110001001001110

4) 110000001011110

Дополнительные задачи

9) Для кодирования букв О, К, Г, Д, Р решили использовать двоичное представление чисел 0, 1, 2, 3 и 4 соответственно (с сохранением одного незначащего нуля в случае одноразрядного представления). Закодируйте последовательность букв ГОРОДОК таким способом и результат запишите восьмеричным кодом.

10) Для передачи по каналу связи сообщения, состоящего только из символов А, Б, В и Г, используется неравномерный (по длине) код: А- 10, Б-11, В-001, Г-011. Через канал связи передается сообщение: АБГВГБ. Закодируйте сообщение данным кодом. Полученное двоичное число переведите в шестнадцатеричный вид

Ответы к заданиям для тренировки

1) 1

2) 1

3) 3

4) 3

5) 2

6) 4

7) 3

8) 1

9) 42061

10) 5B2F.

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Список используемых источников
4. Выводы и предложения
5. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Кодирование как изменение формы представления информации.

2. Как компьютер распознает информацию?

3. Как измерить информацию?

4. Единицы измерения информации.

Практическое занятие № 9Арифметические и логические основы работы компьютера

Цель занятия:

изучение логических элементов компьютера и их таблиц истинности.

Исходные данные: теоретический материал, таблицы истинности

Содержание и порядок выполнения задания:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

Алгебра логики появилась в середине XIX в. в трудах английского математика Джорджа Буля.

Логическое высказывание – это любое повествовательное предложение, в отношении которого можно однозначно сказать, истинного оно или ложно. При этом не всякое предложение является логическим высказыванием. Например, «Хороший студент» не является логическим высказыванием, так как невозможно судить об его истинности или ложности, а высказывание «Иванов – хороший студент» является логическим высказыванием.

Употребляемые слова и словосочетания «*не*», «*и*», «*или*», «*если, то*», «*тогда и только тогда*» позволяют из уже заданных высказываний строить новые. Такие слова и словосочетания называю*тся логическими связками*. При этом высказывания, образованные из других высказываний с помощью логических связок, называются *составными*. Высказывание, не являющееся составным, *называется элементарным*.

Для того чтобы обращаться к логическим высказываниям им назначаются имена. Например, через А обозначим высказывание «Петя был во Франции», а через В высказывание «Петя был в Италии». Тогда составное высказывание примет вид - «Петя был и во Франции, и в Италии», далее можно записать кратко: «А И В». Здесь И – логическая связка, А,В – логические переменные, которые могут принимать только два значения: истинна и ложь, обозначаемые 1 и 0.

В алгебре логики высказывания могут принимать лишь два значения: истинна – 1 и ложь – 0.

Работу логических элементов описывают с помощью таблиц истинности.

Таблица истинности – это табличное представление логической схемы (операции), в котором перечислены все возможные сочетания значений истинности входных сигналов (операндов) вместе со значением истинности выходного сигнала (результата операции) для каждого из этих сочетаний.

1. Операции с логическими высказываниями

С логическими высказываниями можно производить следующие операции:

1. Операция отрицания.

Операция, выражаемая словом НЕ, называется отрицанием, или инверсией, и обозначается чертой над высказыванием или знаком ⌉.

Таблица 1. Таблица истинности логического отрицания

A	$\overline{A}$

0	1
1	0

2. Операция конъюнкции.

Операция, выражаемая связкой И, называется соединением, или конъюнкцией, или логическим умножением, и обозначается знаком «&» (может обозначаться знаком «^» или «·». Высказывание F=A&B истинно только тогда, когда оба высказывания А и В истинны.

Таблица 2. - Таблица истинности логического умножения

A	B	F=A&B
0	0	0
0	1	0
1	0	0
1	1	1

3. Операция дизъюнкции.

Операция, выражаемая связкой ИЛИ, называется разделением, или дизъюнкцией, или логическим сложением, и обозначается знаком «v» (или «+»). ВысказываниеF=AvВложно тогда и только тогда, когда оба высказывания А и В ложны.

Таблица3. -Таблица истинности логического сложения

A	B	F=AvB
0	0	0
0	1	1
1	0	1
1	1	1

4. Операция импликации.

Операция следования, выражаемая связками «если…, то», «из … следует», «… влечет … », называется импликацией и обозначается знаком «→». ВысказываниеA→B ложно тогда и только тогда, когда А истинно, а В ложно.

Таблица 4 - Таблица истинности логической функции импликации

A	B	F=A→B
0	0	1
0	1	1
1	0	0

1	1	1

5. Операция эквиваленции.

Операция равенства, выражаемая связками «тогда и только тогда», «необходимо и достаточно», «… равносильно …», называется эквиваленцией, или двойной импликацией, и обозначается знаками «↔» или «~». Высказывание A↔B истинно тогда и только тогда, когда значения A и B совпадают.

Таблица 5 - Таблица истинности эквиваленции

A	B	F=A↔B
0	0	1
0	1	0
1	0	0
1	1	1

Порядок выполнения логических операций задается круглыми скобками, но для уменьшения числа скобок договорились считать, что сначала выполняется операция отрицание (НЕ), затем конъюнкции (И), затем дизъюнкции (ИЛИ) и в последнюю очередь - импликации. Это называется приоритетом операций.

Порядок выполнения логических операций в сложном логическом выражении:

1. Инверсия;
2. Конъюнкция;
3. Дизъюнкция;
4. Импликация;
5. Эквивалентность.

Для изменения указанного порядка выполнения логических операций используются скобки.

2. Основные законы алгебры логики

В алгебре логики выполняются следующие основные законы, позволяющие проводить тождественные преобразования (упрощения) логических выражений, показанные в таблице 2.1. Законы алгебры логики

Закон	для «ИЛИ»	для «И»
1. Двойного отрицания	$A=\overline{\overline{A}}$	
2. Переместительный	$A \vee B=B \vee A$	$A\&B=B\&A$
3. Сочетательный	$(A \vee B) \vee C=A \vee (B \vee C)$	$(A\&B)\&C=A\&(B\&C)$
4. Распределительный	$(A \vee B)\&C=(A\&C) \vee (B\&C)$	$(A \& B) \vee C=(A \vee C) \& (B \vee C)$
5. Законы де Моргана	$\overline{A \vee B}=\overline{A}\&\overline{B}$	$\overline{A\&B}=\overline{A} \vee \overline{B}$
6. Идемпотентности	$A \vee A=A$	$A\&A=A$
7. Исключения констант	$A \vee 1=1;\ A \vee 0=A$	$A\&1=A;\ A\&0=0$
8. Противоречия	—	$A\&\overline{A}=0$
9. Исключение третьего	$A \vee \overline{A}=1$	—
10. Поглощения	$A \vee (A\&B)=A$	$A\&(A \vee B)=A$
11. Исключения	$(A\&B) \vee (\overline{A}\&B)=B$	$(A \vee B)\&(\overline{A} \vee B)=B$
12. Контрапозиции	$(A \leftrightarrow B)=(B \leftrightarrow A)$	

Задания

№ 1. Какие из предложений являются высказываниями? Определите их истинность. Определите тип высказывания: общее, частное или единичное.

1. Все солдаты храбрые
2. Некоторые ученики двоечники
3. Все ананасы приятны на вкус
4. Некоторые мои друзья собирают марки
5. Все лекарства неприятны на вкус
6. А — первая буква в алфавите
7. Некоторые медведи — бурые
8. Тигр — хищное животное
9. У некоторых змей нет ядовитых зубов
10. Все металлы проводят тепло

A	**B**	¬A инверсия	AVB дизъюнкция	A&B конъюнкция	A→B импликация	A↔B эквиваленция
0	0	1	0	0	1	1
0	1	1	1	0	1	0
1	0	0	1	0	0	0
1	1	0	1	1	1	1

Пример. Какое из приведенных имен удовлетворяет логическому условию:

¬ (последняя буква гласная → первая буква согласная) & вторая буква согласная

1) ИРИНА 2) АРТЕМ 3) СТЕПАН 4) МАРИЯ

Имя	X1: последняя буква гласная	X2: первая буква согласная	X3: вторая буква согласная	X1→X2	¬(X1→X2)	¬(X1→X2)&X3
Ирина	1	0	1	0	1	1
Артём	0	0	1	1	0	0
Степан	0	1	1	1	0	0
Мария	1	1	0	1	0	0

№ 2. а) Какое из приведенных названий животных удовлетворяет логическому условию

¬(есть мягкий знак & (вторая буква гласная → пятая буква согласная))

1) МЕДВЕДЬ 2) ВЫХУХОЛЬ 3) МУРАВЬЕД 4) ОБЕЗЬЯНА

б) Какое из приведенных имен удовлетворяет логическому условию

¬(первая буква гласная → последняя буква гласная) & вторая буква согласная

1) ИРИНА 2) ОЛЕГ 3) СТЕПАН 4) ИЛОНА

в) Какое из приведенных имен удовлетворяет логическому условию:

¬ (первая буква согласная → вторая буква согласная) & (предпоследняя буква гласная → последняя буква гласная)

1) КРИСТИНА 2) МАКСИМ 3) СТЕПАН 4) МАРИЯ

Выводы и предложения о проделанной работе:

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Список используемых источников
4. Выводы и предложения
5. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Что такое логическое высказывание?
2. Перечислите основные бинарные логические операции и связки.
3. Опишите операцию инверсии – НЕ.
4. Опишите операцию конъюнкции – И.
5. Опишите операцию дизъюнкции-ИЛИ.

Практическое занятие №10 Составление таблиц истинности по логическим выражениям

Цель занятия:

построение таблиц истинности логических высказываний.

применять основные формулы и правила алгебры логики

*Исходные данные:*таблицы истинности

Содержание и порядок выполнения задания:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

Решение логических выражений принято записывать в виде **таблиц истинности**– таблиц, в которых по действиям показано, какие значения принимает логическое выражение при всех возможных наборах его переменных.

При составлении таблицы истинности длялогического выражения необходимо учитывать**порядок выполнения логических операций**,а именно:

1. действия в скобках,
2. инверсия (**отрицание**),
3. & (**конъюнкция**),
4. v (**дизъюнкция**),
5. => (**импликация**),
6. <=> (**эквивалентность**).

Алгоритм составления таблицы истинности:

1. Выяснить количество строк в таблице (вычисляется как 2^n, где n – количество переменных + строка заголовков столбцов).

2. Выяснить количество столбцов (вычисляется как количество переменных + количество логических операций).

3. Установить последовательность выполнения логических операций.

4. Построить таблицу, указывая названия столбцов и возможные наборы значений исходных логических переменных.

5. Заполнить таблицу истинности по столбцам.

6. Записать ответ.

Примеры.

1. **Составим таблицу истинности** для формулы **«А или В или не В»,** которая содержит две переменные А и В. В первых двух столбцах таблицы запишем четыре возможных пары значений этих переменных, в последующих столбцах — значения промежуточных формул и в последнем столбце — значение формулы.

Формулу перепишем в терминах алгебры логики: $A \vee B \vee \overline{B}$

В результате получим таблицу:

Переменные		Промежуточные логические формулы		Формула
A	B	$\overline{B}$	B∨A	A∨B∨$\overline{B}$
0	0	1	0	1
1	0	1	1	1
0	1	0	1	1
1	1	0	1	1

Т.е. формула A∨B∨$\overline{B}$является **тождественно истинной**.

Пример 2

Построим таблицу истинности для выражения **F=(AvB)&(¬Av¬B)**.

1. Количество строк=2^2 (2 переменных+строка заголовков столбцов)=5.

2. Количество столбцов=2 логические переменные (A, B)+ 5 логических операций (v,&,¬,v,¬) = 7.

3. Расставим порядок выполнения операций: 1 5 2 4 3

(A v B)&(¬A v ¬B)

4-5. Построим таблицу и заполним ее по столбцам:

A	B	AvB	¬A	¬B	¬Av¬B	(AvB)&(¬Av¬B)
0	**0**	0	1	1	1	**0**
0	1	1	1	0	1	1
1	0	1	0	1	1	1
1	**1**	1	0	0	0	**0**

6. Ответ: F=0, при A=B=0 и A=B=1

Пример 3

Построим таблицу истинности для логического выражения **F=XvY&¬Z**.

1. Количество строк=2^3+1=(3 переменных+строка заголовков столбцов)=9.

2. Количество столбцов=3 логические переменные+3 логических операций = 6.

3. Укажем порядок действий: 3 2 1

X v Y & ¬Z

4-5. Построим таблицу и заполним ее по столбцам:

X	Y	Z	¬Z	Y&¬Z	XvY&¬Z
0	**0**	**0**	1	0	**0**
0	**0**	**1**	0	0	**0**
0	1	0	1	1	1
1	0	0	1	0	1
1	0	1	0	0	1
1	1	0	1	1	1

1	1	1	0	0	1

6. Ответ: F=0, при X=Y=Z=0; при X=Y=0 и Z=1.

Задание:

Построить таблицу истинности функции F

1	$F = A \vee \overline{B} \vee (\overline{A} \vee C)$	16	$F = A \leftrightarrow C \vee B \rightarrow A$
2	$F = A \rightarrow \overline{B} \vee C$	17	$F = A \leftrightarrow \overline{C} \vee B \rightarrow \overline{A}$
3	$F = B \vee (\overline{A} \leftrightarrow C)$	18	$F = (A \leftrightarrow C) \vee (B \rightarrow A)$
4	$F = \overline{B} \vee (A \leftrightarrow C)$	19	$F = A \leftrightarrow C \vee (B \rightarrow \overline{A})$
5	$F = A \wedge B \rightarrow \overline{B} \wedge C$	20	$F = A \leftrightarrow (C \vee B \rightarrow A)$
6	$F = A \wedge B \leftrightarrow \overline{B} \vee C$	21	$F = (\overline{A} \leftrightarrow C) \vee B \rightarrow A$
7	$F = (A \vee \overline{B}) \vee (\overline{A} \rightarrow C)$	22	$F = \overline{A} \leftrightarrow (C \vee \overline{B} \rightarrow A)$
8	$F = (A \rightarrow \overline{B}) \vee C$	23	$F = A \wedge (B \rightarrow \overline{C}) \wedge C$
9	$F = B \vee C \leftrightarrow \overline{A} \vee \overline{C}$	24	$F = A \wedge (B \leftrightarrow \overline{A}) \vee C$
10	$F = \overline{B} \vee (A \wedge C \rightarrow B)$	25	$F = (C \vee \overline{B}) \vee (\overline{A} \vee C)$
11	$F = A \vee B \rightarrow \overline{B} \vee C$	26	$F = A \rightarrow \overline{B} \vee (C \rightarrow B)$
12	$F = A \wedge B \leftrightarrow \overline{B} \vee C$	27	$F = (A \wedge B \rightarrow \overline{B}) \wedge (C \vee \overline{A})$
13	$F = A \rightarrow \overline{B} \vee (\overline{A} \vee C)$	28	$F = \overline{B} \vee (A \leftrightarrow C) \wedge C$
14	$F = \overline{A} \wedge B \rightarrow \overline{B} \vee C$	29	$F = A \wedge B \rightarrow \overline{B} \wedge C$
15	$F = B \vee (\overline{A} \leftrightarrow C) \wedge A$	30	$F = A \wedge B \leftrightarrow \overline{B} \vee C$

Выводы и предложения о проделанной работе:

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Вариант задания
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1 Какие бывают логические выражения?

2 Что такое алгебра логики?

3 Понятие и обозначение инверсии.

4 Таблицы истинности инверсии

5 Понятие и обозначение конъюнкции.

6 Таблицы истинности конъюнкции.

7 Понятие и обозначение дизъюнкции.

8 Таблицы истинности дизъюнкции.

9 Способ изменения порядка действий в логических выражениях.

Практическое занятие № 11 Построение логических схем

*Цель занятия:*Научиться строить логические схемы

*Исходные данные:*теоретический материал

Содержание и порядок выполнения задания:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

Логические основы устройства компьютера

Логический элемент компьютера – это часть электронной логической схемы, которая реализует элементарную логическую функцию.

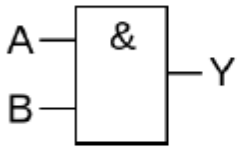
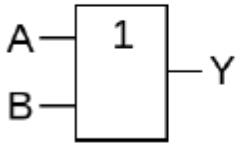
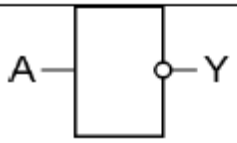
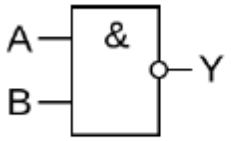
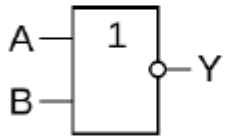
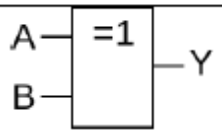
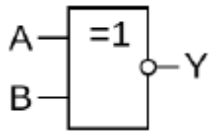
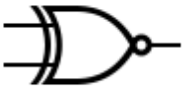
Логическими элементами компьютеров являются электронные схемы И, ИЛИ, НЕ, И-НЕ, ИЛИ-НЕ и др. (называемые также вентилями), а также триггер.

С помощью этих схем можно реализовать любую логическую функцию, описывающую работу устройств компьютера. Обычно у элементов бывает от 2 до 8 входов и один или два выхода. Чтобы представить два логических состояния 1 и 0, соответствующие им входные и выходные сигналы имеют один из двух установленных уровней напряжения, например 5 и 0 В. Высокий уровень обычно соответствует значению «истинна» (1), а низкий – значению «ложь» (0).

Каждый логический элемент имеет свое условное обозначение, которое выражает его логическую функцию, но не указывает на то, какая электронная схема в нем реализована. Это упрощает запись и понимание сложных схем.

Работу логических элементов описывают с помощью таблиц истинности. Основные структурные схемы логических элементов компьютера и их таблицы истинности, представлены в таблице 3.1.

Таблица 3.1. - Структурные схемы логических элементов компьютера

Условное обозначение	Структурная схема (отечественное обозначение)	Структурная схема (зарубежное обозначение)	Таблица истинности
И			A B Y (A&B) 0 0 0 0 1 0 1 0 0 1 1 1
ИЛИ			A B Y (A∨B) 0 0 0 0 1 1 1 0 1 1 1 1
НЕ			A Y ($\bar{A}$) 0 1 1 0
И-НЕ			A B Y 0 0 1 0 1 1 1 0 1 1 1 0
ИЛИ-НЕ			A B Y 0 0 1 0 1 0 1 0 0 1 1 0
Исключающее ИЛИ			A B Y 0 0 0 0 1 1 1 0 1 1 1 0
			
Исключающее ИЛИ-НЕ			A B Y 0 0 1 0 1 0 1 0 0 1 1 1

Почему необходимо уметь строить логические схемы?

Дело в том, что из вентилей составляют более сложные схемы, которые позволяют выполнить арифметические операции и хранить информацию. Причем схему, выполняющую определенные функции, можно построить из различных по сочетанию и количеству вентилей. Поэтому значение формального представления логической схемы чрезвычайно велико. Оно необходимо для того, чтобы разработчик имел возможность выбрать наиболее подходящий ему вариант построения схемы из вентилей. Процесс разработки общей логической схемы устройства (в том числе и компьютера в целом) таким образом становится иерархическим, причем на каждом следующем уровне в качестве «кирпичиков» используются логические схемы, созданные на предыдущем этапе.

Алгебра логики дала в руки конструкторам мощное средство разработки, анализа и совершенствования логических схем. В самом деле, гораздо проще, быстрее и дешевле изучать свойства и доказывать правильность работы схемы с помощью выражающей ее формулы, чем создавать реальное техническое устройство. Именно в этом состоит смысл любого математического моделирования.

Логические схемы необходимо строить из минимально возможного количества элементов, что в свою очередь, обеспечивает большую скорость работы и увеличивает надежность устройства.

Алгоритм построения логических схем:

1) Определить число логических переменных.

2) Определить количество базовых логических операций и их порядок.

3) Изобразить для каждой логической операции соответствующий ей вентиль.

4) Соединить вентили в порядке выполнения логических операций.

Пример 1

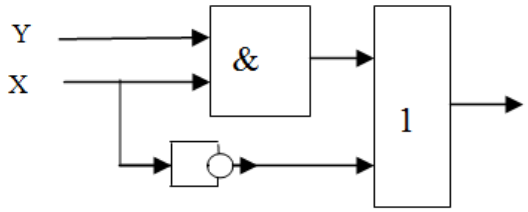
Составить логическую схему для логического выражения: F=¬XvY&X.

1) Две переменные – X и Y.

2) Две логические операции: 1 3 2

¬XvY&X.

3) Строим схему, соединяя вентили в порядке выполнения логических операций:



Пример 2

Постройте логическую схему, соответствующую логическому выражению F=X&Yv¬(YvX).

Вычислить значения выражения для X=1, Y=0.

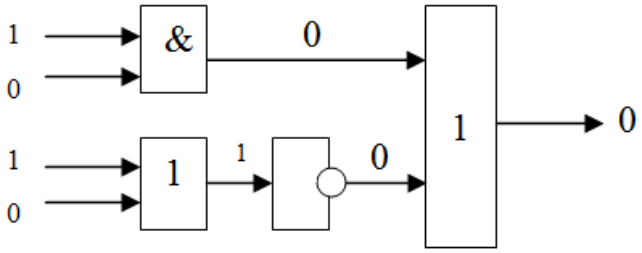
1) Переменных две: X и Y.

2) Логических операций четыре: конъюнкция, две дизъюнкции и отрицание. Определяем порядок выполнения операций:

1 4 3 2

X&Yv ¬(YvX).

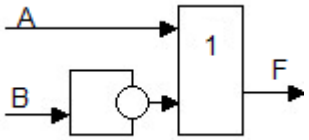
3) Схему строим слева направо в соответствии с порядком выполнения логических операций:



4) Вычислим значение выражения: F=1&0v¬(0v1)=0.

Выполним обратное задание. Дана логическая схема. Необходимо построитьпо ней логическое выражение.

Пример - Задание 1.

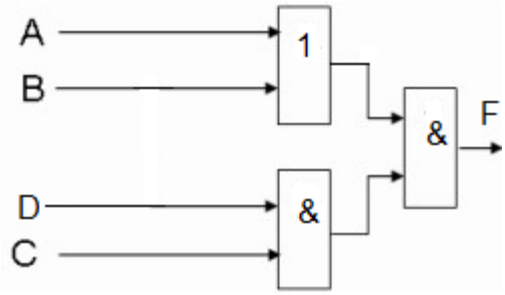


Логических переменных на схеме две (А иВ), вентилей два: один инвертор, один дизъюнктор. Строим логическуюсхему в порядке выполнения логических операций.

F = ¬ B V A

Пример - Задание 2.

Постройте логическую схему, соответствующую логическому выражению и найдите значение логического выражения.



F = (A V B) & (D& C)

Задание 1

Постройте логическую схему, соответствующую логическому выражению, и найдите значение логического выражения:

1) $F = (A\&B)V(B\&C)V(\overline{A}\&C)$

2) $F = \overline{(A\&B)V(BVC)}\&(\overline{A}\&C)$

3) $F = \overline{(A\&B)V(B\&C)}V(\overline{A\&C})$

4) $F = \overline{(A\&B)V(B\&C)}V(\overline{A}\&C)$

5) $F = (\overline{A\&B})V(B\&C)V(\overline{A}\&C)$

6) $F = \overline{(A\&\overline{B})V(BVC)}\&(\overline{A}\&C)$

7) $F = \overline{(A\&B)V(B\&C)}V(A\&C)$

8) $F = \overline{(A\&B)V(B\&C)}V(\overline{A}\&\overline{C})$

9) $F = (A\&B)V(\overline{B\&C})V(\overline{A}\&C)$

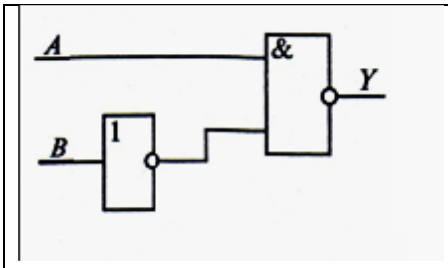
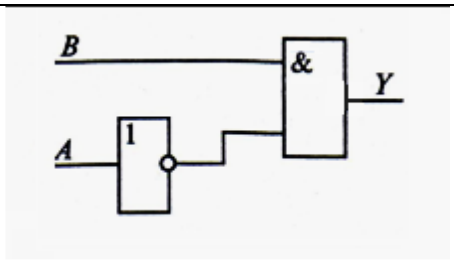
10) $F = \overline{(A\&B)V(BV\overline{C})}\&(\overline{A}\&C)$

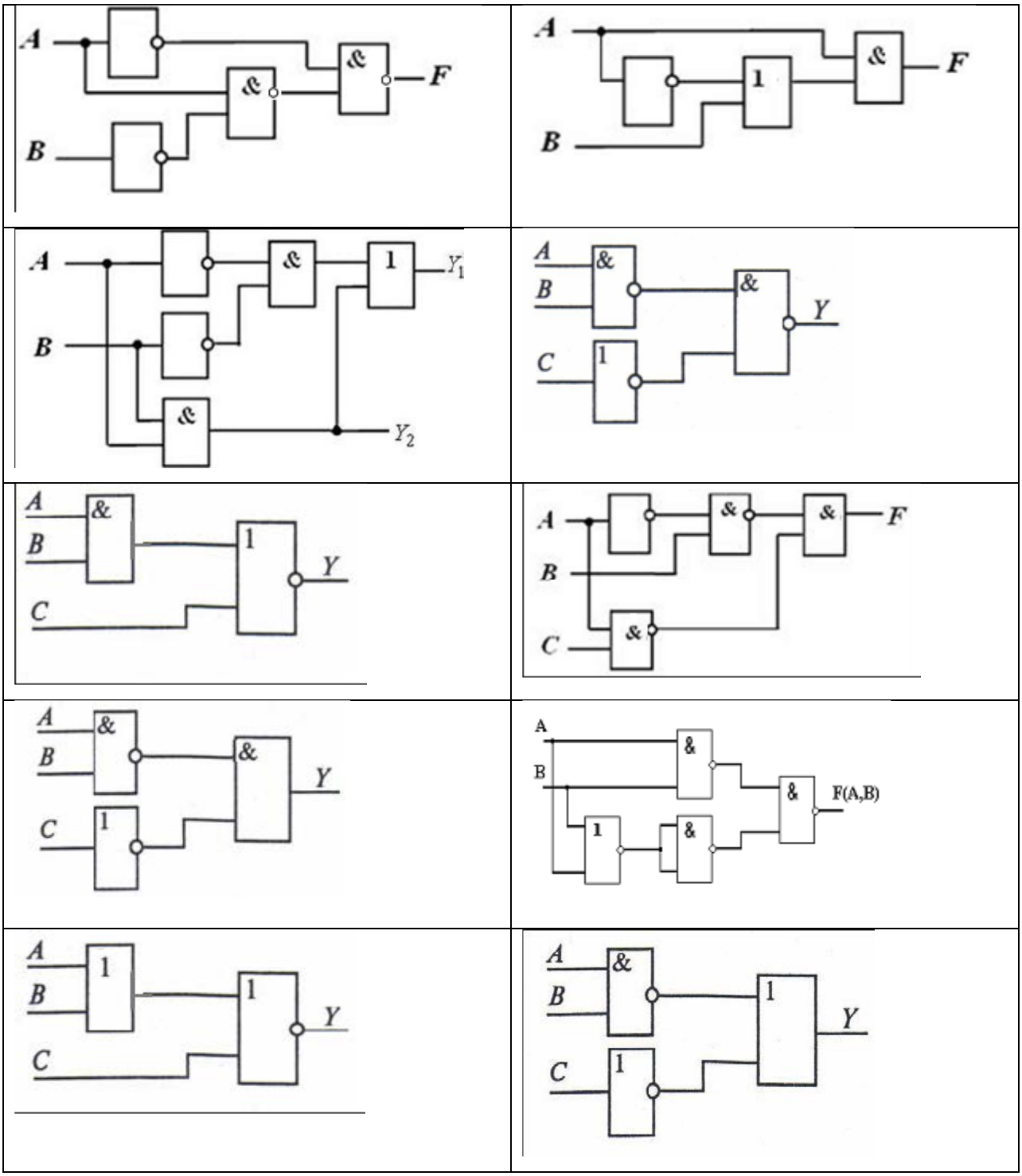
11) $F = (A\&B)V(B\&C)V(\overline{A\&C})$

12) $F = \overline{(\overline{A}\&B)V(B\&C)}V(\overline{A}\&C)$

Задание 2

По логической схеме составить логическую функцию



Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Вариант задания
4. Список используемых источников
5. Выводы и предложения

6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Перечислите основные логические операции.
2. Что такое логическое умножение?
3. Что такое логическое сложение?
4. Что такое инверсия?
5. Что такое таблица истинности?
6. Что такое сумматор?
7. Что такое полусумматор?

Практическое занятие № 12 Графы. Введение и понятия

Цель занятия: познакомить обучающихся с основными понятиями теории графов.

*Исходные данные:*теоретический материал

Содержание и порядок выполнения:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

Понятие графа

Граф – это множество точек или вершин и множество линий или ребер, соединяющих между собой все или часть этих точек.

Вершины, прилегающие к одному и тому же ребру, называются смежными.

Два ребра, у которых есть общая вершина, также называютсясмежными(или соседними).

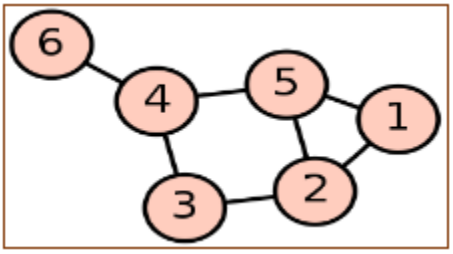


Рис. 1. Граф с шестью вершинами и семью ребрами

Элементы графа

Петля – это дуга, начальная и конечная вершина которой совпадают.

Пустым (нулевым) называется граф без ребер.

Полным называется граф, в котором каждые две вершины смежные.

Нулевой граф – граф, состоящий из «изолированных» вершин.

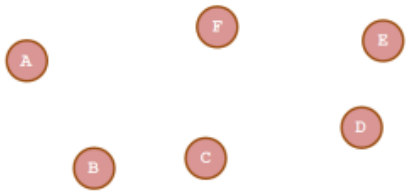


Рис. 2. Нулевой граф

Неполный граф – графы, в которых не построены все возможные ребра.

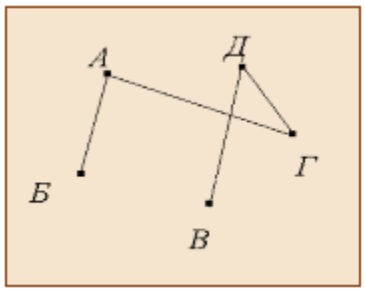


Рис. 3 Неполный граф

Степень графа

Количество рёбер, выходящих из вершины графа, называется степенью вершины. Вершина графа, имеющая нечётную степень, называется нечетной, а чётную степень – чётной.

Если степени всех вершин графа равны, то граф называется однородным. Таким образом, любой полный граф — однородный.

Кружки называются вершинами графа, линии со стрелками - дугами, без стрелок - ребрами. Граф, в котором направление линий не выделяется (все линии являются ребрами), называется неориентированным (рис. 1, А); граф, в котором направление линий принципиально (линии являются дугами) называется ориентированным (рис. 1, Б).

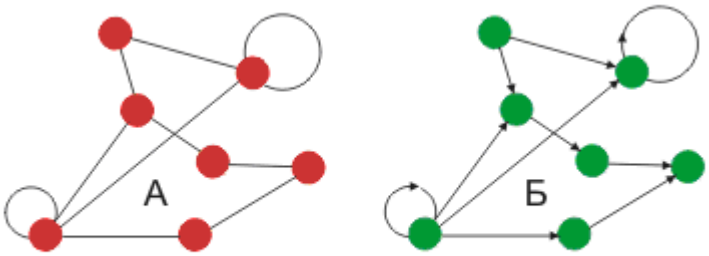


рис.1

Ориентированный граф

Граф называется ориентированным (или орграфом), если некоторые ребра имеют направление. Это означает, что в орграфе некоторая вершина может быть

соединена с другой вершиной, а обратного соединения нет. Если ребра ориентированы, что обычно показывают стрелками, то они называются дугами.

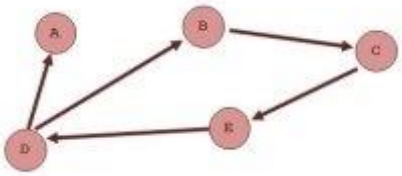


Рис. 4. Ориентированный граф

Ориентированный и неориентированный графы

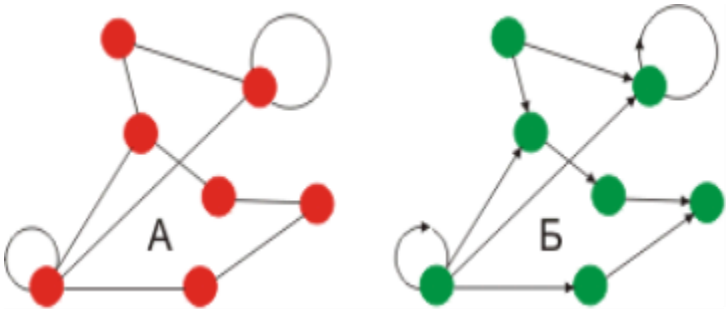


Рис. 5. Примеры неориентированного и ориентированного графов (А и Б)

<u>Задание 1.</u> Построить граф по заданному условию:

В соревнованиях по футболу участвуют 6 команд. Каждую из команд обозначили буквами A, B, C, D, E и F. Через несколько недель некоторые из команд уже сыграли друг с другом:

A с C, D, F;

B с C, E, F;

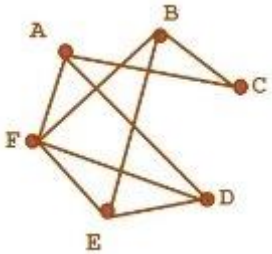
C с A, B;

D с A, E, F;

E с B, D, F;

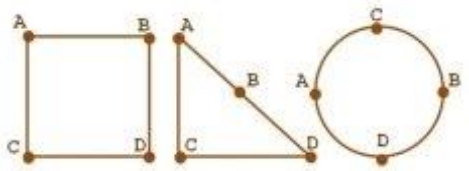
F с A, B, D.

Ответ:



<u>Задание 2.</u>

Определить изображают ли фигуры на рисунке один и тот же граф или нет.



1. 2) 3)

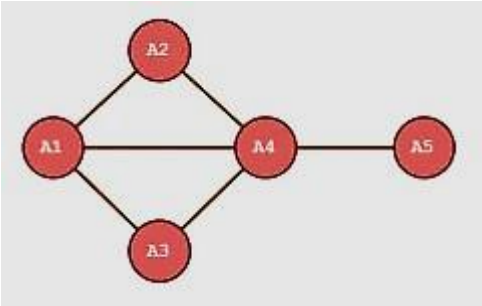
Ответ: Рисунок 1 и рисунок 2 являются изображениями одного графа. Рисунок 3 изображением другого графа.

Путь в графе

Путём в графе называется такая последовательность ребер, в которой каждые два соседних ребра имеют общую вершину и никакое ребро не встречается более одного раза.

Задание 3. Определить какая из перечисленных последовательностей путём не является.

1. (A1 A4); (A4 A5).
2. (A1 A2); (A2 A4); (A4 A5).
3. (A1 A4); (A4 A2); (A2 A1); (A1 A4); (A4, A5).
4. (A1 A4); (A4 A2); (A2 A1); (A1 A3); (A3 A4); (A4, A5).



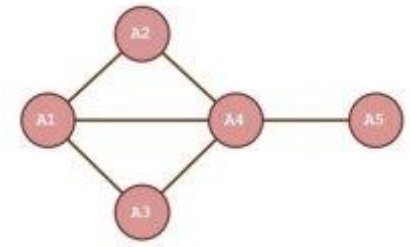
Ответ: Третья последовательность (A1 A4); (A4 A2); (A2 A1); (A1 A4); (A4, A5)

Путь называется простым, если он не проходит ни через одну из вершин графа более одного раза.

Задание 4. Определите, какие последовательности ребер являются путями, и какие из них простые. Если последовательность не является путем укажите почему.

1. (A1 A4); (A4 A5).
2. (A1 A2); (A2 A4); (A4 A5).
3. (A1 A4); (A4 A2); (A2 A1); (A1 A4); (A4, A5).

4. (A1 A4); (A4 A2); (A2 A1); (A1 A3); (A3 A4); (A4, A5).



Ответ: Первая, вторая и четвертая последовательности являются путями, а третья нет, т.к. ребро (A1, A4) повторяется.

Первая и вторая последовательность являются простыми путями, а четвертая нет, т.к. вершины A1 и A4 повторяются.

Понятие цикла в графе

Циклом называется путь, в котором совпадают его начальная и конечная вершины.

Простым циклом в графе называется цикл, не проходящий ни через одну из вершин графа более одного раза.

Задание 5. Назовите в графе циклы, содержащие

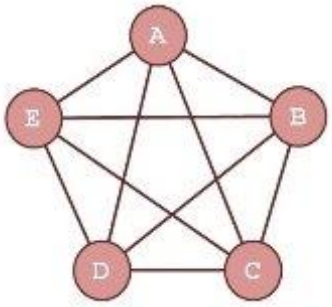
a) 4 ребра;

b) 6 ребер;

c) 5 ребер;

d) 10 ребер.

Какие из этих циклов являются простыми?



Ответ:

Решение:

a. (AB, BC, CE, EA), (CD, DA, AB, BC), (EB, BC, CD, DE) и т.д. – простые циклы.

b. (DB, BE, EA, AB, BC, CD), (EC, CA, AB, BC, CD, DE) ит.д. – циклы.

c. (AB, BC, CD, DE, EA), (AC, CE, EB, BD, DA) ит.д. – простыециклы.

(AC, CE, EB, BD, DA, AB, BC, CD, DE, EA), (EB, BD, DA, AC, CE, EA, AB, BC, CD, DE) и т.д. – циклы.

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Список используемых источников
4. Выводы и предложения
5. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

Изобразите графически:

1. Неориентированное и ориентированное ребро;
2. Неориентированный граф G(V,E), заданный множеством

V={v0, v1, v2, v3, v4, v5}

E = {e1=(v0, v1), e2=(v0, v2}, e3=(v1, v2), e4=(v1,v4), e5=(v2, v5), e6=(v3, v4)}

3. Плоский граф;
4. Полный неориентированный граф на трех, четырех и пяти вершинах;
5. Неполный ориентированный граф на пяти вершинах;
6. Петлю графа.
7. Цикл графа.

Практическое занятие №13 Графы. Способы задания графов

Цель работы: задание графа, вычисление степеней вершин.

Исходные данные: теоретический материал

Содержание и порядок выполнения задания:

Изучите теоретический материал

Выполните задания

Теоретическая часть

1 Граф G - совокупность двух множеств: вершин V и ребер E, между которыми определено отношение инцидентности. Если |V(G)|=n, |E(G)|=m, то граф G есть (n,m) граф, где n - порядок графа, m - размер графа.

2 Каждое ребро e из E инцидентно ровно двум вершинам v', v'', которые оно соединяет. При этом вершина v' и ребро e называются инцидентными друг другу, а вершины v' и v'' называются смежными.

3 Ребро (v',v'') может быть ориентированным и иметь начало (v') и конец (v'') (дуга в орграфе).

4 Ребро (v,v) называется петлей (концевые вершины совпадают).

5 Граф, содержащий ориентированные ребра (дуги), называется орграфом.

6 Граф, не содержащий ориентированные ребра (дуги), называется неографом.

7 Ребра, инцидентные одной паре вершин, называются параллельными или кратными.

8 Конечный граф - число вершин и ребер конечно.

9 Пустой граф - множество ребер пусто (число вершин может быть произвольным).

10 Полный граф - граф без петель и кратных ребер, каждая пара вершин соединена ребром.

11 Локальная степень вершины - число ребер ей инцидентных.

12Внеографе сумма степеней всех вершин равна удвоенному числу ребер (лемма о рукопожатиях). Петля дает вклад, равный 2 в степень вершины.

13В орграфе сумма входящих ребер всех вершин равна сумме исходящих ребер всех вершин и равна числу ребер графа.

14 Графы равны, если множества вершин и инцидентных им ребер совпадают.

15 Графы, отличающиеся только нумерацией вершин и ребер, называются изоморфными.

16 Способы задания графов: − явное задание графа как алгебраической системы; − геометрический; − матрица смежности; − матрица инцидентности

17 Матрица инцидентности: По вертикали указываются вершины, по горизонтали - ребра. aij=1 если вершина i инцидентна ребру j, в противном случае aij=0. Если ребро - петля, то aij=2. Матрицей инцидентности (инциденций) ориентированного графа называется матрица, для которой aij=1, если вершина

является началом дуги ,aij= – 1, если является концом дуги, в остальных случаях aij=0.

18 Матрица смежности - квадратная симметричная матрица. По горизонтали и вертикали - все вершины.aij= число ребер, соединяющее вершины i,j. Матрицей смежности ориентированного графа называется матрица, для которой aij=1, если вершина является началом дуги, в остальных случаях aij=0.

19 Маршрут - последовательность ребер, в которых каждые два соседних ребра имеют общую вершину.

20 Маршрут, в котором начало и конец совпадают - циклический.

21 Маршрут в неографе, в котором все ребра разные - цепь.

22 Маршрут в орграфе, в котором все дуги разные - путь.

23 Вершины связанные, если существует маршрут из одной вершины в другую.

24 Связанный граф - если все его вершины связаны.

25 Плоский граф - граф с вершинами, расположенными на плоскости и непересекающимися ребрами.

26 Вершины графа, которые не принадлежат ни одному ребру, называются изолированными.

27 Дерево - связный граф без циклов.

Пример выполнения:

Исходные данные:

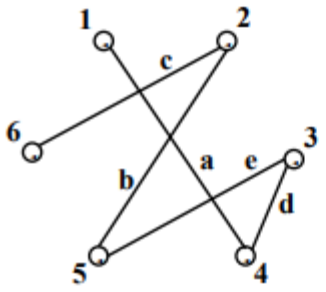
1 Задать неограф, представленный множеством вершин и ребер, графически и матрицами, преобразовать граф в плоский, вычислить степени его вершин.

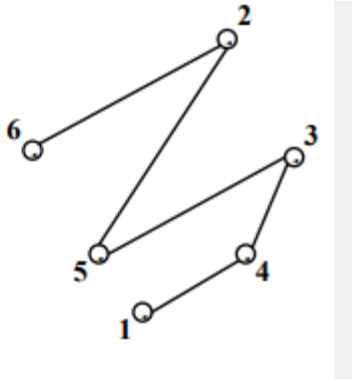
V = {1; 2; 3; 4; 5; 6};E = {a; b; c; d; e}

E = {(1; 4); (2; 5); (2; 6); (3; 4); (3; 5)}

Решение:

1Изобразим граф, соединив вершины: Ребро a соединяет вершины 1 и 4, b соединяет вершины 2 и 5 и т. д. Затем преобразуем этот граф в плоский:





2 Составим матрицу смежности. В первом столбце и первой строке выпишем вершины. Ребру а инцидентны вершины 1 и 4, следовательно в колонке 1 и строке 4 ставим 1, а также колонке 4 и строке 1 ставим 1. Ребру b инцидентны вершины 2 и 5, следовательно в колонке 2 в строке 5 и колонке 5 строке 2 ставим 1 и т.д. Остальные ячейки таблицы содержат нули.

3 Составим матрицу инцидентности. В первом столбце выпишем вершины, первой строке – ребра. Ребру а инцидентны вершины 1 и 4, следовательно в колонке а в строке 1 и строке 4 ставим 1. Ребру b инцидентны вершины 2 и 5, следовательно в колонке b в строке 2 и строке 5 ставим 1и т.д. Остальные ячейки таблицы заполняем нулями

Матрица смежности

	1	2	3	4	5	6
1	0	0	0	1	0	0
2	0	0	0	0	1	1
3	0	0	0	1	1	0
4	1	0	1	0	0	0
5	0	1	1	0	0	0
6	0	1	0	0	0	0

Матрица инцидентности

	a	b	c	d	e
1	1	0	0	0	0
2	0	1	1	0	0
3	0	0	0	1	1
4	1	0	0	1	0
5	0	1	0	0	1
6	0	0	1	0	0

4 Вычислим степени вершин:

ρ (1) = 1ρ (2) = 2ρ (3) = 2ρ (4) = 2ρ (5) = 2ρ (6) = 1

ρ (1) + ρ (2) + ρ (3) + ρ (4) + ρ (5) + ρ (6) = 10 = 2 * q

q = 5 (ребер 5)

Исходные данные:

2Задать граф, представленный матрицей инцидентности, алгебраически, графически и матрицей смежности, преобразовать граф в плоский, вычислить степени его вершин

	a	b	c	d	e	f
1	-1	-1	0	0	0	0
2	1	0	-1	1	0	0
3	0	0	0	-1	0	0
4	0	0	1	0	1	0
5	0	0	0	0	-1	-1
6	0	1	0	0	0	1

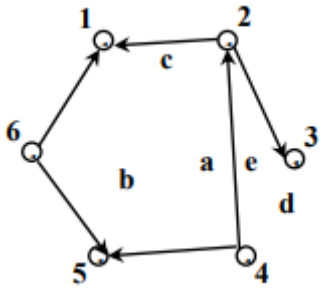
Решение:

1 Количество вершин – 6. V = {1; 2; 3; 4; 5; 6}.

2 Ребро а выходит из вершины 2, т.к. в ячейке (2; 1) стоит 1, а приходит в вершину 1 (в ячейке (1; 1) находится -1) и т.д.

Получим множество E = {(2; 1); (6; 1); (4; 2); (2; 3); (4; 5); (6; 5)}

3Изобразим граф, соединив вершины, этот граф уже плоский, т.к. ребра не пересекаются:



4 Составим матрицу смежности.

	1	2	3	4	5	6
1	0	0	0	0	0	0
2	1	0	1	0	0	0
3	0	0	0	0	0	0
4	0	1	0	0	1	0
5	0	0	0	0	0	0
6	1	0	0	0	1	0

5 Вычислим степени вершин:

ρ1 (1) = 0 ρ2 (1) = 2

ρ1 (2) = 2 ρ2 (2) = 1

ρ1 (3) = 0 ρ2 (3) = 1

ρ1 (4) = 2 ρ2 (4) = 0

ρ1 (5) = 0 ρ2 (5) = 2

ρ1 (6) = 2 ρ2 (6) = 0

ρ1 (1) + ρ1 (2) + ρ1 (3) + ρ1 (4) + ρ1 (5) + ρ1 (6) = 6

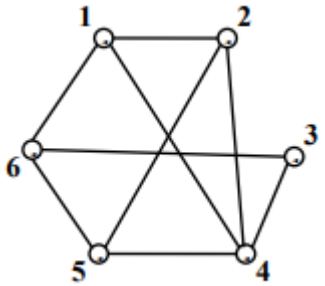
ρ2 (1) + ρ2 (2) + ρ2 (3) + ρ2 (4) + ρ2 (5) + ρ2 (6) = 6

q = 6(ребер 6)

Задания

1. Орграф V = {1; 2; 3; 4; 5; 6}

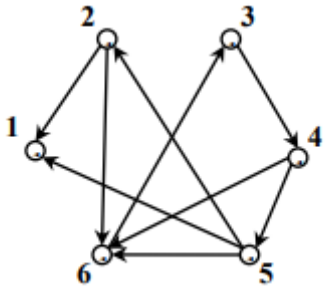
E = {(1; 3); (1; 6); (2; 5); (3; 2); (3; 4); (4; 1); (4; 5); (5; 3); (6; 2)}



2.

3 НеографV = {1; 2; 3; 4; 5; 6}

E = {(1; 2); (1; 5); (2; 3); (3; 1); (3; 4); (4; 2); (4; 5); (4; 6); (5; 3)}



4.

	a	b	c	d	e	f	g	h	i
1	1	1	0	0	0	0	0	0	0
2	1	0	1	1	0	0	0	1	0
3	0	0	0	0	1	1	0	0	0
4	0	1	0	0	0	0	0	1	1
5	0	0	0	1	1	0	1	0	0
6	0	0	1	0	0	1	1	0	1

5.

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Список используемых источников
4. Выводы и предложения
5. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Что такое граф?

2. Что такое инцидентное ребро или инцидентная вершина?

3 Что такое петля?

4 Какое ребро называется ориентированным?

5 Что такое кратные ребра?

6 Что такое неограф?

7 Что такое орграф?

Практическое занятие № 14Решение логических задач с помощью графов

Цель занятия:

- обобщить и систематизировать знания о графах, их видах и свойствах;
- рассмотреть типы задач, которые можно решить с использованием графов;
- отработать навыки преобразования ориентированного графа в дерево;
- сформировать навыки построение путей в графе, поиска кратчайшего пути

Содержание и порядок выполнения задания:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

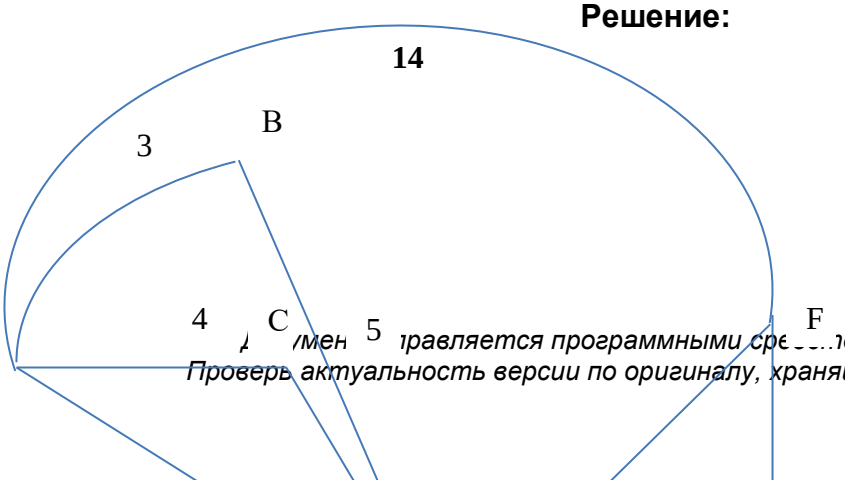
При решении логических задач по информатике часто приходится сталкиваться с заданиями, которые вызывает затруднение при решении. Если же построить граф, то задача существенно упрощается. Теория графов позволяет решать наиболее легким способом, наглядно многие логические задачи, которые способствуют развитию мышления и интеллекта. Слово «граф» означает картинку, где нарисовано несколько точек, некоторые из которых соединены линиями. Графами являются блок – схемы программ для ЭВМ, сетевые графики строительства, где вершины – события, означающие окончания работ на некотором участке, а ребра, связывающие эти вершины, - работы, которые возможно начать по совершении одного события и необходимо выполнить для совершения следующего.

Задача 1. Между населёнными пунктами A, B, C, D, E, F построены дороги, протяжённость которых приведена в таблице. Отсутствие числа в таблице означает, что прямой дороги между пунктами нет.

	A	B	C	D	E	F
A		3	4	7		14
B	3			5		
C	4			2		
D	7	5	2		5	8
E				5		1
F	14			8	1	

Определите длину кратчайшего пути между пунктами A и F при условии, что передвигаться можно только по указанным в таблице дорогам.

Решение:



Документ формирует управляется программными средствами 1С: Колледж
Проверь актуальность версии по оригиналу, хранящемуся в 1С: Колледж

A

8 1

2

7 E

5

D

AF=14

AF=AB+BD+DF=3+5+8=16

AF=AC+CD+DF=4+2+8=14

AF=AD+DE+EF=7+5+1=13

Ответ: 13.

Задача 2

Сколькими способами можно рассадить в ряд на три стула трёх студентов? Выписать все возможные случаи.

Решение этой задачи удобнее всего представить в виде дерева. За его корневую вершину возьмём произвольную точку плоскости О.

На первый стул можно посадить любого из трёх студентов — обозначим их А, В и С. На схеме это соответствует трём ветвям, исходящим из точки О (рис. 1).

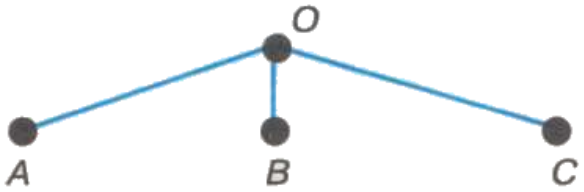


Рис. 1

Посадив на первый стул студента А, на второй стул можно посадить студента В или С. Если же на первый стул сядет В, то на второй можно посадить А или С. Если на первый стул сядет С, то на второй можно будет посадить А или В. Это соответствует на схеме двум ветвям, исходящим из каждой вершины первого уровня (рис. 2).

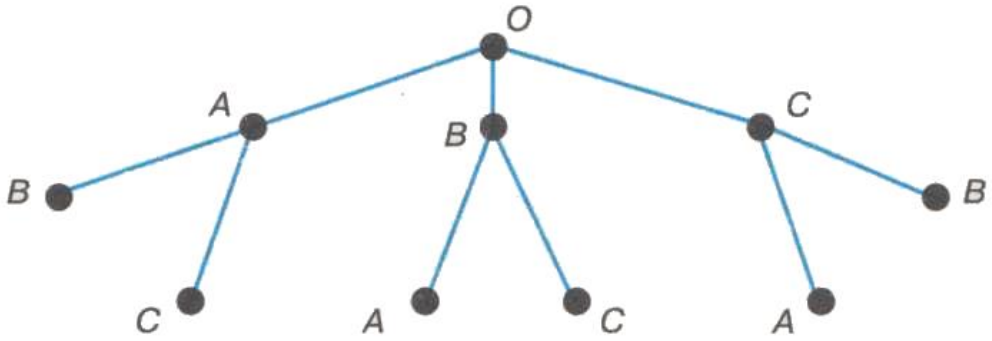


Рис. 2

Очевидно, что третий стул в каждом случае займёт оставшийся студент. Это соответствует одной ветви дерева, которая «вырастает» на каждой из предыдущих ветвей (рис. 3).

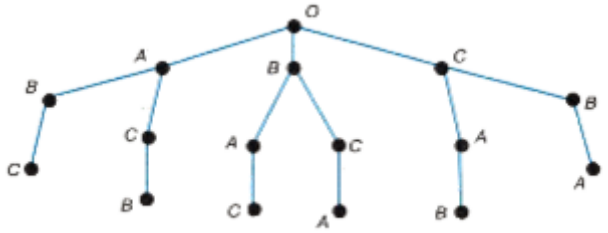


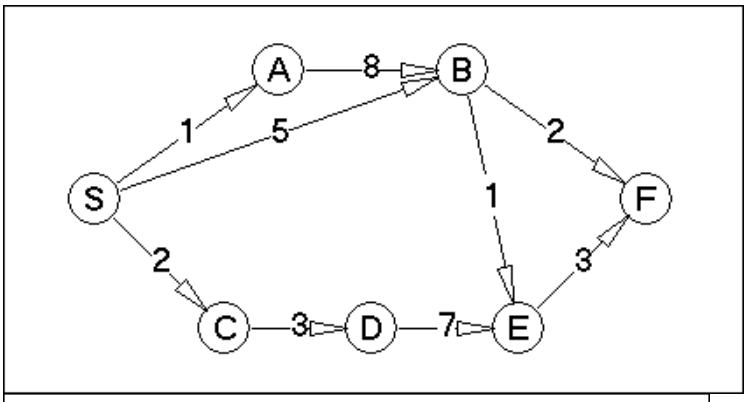
Рис. 3

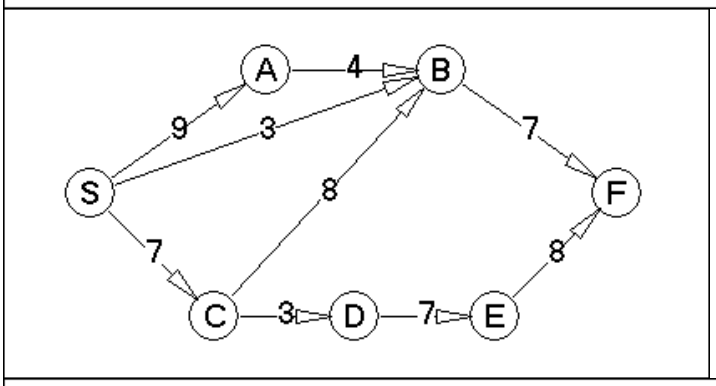
Выпишем все пути от вершин первого уровня к вершинам третьего уровня: А-В-Су А-С-В, В-А-С, В-С-А, С-А-Б, С-В-А. Каждый из выписанных путей определяет один из вариантов рассаживания учеников на стулья. Так как других путей нет, то искомое число способов — 6.

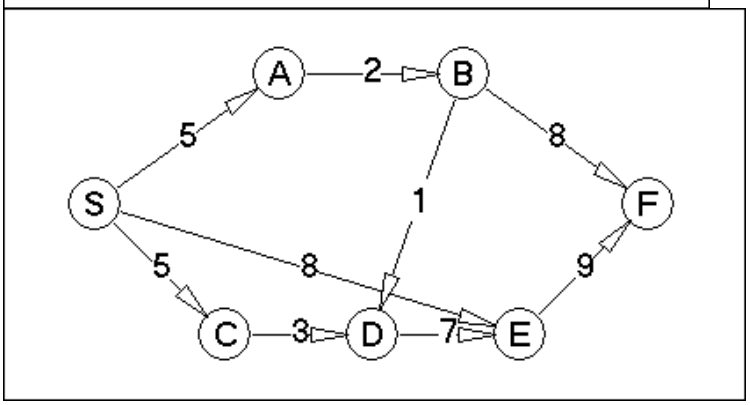
Дерево можно не строить, если не требуется выписывать все возможные варианты, а нужно просто указать их число. В этом случае рассуждать нужно так: на первый стул можно усадить одного из трёх человек, на второй — одного из двух оставшихся, на третий — одного оставшегося: 3-2-1 = 6.

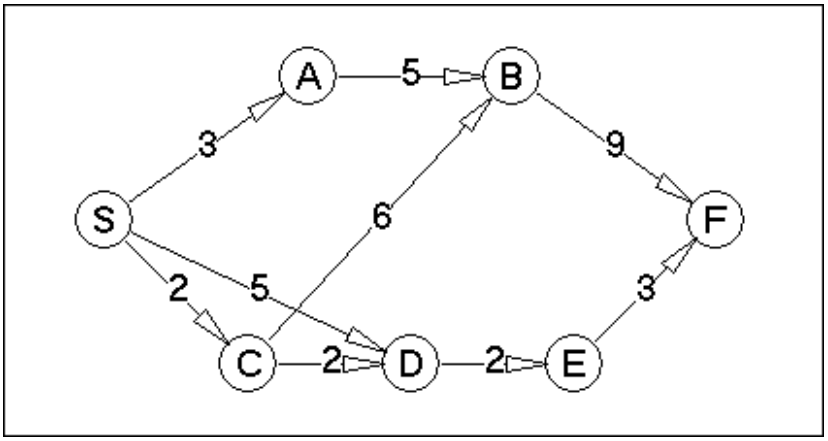
Задание1

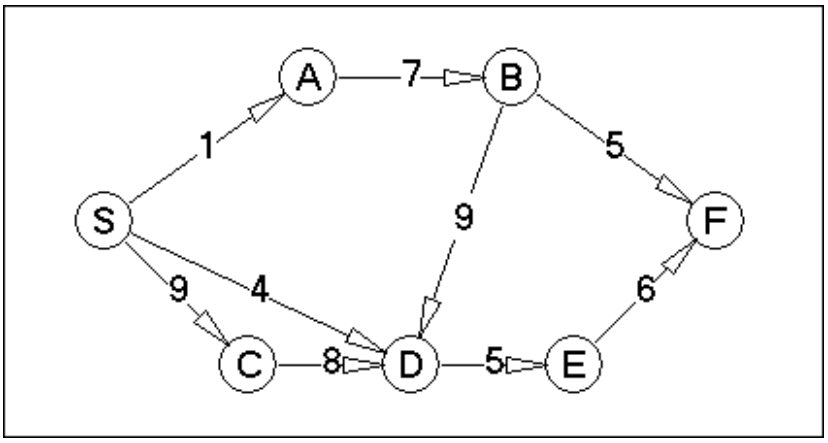
Для данного графа найти наикратчайшее расстояние


A
B
S
F
C
D
E
8
1
5
2
1
2
3
3
7


A
B
S
F
C
D
E
4
9
3
7
8
7
8
3
7


A
B
S
F
C
D
E
2
5
8
1
5
8
9
3
7


A
B
S
F
C
D
E
5
3
9
6
2
5
3
2
2



Задание 2

Между населёнными пунктами A, B, C, D, E, F построены дороги, протяжённость которых приведена в таблице:

	A	B	C	D	E	F
A		6	4	2	1	
B	6		1			
C	4	1		3		2
D	2		3		2	
E	1			2		6
F			2		6	

Определите длину кратчайшего пути между пунктами A и F. Передвигаться можно только по дорогам, протяжённость которых указана в таблице.

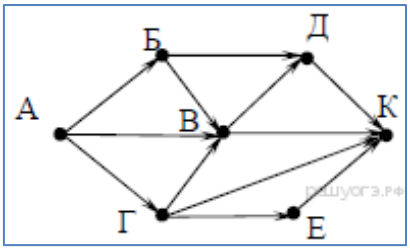
Задание 3

В таблице приведена стоимость перевозок между пятью железнодорожными станциями, обозначенными буквами A, B, C, D и E. Построить схему, соответствующую таблице.

	A	**B**	**C**	**D**	**E**
A		1	4		1
B	1			3	
C	4				2
D		3			
E	1		2		

Задание 4

На рисунке – схема дорог, связывающих города А, Б, В, Г, Д, Е, К. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из города А в город К?



Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Вариант задания
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Скажите, что такое граф?
2. Что является вершинами и ребрами графа?
3. Какой граф можно назвать взвешенным?
4. Какой граф можно назвать сематической сетью?

Практическое занятие № 15Файл как единица хранения информации на компьютере. Атрибуты файла и его объем. Учет объемов файлов при их хранении, передаче

Цель занятия: научиться создавать файл, архивировать, определять объём файла, записывать на различные носители.

Исходные данные: теоретический материал

Содержание и порядок выполнения:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть:

Информационный объект — это совокупность логически связанной информации.

Файл — это определенное количество информации (программа или данные), имеющее имя и хранящееся в долговременной (внешней) памяти.

Имя файла состоит из двух частей, разделенных точкой: собственно имя файла и расширение

Расширение файла - конкретная последовательность символов (букв и цифр), следующая за именем файла через символ точки "." и применяемая для идентификации типа файла программными продуктами и/или пользователем. То есть с помощью расширения файла программы и человек понимают (если знают) какой тип данных заключен в конкретном файле, какими особенностями он обладает, что необходимо для его исполнения.

Тип файла - это определенная спецификация (одна из многих), описывающая действительную структуру файла. В соответствии с данной структурой файл обрабатывается программами и хранится на носителе. Видимая для пользователя часть типа файла - это расширение файла.

Роль расширения имени файла чисто информационная, а не командная. Если файлу с рисунком присвоить расширение имени txt, то содержимое файла от этого не превратится в текст. Его можно просмотреть в программе, предназначенной для работы с текстами, но ничего вразумительного такой просмотр не даст.

Список расширений, наиболее часто встречающихся на компьютере, вместе со списком программ, которые данный тип файла открывают указан в табл. 1.

Табл. 1

Вид расширения	Тип (формат) файла	Программа\утилита
.exe	Исполняемый файл (программный бинарный файл)	Любая рабочая программа Windows, DOS, Symbian, OS/2
.msi	Инсталлятор программ (установщик)	Пакет программ, требующих установки (MS Office, например)
.doc(docx)	Документ Word (Word 2007 и выше)	MS Word, одно из приложений Office
.xls(xlsx)	Файл таблиц Excel (Excel 2007 и выше)	MS Excel, одно из приложений Office
.txt	Текстовый файл простого формата (документ)	Блокнот
.ppt (pptx)	Файл презентаций PowerPoint	MS PowerPoint, одно из приложений Office
.accdb	База данных Access	MS Access, одно из приложений Office
.mp3, .flac, .ape, .ogg, .wav, .ac3, .wma, .m4a, .aac, .midi, .karидр.	Звуковой (цифровой) файл	Любой аудио-плеер (не только Windows) с необходимыми кодеками

.bmp, .jpg(jpeg), .png, .gif, .tiff, .pds, .ico, .raw	Файл изображения	Стандартные менеджеры изображений, иногда специальные программы для конкретного формата (MOPM,ACDC,Stone)
.avi, .wmw, .mkv, .3gp, .flv, .mpeg, .mp4, .mov, .vob	Видео файлы	Различные плееры, главное условие - установлен необходимый кодек для вывода изображений соответствующего формата
.swf, .flv	Флеш или видео файлы в интернете	Воспроизводятся любым браузером с установленным flash-проигрывателем
.rar, .zip, .7z, .tar, .gzip, .gz, .jar	Архивный контейнер	В большинстве случает хватает WinRar и 7-Zip для работы со всеми популярными архивами
.html, .htm, .php	Веб-страница	Браузеры
.dll	Программный модуль	Не должен открываться, подключается в виде библиотеки различными компонентами MS Windows при возникновении необходимости
.ini	Конфигурационный файл	Используется Windows для загрузки настроек различных компонентов, к которым относится конкретный ini-файл
.bat	Пакетный файл. Имеет свой синтаксис, может быть создан из обычного тестового файла заменой расширения	Используется командным интерпретатором (CMD.exe/Command.com) для выполнения заложенных в bat-файл последовательных команд.
.iso, .mds/.mdf, .vdf, .img, .daa, .vcd, .nrg	Файл образа диска	Различные форматы открываются различными программами. Например, Alcohol, UltraISo, VirtualCD, Nero, Daemon Tools, PowerIsoидругими.
.djvu	Файл сжатых изображений. Используется для точной передачи информации (сканированные книги, исторические документы) без потери данных.	Любая программа для чтения djvu-файлов. WinDJView, DJVUReaderит.д.
.pdf	Файл электронного документа, подходит для передачи любой полиграфической продукции и прочего	Adobe Reader, Foxit PDF Reader ипрочие

Полное имя файла состоит из пути к каталогу, в котором находится файл, и имени файла, разделенных \, перед которым может стоять обозначение дисковода. Если дисковод не указан, подразумевается текущий дисковод. Если путь не указан, подразумевается текущий каталог.

дисковод:\путь\имя файла

Текущий дисковод – дисковод, с которым работает пользователь.

Путь к файлу – последовательность из имен каталогов или символов .., разделенных \.

Этот путь задает маршрут от текущего каталога или от корневого каталога диска к тому каталогу, в котором находится нужный файл.

Атрибуты файлов – совокупность байтов, выделяющих файл из множествадругих файлов. Атрибутами файла являются:

- имя файла и типсодержимого;

-дата и время создания файла;

- имя владельца файла;

- размер файла;

- права доступа к файлу;

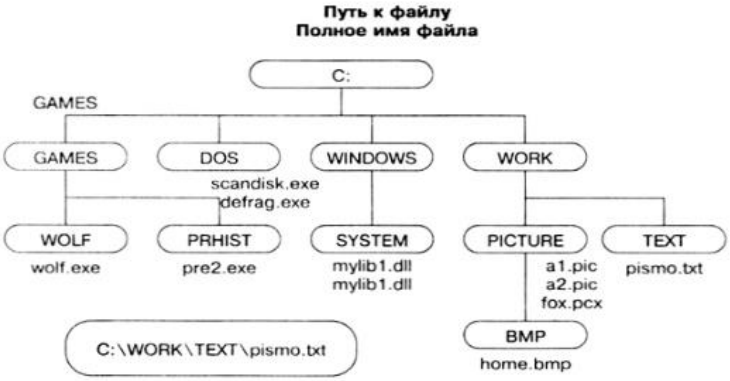
- метод доступак файлу.

При передачи и хранении различных файлов необходимо учитывать объем этих файлов. Если объем слишком велик, можно создать архив файлов с помощью программ архиваторов (7-zip, WinRAR, WinZip).

Файловая система - это функциональная часть операционной системы, обеспечивающая выполнение операций над файлами. Файловая система позволяет работать с файлами и директориями (каталогами) независимо от их содержимого, размера, типа и т. д.

Файловая система определяет общую структуру именования, хранения и организации файлов в операционной системе.

Иерархическая файловая система:



Над файлами могут производиться различные операции:

- Копирование (копия файла помещается из одного каталога в другой)
- Перемещение (сам файл перемещается в другой каталог)
- Удаление (запись о файле удаляется из каталога)
- Переименование (изменяется имя файла) и т.д.

Правила создания имени файла:

В длинных именах нельзя использовать следующие символы:? : * / \ “ >< |

Выполнив задания данной темы, вы научитесь:

- работать в наиболее распространенных программах-архиваторах;
- создавать архивные файлы;
- извлекать файлы из архивов.

Задания:

Задание №1. Предложите варианты имен и типов для перечисленных ниже файлов. Перенесите в тетрадь таблицу и заполните ее.

Содержание	**Имя**	**Тип**	**Полное имя файла**
Фото моей семьи			
Рецепт яблочного пирога			
Буклет «Мой колледж»			
Открытое письмо Биллу Гейтсу	BillG	doc	BillG.doc
Семейный альбом «Моя родословная»			
Репродукция картины Малевича «Черный квадрат»			
Петиция директору колледжа об увеличении числа уроков информатики			
Реферат по истории			
Реклама концерта рок-группы			
Статья в журнал «Информатика и образование»			

Задание № 2. Предложите варианты программ, открывающих файлы с тем или иным расширением. Перечертите таблицу в тетрадь и заполните ее.

Расширение имени файла	**Программа**
TXT	
DOC	
RTF	
BMP	
ARJ	
HTML	

*Задание № 3***.** Выполните задания в тетради.

а) Придумай имя текстового файла, в котором будет содержаться информация о твоем доме. Подчеркни собственное имя файла.

b) Придумай имя графического файла, в котором будет содержаться рисунок твоего дома. Подчеркни расширение файла.

c) Выпиши в один столбик правильные имена файлов, а во второй правильные имена каталогов: **Письмо.18, letter.txt, WinWord, письмо.doc, Колледж?12, Мои документы, роза.bmp, crop12.exe, 1С, red.com**

Имена файлов	Имена каталогов

Задание №4

Определите тип файла по заданному расширению. Укажите пример программного приложения, которое может быть использовано для работы с документами указанного типа. Ответы запишите в виде табл.

Пример	**Тип файла**	**Пример программного приложения**
ACDSee9.exe		
Command.com		
Резюме.doc		
Каталог.xls		
Рецензия.txt		
Выступление.ppt		
Книга.htm		
Книга.html		
Windows.hlp		
Картина.bmp		
Обложка.jpg		
Светофор.tif		
Техникум.gif		
Студент.mp3		
Праздник.mpeg		
Урок.avi		
Реферат.zip		
Реферат.rar		

*Задание 5.*Рассчитайте время передачи файла по интернету, если скорость соединения 128 кбит/сек, а объём файла 3 мбайт. Используется формула

$$время_передачи(c) = \frac{объём_файла(бит)}{скорость(бит/c)}$$

*Задание 6.*Если скорость сети Интернет 512 кбит/сек, то какой объем файла можно скачать за 1,5 мин.

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Вариант задания
4. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»
5. Список используемых источников
6. Выводы и предложения
7. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

Что называют внешней памятью компьютера?

Какова структура диска компьютера?

Какой формулой определяется полный объем памяти диска?

Что называют файлом, каталогом и файловой структурой?

Из чего состоит полное имя файла?

Практическое занятие №16 Файл и файловая структура

Цель занятия:

Выработать практические навыки чтения атрибутов файлов

Выработать практические навыки определение объемов файлов при их наиболее удобном хранении и передаче по сети.

Исходные материалы и данные: ПК, локальная сеть, выход в Интернет

Содержание и порядок выполнения:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

Над файлами могут производиться различные операции:

- Копирование (копия файла помещается из одного каталога в другой)
- Перемещение (сам файл перемещается в другой каталог)
- Удаление (запись о файле удаляется из каталога)
- Переименование (изменяется имя файла) и т.д.

Правила создания имени файла:

1.Нельзя использовать следующие символы, которые зарезервированы для специальных функций: ? . , ; : = + * / \ “ | <> [ ] ПРОБЕЛ

2.В длинных именах нельзя использовать следующие символы:? : * / \ “ >< |

Современные операционные системы поддерживают многоуровневую организацию файлов на дисковых устройствах внешней памяти — иерархическую файловую структуру. Для облегчения понимания этого вопроса воспользуемся аналогией с традиционным «бумажным» способом хранения информации. В такой аналогии файл представляется как некоторый озаглавленный документ (текст, рисунок) на бумажных листах. Следующий элемент файловой структуры называется каталогом. Продолжая «бумажную» аналогию, каталог будем представлять как папку, в которую можно вложить множество документов, т. е. файлов. Каталог также получает собственное имя (представьте, что оно написано на обложке папки).

Каталог сам может входить в состав другого, внешнего по отношению к нему каталога. Это аналогично тому, как папка вкладывается в другую папку большего размера. Таким образом, каждый каталог может содержать внутри себя множество файлов и вложенных каталогов (их называют подкаталогами). Каталог самого верхнего уровня, который не вложен ни в какой другой каталог, называется корневым каталогом.

В операционной системе Windows для обозначения понятия «каталог» используется термин «папка».

Графическое изображение иерархической файловой структуры называется деревом.

На дереве корневой каталог обычно изображается символом ****. На рисунке имена каталогов записаны прописными буквами, а файлов — строчными. Здесь в корневом каталоге имеются две папки: IVANOV и PETROV и один файл fin.com. Папка IVANOV содержит в себе две вложенные папки PROGS и DATA. Папка DATA пустая; в папке PROGS имеются три файла и т. д.

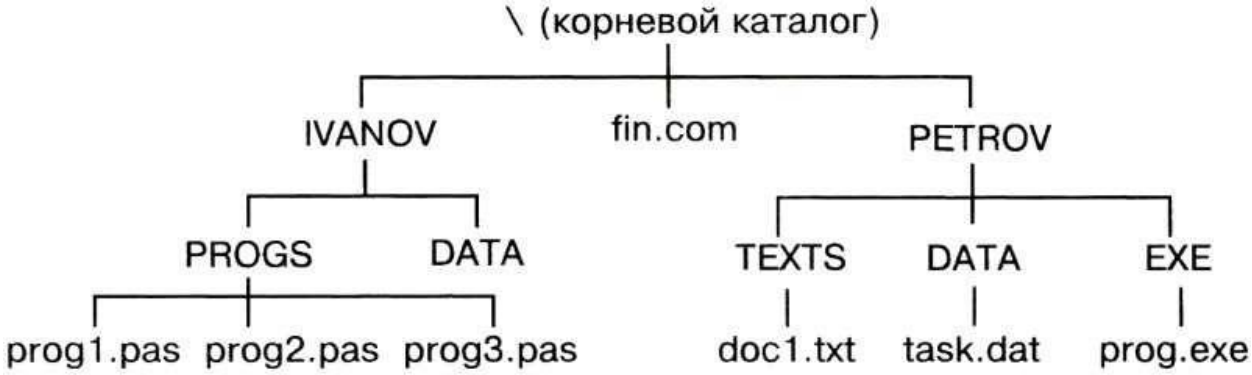


Рис. Пример иерархической файловой структуры

Все файлы храниться по определенной структуре. Её называют файловой структурой диска.

Файловая структура диска- это совокупность файлов на диске и взаимосвязей между ними.

Каталог- это поименованная совокупность атрибутов файлов и подкаталогов на диске. Корневой каталог – это каталог самого верхнего уровня, он никогда не может быть вложенным каталогом.

Путь к файлу- это последовательность, состоящая из имени каталогов, разделенных символом «\» слеш, начиная от корневого и заканчивая тем, в котором непосредственно храниться файл.

Полное имя файла-состоит из имени логического диска, пути к файлу и имени файла. В одном каталоге не может быть несколько файлов и каталогов с одинаковыми именами. В разных каталогах это допустимо.

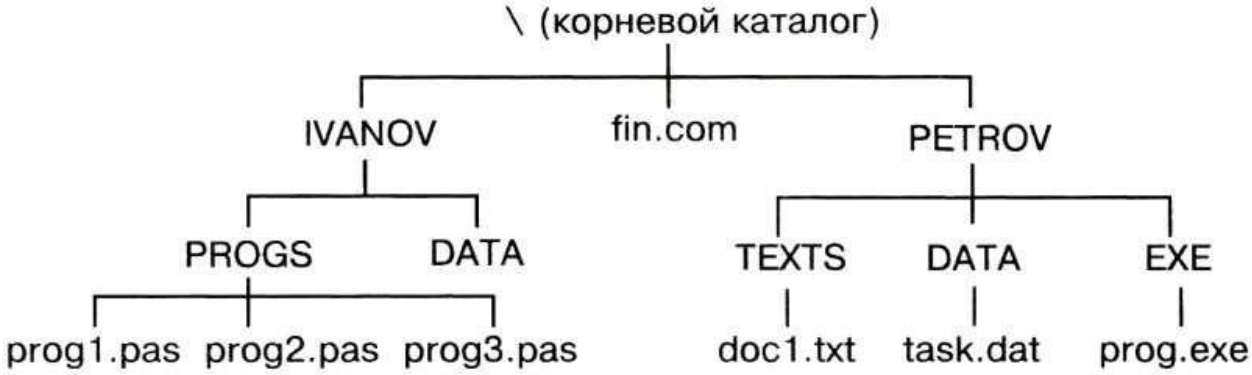


Рис. Пример иерархической файловой структуры

Полные имена некоторых входящих в нее файлов в **символике операционной системы MicrosoftWindows выглядят так**:

C:\fin.com

C: \IV ANOV\PROGS\prog 1. Pas

C: \PETROV\DATA\task. dat

Дерево- это графическое изображение иерархической файловой структуры диска.

Просмотр файловой структуры

Операционная система предоставляет пользователю возможность просматривать на экране содержимое каталогов (папок).

Сведения о файловой структуре диска содержатся на этом же диске в виде таблицы размещения файлов. Используя файловую систему ОС, пользователь может последовательно просматривать на экране содержимое каталогов (папок), продвигаясь по дереву файловой структуры вниз или вверх.

На рисунке показан пример отображения на экране компьютера дерева каталогов в ОС Windows.

Рис Дерево каталогов

В правом окне представлено содержимое папки ARCON. Это множество файлов различных типов. Отсюда, например, понятно, что полное имя первого в списке файла следующее:

E:\GAME\GAMES\ARCON\dos4gw.exe

Из таблицы можно получить дополнительную информацию о файлах. Например, файл dos4gw.exe имеет размер 254 556 байтов и был создан 31 мая 1994 года в 2 часа 00 минут.

Найдя в таком списке запись о нужном файле, применяя команды ОС, пользователь может выполнить с файлом различные действия: исполнить программу, содержащуюся в файле; удалить, переименовать, скопировать файл. Выполнять все эти операции вы научитесь на практическом занятии.

Файловая система. На каждом носителе информации (гибком, жестком или лазерном диске) может храниться большое количество файлов. Порядок хранения файлов на диске определяется установленной файловой системой.

Файловая система- это система хранения файлов и организации каталогов.

Для дисков с небольшим количеством файлов (до нескольких десятков) удобно применять одноуровневую файловую систему, когда каталог (оглавление диска) представляет собой линейную последовательность имен файлов. Для отыскания файла на диске достаточно указать лишь имя файла.

Если на диске хранятся сотни и тысячи файлов, то для удобства поиска файлы организуются в многоуровневую иерархическую файловую систему, которая имеет «древовидную» структуру (имеет вид перевернутого дерева).

Путь к файлу. Для того чтобы найти файл в иерархической файловой структуре необходимо указать путь к файлу. В путь к файлу входят записываемые через разделитель "\" логическое имя диска и последовательность имен вложенных друг в друга каталогов, в последнем из которых находится данный нужный файл.

Чтобы найти файл в компьютере, надо знать логический диск, на котором находится файл, и путь к файлу на диске, определяющий положение файла на этом диске. Путь к файлу — это последовательность, состоящая из имен каталогов, начиная от корневого и заканчивая тем, в котором непосредственно хранится файл. Вот всем знакомая сказочная аналогия понятия «путь к файлу»: «На дубе висит сундук, в сундуке — заяц, в зайце — утка, в утке — яйцо, в яйце — игла, на конце которой смерть Кощеева».

Полное имя файла.

Путь к файлу вместе с именем файла называют полным именем файла. *Пример полного имени файлов:*

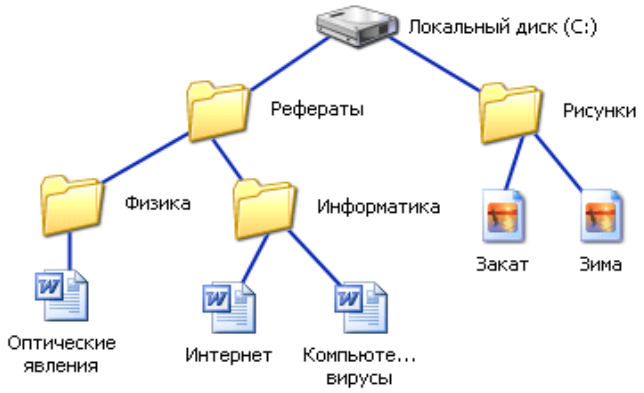
C:\Рефераты\Физика\Оптические явления.doc

C:\Рефераты\Информатика\Интернет.doc

C:\Рефераты\Информатика\Компьютерные вирусы.doc

C:\Рисунки\Закат.jpg

C:\Рисунки\ Зима.jpg



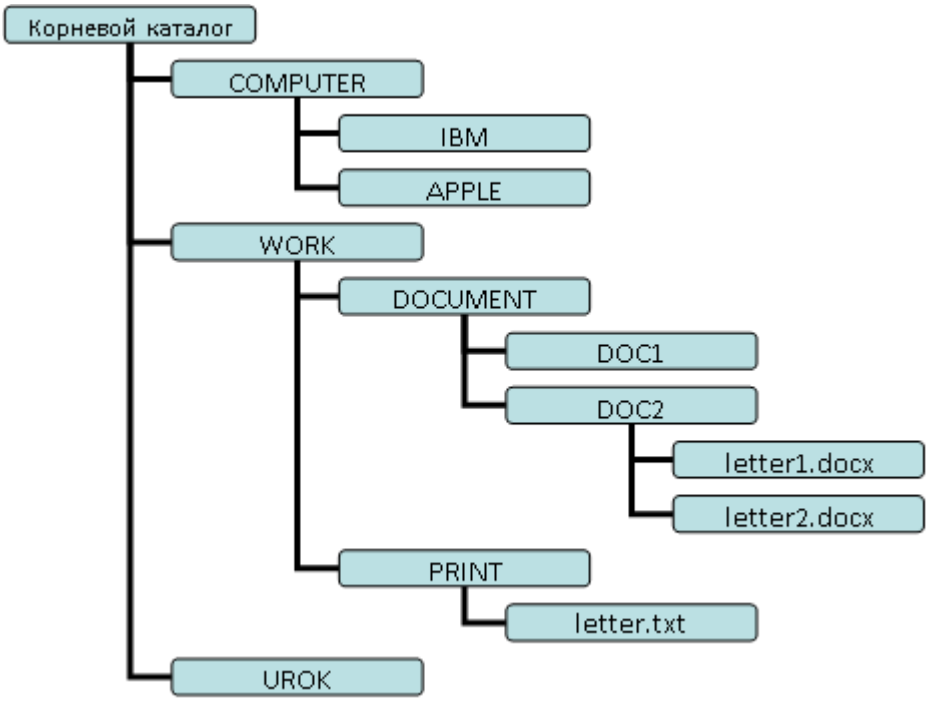
Задание №1

Дано дерево файлов структуры диска.

Заглавными буквами обозначены имена каталогов, строчными – имена файлов.

Перечислите имена каталогов 1-ого, 2-ого, 3 –его уровней. Укажите путь к файлу letter. txt от корневого каталога, путь к файлу letter1.docx от корневого каталога, к файлу letter2.docx от каталога WORK. Укажите полное имя файлов letter1.docx

letter2.docx , если файловая структура хранится на диске D



Задание №2

<u>Постройте дерево каталогов</u>

C:\Рисунки\Природа\Небо.bmp

C:\Рисунки\Природа\Снег.bmp

C:\Рисунки\Компьютер\Монитор.bmp

C:\Мои документы\Доклад.doc

*Задание 3*Записать полный путь ко всем файлам

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки

1. Как называется операционная система, используемая в вашем компьютерном классе?

2. Каким правилам подчиняются имена файлов в вашей ОС?

3. Что такое путь к файлу на диске, полное имя файла?

Практическое занятие №17 Сжатие информации. Алгоритм Хаффмана.Создание архива данных. Извлечение данных из архива

Цель занятия:

- изучение принципов архивации файлов, функций и режимов работы наиболее распространенных архиваторов,

- приобретение практических навыков работы по созданию архивных файлов и извлечению файлов из архивов.

Исходные материалы: ПК, ОС Windows, программа для архивации данных WinZip, WinRar.

Теоретические сведения

Архивация (упаковка) — помещение (загрузка) исходных файлов в архивный файл в сжатом или несжатом виде.

Архивация предназначена для создания резервных копий используемых файлов, на случай потери или порчи по каким-либо причинам основной копии (невнимательность пользователя, повреждение магнитного диска, заражение вирусом и т.д.).

Для архивации используются специальные программы, архиваторы, осуществляющие упаковку и позволяющие уменьшать размер архива, по сравнению с оригиналом, примерно в два и более раз.

Архиваторы позволяют защищать созданные ими архивы паролем, сохранять и восстанавливать структуру подкаталогов, записывать большой архивный файл на несколько дисков (многотомный архив).

Сжиматься могут как один, так и несколько файлов, которые в сжатом виде помещаются в так называемый архивный файл или архив. Программы большого объема, распространяемые на дискетах, также находятся на них в виде архивов.

Архивный файл — это специальным образом организованный файл, содержащий в себе один или несколько файлов в сжатом или несжатом виде и

служебную информацию об именах файлов, дате и времени их создания или модификации.

Выигрыш в размере архива достигается за счет замены часто встречающихся в файле последовательностей кодов на ссылки к первой обнаруженной последовательности и использования алгоритмов сжатия информации.

Степень сжатия зависит от используемой программы, метода сжатия и типа исходного файла. Наиболее хорошо сжимаются файлы графических образов, текстовые файлы и файлы данных, для которых степень сжатия может достигать 5 - 40%, меньше сжимаются файлы исполняемых программ и загрузочных модулей — 60 - 90%. Почти не сжимаются архивные файлы. Программы для архивации отличаются используемыми методами сжатия, что соответственно влияет на степень сжатия.

Для того чтобы воспользоваться информацией, запакованной в архив, необходимо архив раскрыть или распаковать. Это делается либо той же программой-архиватором, либо парной к ней программой-разархиватором.

Разархивация (распаковка) — процесс восстановления файлов из архива в первоначальном виде. При распаковке файлы извлекаются из архива и помещаются на диск или в оперативную память.

Самораспаковывающийся архивный файл — это загрузочный, исполняемый модуль, который способен к самостоятельной разархивации находящихся в нем файлов без использования программы-архиватора.

Самораспаковывающийся архив получил название SFX-архив (SelF-eXtracting). Архивы такого типа в обычно создаются в форме .EXE-файла.

Архиваторы, служащие для сжатия и хранения информации, обеспечивают представление в едином архивном файле одного или нескольких файлов, каждый из которых может быть при необходимости извлечен в первоначальном виде. В *оглавлении архивного файла* для каждого содержащегося в нем файла хранится следующая информация:

- имя файла;
- сведения о каталоге, в котором содержится файл;
- дата и время последней модификации файла;
- размер файла на диске и в архиве;
- код циклического контроля для каждого файла, используемый для проверки целостности архива.

***Архиваторы имеют следующие функциональные возможности**:*

1) Уменьшение требуемого объема памяти для хранения файлов от 20% до 90% первоначального объема.

2) Обновление в архиве только тех файлов, которые изменялись со времени их последнего занесения в архив, т.е. программа-упаковщик сама следит за изменениями, внесенными пользователем в архивируемые файлы, и помещает в архив только новые и измененные файлы.

3) Объединение группы файлов с сохранением в архиве имен директорий с именами файлов, что позволяет при разархивации восстанавливать полную структуру директорий и файлов.

4) Написания комментариев к архиву и файлам в архиве.

5) Создание саморазархивируемых архивов, которые для извлечения файлов не требуют наличия самого архиватора.

6) Создание многотомных архивов – последовательности архивных файлов. Многотомные архивы предназначены для архивации больших комплексов файлов на дискеты.

Задание №1.

1. В операционной системе Windows создайте папку **Archives** по адресу **C:\TEMP**. Создайте папки **Pictures** и **Documents** по адресу **C:\TEMP\Archives**.

2. Найдите и скопируйте в папку **Pictures**по два рисунка с расширением ***.jpg**и ***.bmp**.

3. Сравните размеры файлов ***.bmp** и ***.jpg**. и запишите данные в таблицу_1.

4. В папку **Documents** поместите файлы ***.doc**(не менее 3) и запишите их исходные размеры в таблицу_1.

Задание №2. Архивация файлов WinZip

1. Запустите **WinZip 7.** (**Пуск →Все программы → 7-Zip→7 ZipFileManager**).

2. В появившемся диалоговом окне выберите папку, в которой будет создан архив: **C:\TEMP\Archives\Pictures.** Установите курсор на имя графического файла **Зима.jpg**. Выполните команду **Добавить (+)**.

3. Введите имя архива в поле **Архив** – **Зима.zip** и убедитесь, что в поле **Формат архива** установлен тип **Zip**.

4. Установите в поле **Режим изменения**: *добавить и заменить.*

5. В раскрывающемся списке **Уровень сжатия:** выберите пункт **Нормальный.** Запустите процесс архивации кнопкой **ОК**.

6. Сравните размер исходного файла с размером архивного файла. Данные запишите в таблицу_1.

7. Создайте архив **Зима1.zip**, защищенный паролем. Для ввода пароля в диалоговом окне **Добавит к архиву** в поле **Введите пароль:** ведите пароль, в поле **Повторите пароль:** подтвердите пароль. Обратите внимание на флажок**Показать пароль.** Если он не установлен, пароль при вводе не будет отображаться на экране, а его символы будут заменены подстановочным символом "*". Это мера защиты пароля от посторонних. Однако в данном случае пользователь не может быть уверен в том, что он набрал пароль правильно. Поэтому при не установленном флажке система запрашивает повторный (контрольный) ввод пароля. Щелкните на кнопке **ОК** - начнется процесс создания защищенного архива.

8. Выделите архив **Зима1.zip**, выполните команду **Извлечь.** В появившемся диалоговом окне **Извлечь** в поле **Распаковать в:** выберите папку-приемник - **C:\TEMP\Archives\Pictures\Зима1\.**

9. Щелкните на кнопке **ОК**. Процесс извлечения данных из архива не запустится, а вместо него откроется диалоговое окно для ввода пароля.

10. Убедитесь в том, что ввод неправильного пароля не позволяет извлечь файлы из архива.

11. Убедитесь в том, что ввод правильного пароля действительно запускает процесс.

12. Удалите созданный вами защищенный архив и извлеченные файлы.

13. Создайте самораспаковывающийся ZIP-архив. Для этого установите курсор на имя архива **Зима.zip**, выполните команду **Добавить (+).**

14. Введите имя архива в поле **Архив** – **Зима.7z** и убедитесь, что в поле **Формат архива** установлен тип **7z**.

15. Установите в поле **Режим изменения**: *добавить и заменить.*

16. Установите флажок **СоздатьSFX-архив**.

17. В раскрывающемся списке **Уровень сжатия:** выберите пункт **Нормальный.** Запустите процесс архивации кнопкой **ОК**.

18. Аналогичным образом создайте архивы для файлов Рябина.bmp, Документ1.doc, Документ2.doc, Документ3.doc. Сравнительные характеристики исходных файлов и их архивов занести в таблицу_1.

Задание №3. Архивация файлов WinRar

1. Запустите **WinRar**(**Пуск →Все программы → WinRar**).

2. В появившемся диалоговом окне выберите папку, в которой будет создан архив: **C:\TEMP\Archives\Pictures.**

3. Установите курсор на имя графического файла **Зима.jpg**.

4. Выполните команду **Добавить**. В появившемся диалоговом окне введите имя архива **Зима.rar.** Выберите формат нового архива - **RAR**, метод сжатия - **Обычный**. Убедитесь, что в группе **Параметры архивации** ни в одном из окошечек нет флажков. Щелкните на кнопке **ОК** для создания архива. Во время архивации отображается окно со статистикой. По окончании архивации окно статистики исчезнет, а созданный архив станет текущим выделенным файлом.

5. Аналогичным образом создайте архивы для файлов Рябина.bmp, Документ1.doc, Документ2.doc, Документ3.doc. Сравнительные характеристики исходных файлов и их архивов занести в таблицу_1.

6. Создайте самораспаковывающийся RAR – архив, включающий в себя текстовые и графические файлы.

7. Определите процент сжатия файлов и заполните таблицу_1. Процент сжатия определяется по формуле $P = \frac{S}{S_0} * 100\%$, где S – размер архивных файлов, So– размер исходных файлов.

Таблица_1

	Архиваторы		**Размер исходных файлов**
	WinZip	**WinRar**	
Текстовые файлы: 1. Документ1.doc			
2. Документ2.doc			
3. Документ3.doc			
Графические файлы: 1. Зима.jpg			
2.Рябина.bmp			
Процент сжатия ***текстовой*** информации (для всех файлов)			
Процент сжатия ***графической*** информации (для всех файлов)			

*Задание №4***.** Ответить на вопросы:

1 Что называется архивацией?	
2.Для чего предназначена архивация?	

3. Какой файл называется архивным?	
4. Что называется разархивацией?	
5. Какая информации хранится в оглавлении архивного файла?	
6. Какие функциональные возможности имеют архиваторы?	

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Вариант задания
4. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»
5. Список используемых источников
6. Выводы и предложения
7. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки

1. Как определяются размеры носителей информации?
2. Как сделать файл меньше информационным объемом?
3. Какие программы архиваторы вам известны?
4. Как определяется коэффициент сжатия файла?
5. Какие файлы лучше сжимаются?

Практическое занятие №18Алгоритмы и способы их описания. Линейные алгоритмы

Цели занятия:

- сформировать представление об алгоритме и его свойствах;
- сформировать представление о способах их описания алгоритмов;
- сформировать представление о типах алгоритмов;
- сформировать представление о линейной алгоритмической конструкции;

–научить создавать трассировочные таблицы линейной алгоритмической конструкции.

Исходные данные: теоретический материал

Содержание и порядок выполнения задания

Изучите теоретическую часть

Выполните задания

Теоретическая часть

Слово ***алгоритм*** происходит от латинской формы написания имени великого математика IX века **Аль-Хорезми**, который сформулировал правила выполнения арифметических действий.

Первоначально под алгоритмами понимали только правила выполнения четырёх арифметических действий над многозначными числами.

Алгоритм – описание последовательности действий (план), строгое исполнение которых приводит к решению поставленной задачи за конечное число шагов.

Алгоритмизация – процесс разработки алгоритма (плана действий) для решения задачи.

Шаг алгоритма – это каждое отдельное действие алгоритма.

Исполнитель – это объект, умеющий выполнять определенный набор действий. Исполнителем может быть человек, робот, животное, компьютер.

Система команд исполнителя (СКИ) – это все команды, которые исполнитель умеет выполнять.

Среда исполнителя – обстановка, в которой функционирует исполнитель.

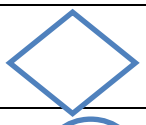
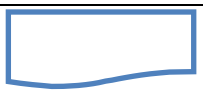
Свойства алгоритма:

- Дискретность - (прерывность, раздельность) – разбиение алгоритма на шаги
- Результативность - получение результата за конечное количество шагов
- Массовость - использование алгоритма для решения однотипных задач
- Конечность - каждое действие в отдельности и алгоритм в целом должны иметь возможность завершения
- Детерминированность - (определенность, точность) – каждое действие должно быть строго и недвусмысленно определено

Способы записи алгоритмов (блок-схема)

Условное обозначение	Назначение блока
	Начало или **конец** алгоритма

Документ управляется программными средствами 1С: Колледж
Проверь актуальность версии по оригиналу, хранящемуся в 1С: Колледж

	Ввод или **вывод** данных. Внутри блока перечисляются данные через запятую.
	Процесс. Внутри блока записываются математические формулы и операции для обработки данных.
	**Проверка условия.** Внутри блока записываются логические условия. Имеет два выхода **Да(+)**и**Нет(-)**.
	Соединительный блок
	**Блок вывода информации на печатающее устройство**
	Блок вывода информации на экран дисплея
	Направление.

Алгоритмы могут быть заданы:

- *словесно*
- *таблично*
- *графически*

Словесное задание описывает алгоритм с помощью слов и предложений естественного языка.

Табличное задание служит для представления алгоритма в форме таблиц и расчётных формул.

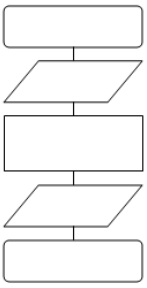
Графическоезадание или ***блок-схема*** – способ представления алгоритма с помощью геометрических фигур, называемых ***блоками***.

Типы алгоритмов

<u>Алгоритмы бывают:</u>

- *линейные*
- *разветвляющиеся*
- *циклические*

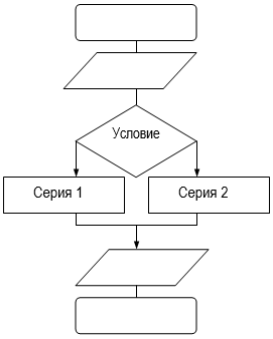
Алгоритм, в котором команды выполняются последовательно одна за другой, называется ***линейным алгоритмом.***



В ***разветвляющиеся алгоритмы*** входит условие, в зависимости от выполнения или невыполнения которого выполняется та или иная последовательность команд (серий).

В алгоритмической структуре ***«ветвление»*** та или иная серия команд выполняется в зависимости от истинности ***условия.***

Условие может быть либо истинным, либо ложным.

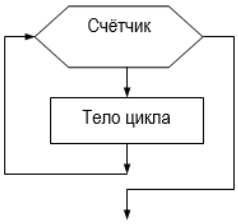


В ***циклические алгоритмы*** входит последовательность команд, выполняемая многократно. Такая последовательность команд называется ***телом цикла.***

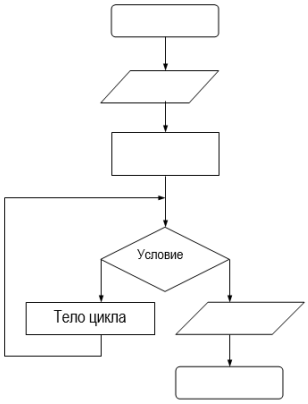
В алгоритмической структуре ***«цикл»*** серия команд (тело цикла) выполняется многократно.

Циклические алгоритмические структуры бывают двух типов:

- *циклы со счётчиком*, в которых тело цикла выполняется определённое количество раз;



- *циклы с условием, в которых тело цикла выполняется, пока условие истинно.*



Примеры выполнения заданий работы

Пример 1. Определить площадь трапеции по введенным значениямоснований (a и b) и высоты (h).

Запись алгоритма в виде блок-схемы (рис. 1):

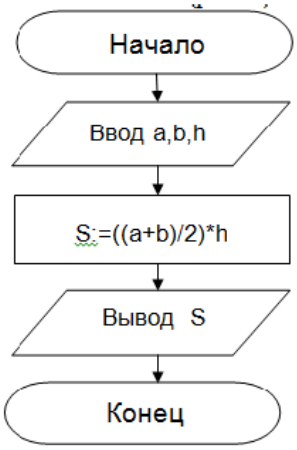


Рисунок 1. Блок-схема линейного алгоритма

Запись решения задачи на алгоритмическом языке:

алгтрапеция

веща,b,h,s

нач

 ввода,b,h

s:=((a+b)/2)*h

выводs

кон

Задание1

Составить алгоритм решения нахождения площадь поверхности и объем усеченного конуса с помощью блок-схемы иалгоритмического языка псевдокод, используя конструкцию линейногоалгоритма.

$$S = \pi (R + r) l + \pi R^2 + \pi r^2 ;$$
$$V = (1/3) \pi (R^2 + r^2 + Rr) h .$$

Пример 2.Дан алгоритм в виде блок-схемы (рис. 11).

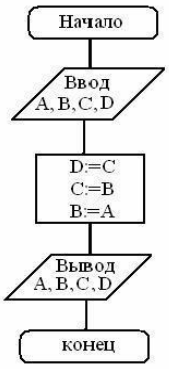
Найти A, B, C, D, если изначально:

а)A=0,B=0,C=5,D=10

б) A=0,B=5,C=0, D=10

в) A=10,B=20, C=6, D=4

г)A=10,B=10,C=4, D=0



Результат работы алгоритма определяется с помощьютрассировочныхтаблиц (а, б, в, г):

а) A=0, B=0, C=5, D=10.

Шаг	1	
Исходныезначения	A	0
	B	0
	C	5
	D	10
Результатвыполнения	A	0
	B	0
	C	0
	D	5
Выводзначений	0,0,0,5	

б) A=0, B=5,C=0, D=10.

Шаг	1	
Исходныезначения	A	0
	B	5
	C	0
	D	10
Результатвыполнения	A	0
	B	0
	C	5
	D	0
Выводзначений	0,0,5,0	

в)A=10, B=20, C=6,D=4.

Шаг	1

Исходныезначения	A	10
	B	20
	C	6
	D	4
Результатвыполнения	A	10
	B	10
	C	20
	D	6
Выводзначений	10,10,20,6	

г)A=10,B=10, C=4, D=0.

Шаг	1	
Исходныезначения	A	10
	B	10
	C	4
	D	0
Результатвыполнения	A	10
	B	10
	C	10
	D	4
Выводзначений	10,10,10,4	

Задание **2** Реализован некоторый алгоритм в виде блок- схемы. Найти А, Вна выходе блок-схемы с заполнение трассировочных таблиц, если изначально, :

1 вариант A=0,=0;

2вариантA=0,B=5;

3 вариант A=10,B=20;

4 вариантA=10,B=10.

5 вариант A=3, B=0;

6 вариант A=0, B=2;

7вариантA=1,B=2;

8 вариант A=12, B=15

9 вариант A=0, B=0;

10 вариант A=2, B=4;

11 вариант A=12, B=24;

12 вариант A=11, B=19

13 вариант A=3, B=8;

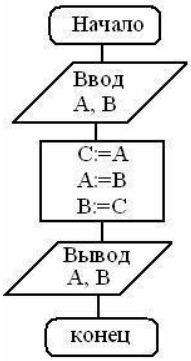
14 вариант A=2, B=7;

15 вариант A=19, B=20;

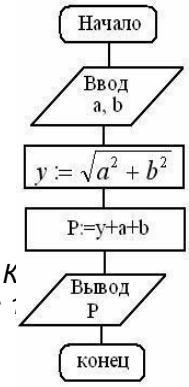
16 вариант A=15, B=10

17 вариант A=60, B=20;

18 вариант A=20, B=65;



Задание **3.** Даны длины двух катетов (a, b) прямоугольного треугольника. Определить периметр этого треугольника (P)с заполнением трассировочных таблиц (см. блок-схему на рис.), если:



a) a=3,b=4;

Документ управляется программными средствами 1С: К
Проверь актуальность версии по оригиналу, хранящемуся в 1

б) a=0,b=3;

в)a=6,b=8;г)a=9,b=12.

***Задание*4.** Реализован некоторый алгоритм в виде блок- схемы (рис.). По данной блок-схеме вычислить S, заполнить трассировочные таблицы, если:

а)a=1, b=2, c=3;

б)a=9, b=0, c=1;

в) a=5, b=6, c=9.

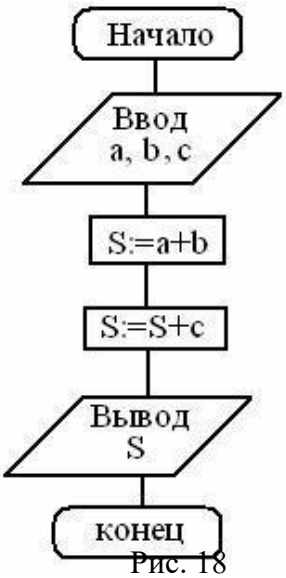


Рис. 18

***Задание*5.** Дана блок-схема (рис.19). Начальные условия: a=8, b=2. Тогда после исполнения алгоритмазначение переменной g будет равно …

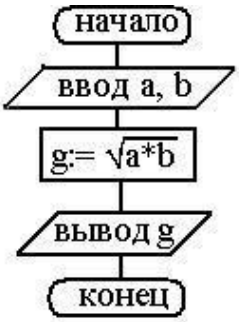


Рис. 19

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Список используемых источников
4. Выводы и предложения
5. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Что такое алгоритм?
2. Свойства алгоритма.
3. Способы записи алгоритма.
4. Основные элементы блок-схемы.
5. Виды алгоритмов.

Практическое занятие №19 Алгоритмы и способы их описания. Условный и циклический алгоритм

Цели занятия:

- сформировать представление об алгоритме и его свойствах;
- сформировать представление о способах их описания алгоритмов;
- сформировать представление о типах алгоритмов;
- сформировать представление о циклических и условных алгоритмических конструкциях;

–научить создавать трассировочные таблицы условных и циклических алгоритмических конструкций.

Исходные данные: теоретический материал

Содержание и порядок выполнения задания

Повторите теоретическую часть, данную в практическом занятии №19.

Изучите примеры 1 и 2 и выполните задания. Результат работы алгоритма определяется с помощьютрассировочных таблиц.

Пример 1. Определить среднее арифметическое двух чисел, если аположительное и частное (a/b) в противном случае.

Запись алгоритма в виде блок-схемы (рис. 2):

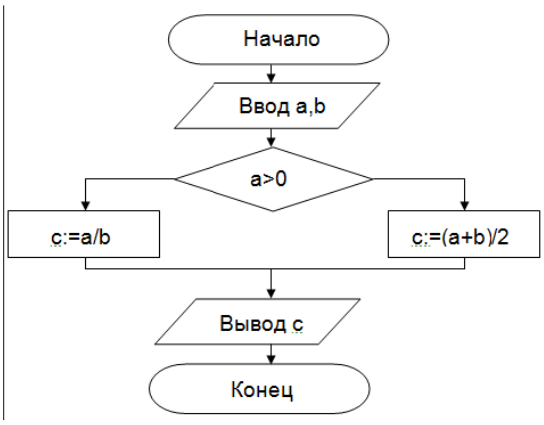


Рисунок 2. Блок-схема алгоритма с ветвлением

Запись решения задачи на алгоритмическом языке:

алг числа

вещ a,b,c

нач

ввода,b

еслиа>0

то c:=(a+b)/2

иначе c:=a/b

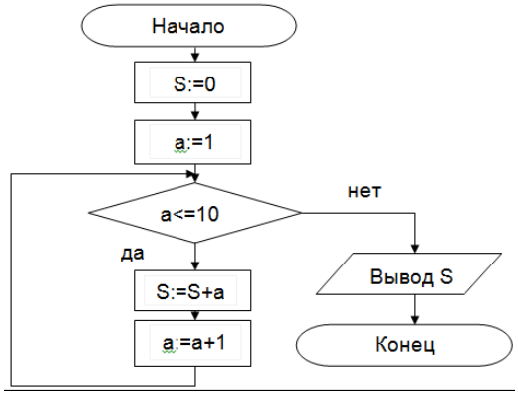
все

вывод c

кон

Пример 2. Составить алгоритм нахождения суммы целых чисел в диапазонеот 1 до 10.

Запись алгоритма в виде блок-схемы (рис.):



Запись решения задачи на алгоритмическом языке:

алг сумма
вещ a,s

нач
S:=0;
A:=1;
нц
покаа<=10
S:=S+a;
A:=a+1;
кц
выводS
кон

В алгоритме с постусловием сначала выполняется тело цикла, а затемпроверяется условие окончания цикла. Решение задачи нахождения суммыпервых десяти целых чисел в данном случае будет выглядеть следующимобразом:

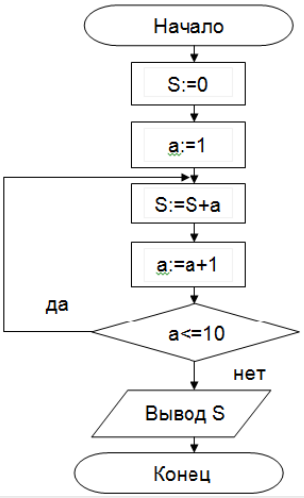
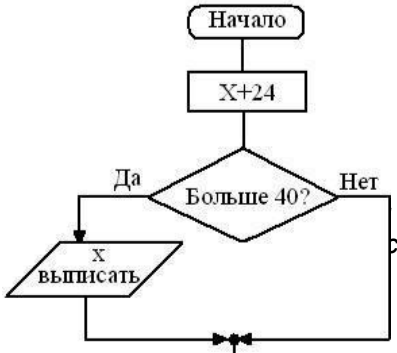


Рис. Циклический алгоритм с постусловием

Запись алгоритма на алгоритмическом языке:

```
алг сумма
    вещ a,s
нач
    S:=0;
    A:=1;
    нц
        S:=S+a;
        A:=a+1;
        пока a<=10
    кц
    вывод S
кон
```

Пример 3.Из ряда чисел 15, 16, 17, 18 выписать значениях, удовлетворяющие условию (см.блок-схемуна рис).



программными средствами 1С: Колледж
сии по оригиналу, хранящемуся в 1С: Колледж

Рис.

Используятрассировочнуютаблицу,получим:

Шаг	1	2	3	4
Исходное значениех	15	16	17	18
Результат выполнения	15+24	16+24	17+24	18+24
Телоцикла	15+24 >40 (Нет)	16+24 >40 (Нет)	17+24>40 (Да)	18+24>40 (Да)
Выводх	–	–	17	18

Задание 1.Составить алгоритм решения задачи с помощью блок-схемы иалгоритмического языка псевдокод, используя конструкцию алгоритма светвлением.

Составить программу для решения квадратного уравнения ax^2+ bx + c = 0.

Задание 2.Составить алгоритм решения задачи с помощью блок-схемы иалгоритмического языка псевдокод, используя конструкцию циклическогоалгоритма.

Найти сумму чисел, кратных трем, в диапазоне от 0 до 30.

Задание3. Вычислить значение функции y. Результат работы алгоритма определяется с помощьютрассировочных таблиц:

(см. блок-схему на рис.), если: а) x=0; б) x=1; в) x= – 5.

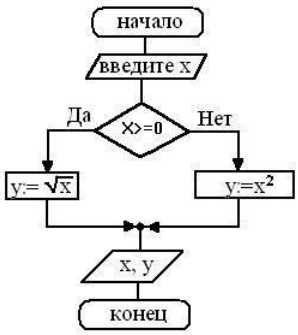
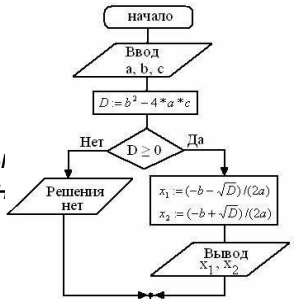


Рис.

Задание4. Используя блок-схему (рис.), найти корни уравнения $ax^2 + bx + c = 0$ ($a \neq 0$) , если:



Документ управляется программны *пледж*
Проверь актуальность версии по оригин *: Колледж*

а)а=1, b=2,

б)а=1, b=4,

в) а=3, b=–8,с=3.

Рис.

Результат работы алгоритма определяется с помощьютрассировочных таблиц

Задание5. Реализован некоторый алгоритм в виде блок- схемы (рис. 22). Что получится на выходе блок-схемы, если:

а) х=0, у=1; б) х=2, у=4; в) х=6, у=0?

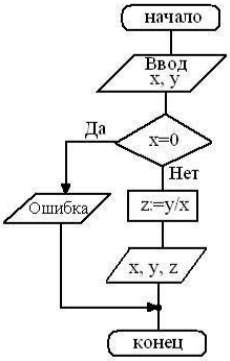


Рис.22

Результат работы алгоритма определяется с помощьютрассировочных таблиц

Задание6. На блок-схеме (рис. 23) представлен алгоритм вычисления стоимости покупки с учетом скидки, где а – цена, b

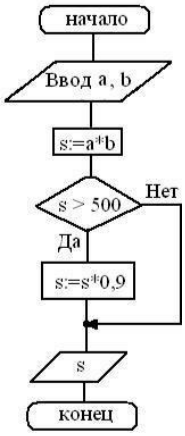
– количество, s – сумма. Какой будет результат на выходе блок- схемы, если:

а) а=50, b=8;

б) а=200, b=5;

в) а=300, b=1;

г) а=800, b=4?



Результат работы алгоритма определяется с помощьютрассировочных таблиц

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Вариант задания
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Что называется алгоритмом?
2. Для чего необходимы трассировочные таблицы?
3. Как представляется блок-схемой циклический алгоритм с пост условием?
4. Как представляется блок-схемой циклический алгоритм с пред условием?

Практическое занятие № 20 Описание алгоритма с помощью блок-схем, выполненных в программе Dia

Цель занятия:Освоитьспособы создания блок-схем различных типов алгоритма в программе Dia.

*Исходные данные:*теоретический материал

Содержание и порядок выполнения задания

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

Правила изображения блок-схем

1.При составлении блок-схем используются только строго определенные типы блоков:

a)***начало*** **и *конец* алгоритма;**

b)***обработки*** (внутри блока записываются формулы, обозначения операций и функций);

c)***условия*** (внутри блока записываются условия выбора направления действия алгоритма);

*d)**ввода/вывода*** информации;

e)***вывода*** информации на экран дисплея;

f)***вывода*** информации на печатающее устройство;

*g)**вычисления по подпрограмме* или стандартной**подпрограмме;

*h)**начало цикла*;**

*i)**соединительный блок*;**

j)***линии потока*** (если поток направлен вниз иливправо стрелку не ставят).

2.Все блоки нумеруются. Номера ставятся вверху слева от блока (блоки "начало", "останов" и соединительные блоки не нумеруются).

3.Стрелки не ставят, если поток направлен сверху вниз или слева направо.

4.Каждый блок имеет единственную точку входа, кроме блока "начало", который не имеет входа.

5.Каждый безусловный блок имеет единственную точку выхода, кроме блока "останов", который не имеет выхода.

6.Условный блок имеет два, в отдельных случаях три выхода.

7.Выход условного блока можно пометить "да", "нет", "+", "–", 0, 1.

8.Линии, идущие на вход некоторого блока, могут соединяться, что соответствует переходу на конкретный этап вычислений после нескольких других этапов.

9.Линия, исходящая из входной точки блока, не может разветвляться на несколько направлений.

Правила построения алгоритмов на языке блок-схем

1.Блок-схема строится сверху вниз.

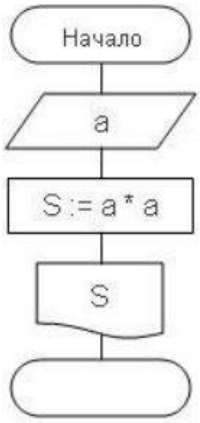
2.В любой блок-схеме имеется только один элемент, соответствующий началу алгоритма, и один элемент, соответствующий концу алгоритма.

3.Должен быть хотя бы один путь из начала блок-схемы к любому элементу.

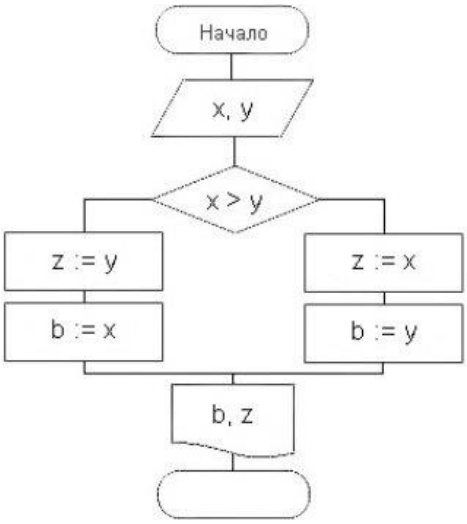
4.Должен быть хотя бы один путь от каждого элемента блок-схемы в конец блок-схемы.

Задание 1. Создание линейного алгоритма.

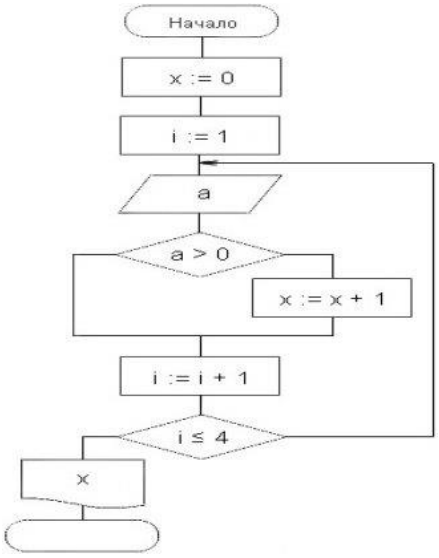
Запустить DIA. Создать алгоритм программы, с помощью фигур. Для этого в пункте меню Вставка выбираем Фигуры и соответствующий элемент библиотеки - блок-схемы.



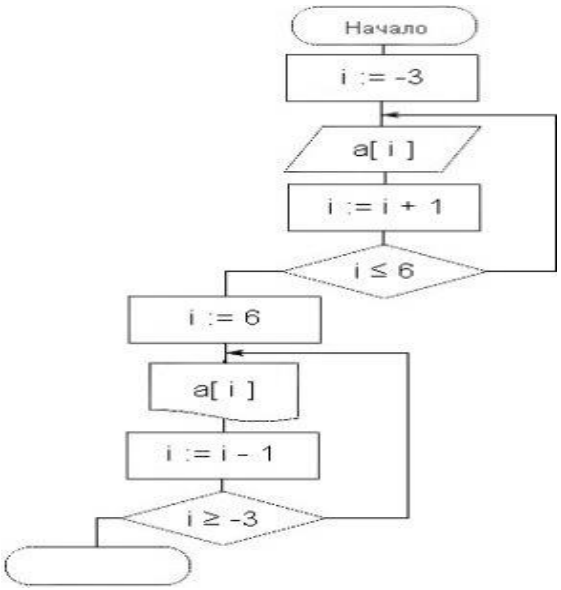
Задание 2. Создание алгоритма ветвления



Задание 3. Создание алгоритма цикла



Задание 4. Создание алгоритма массива



Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Вариант задания
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1. Что называется алгоритмом?
2. Для чего предназначена программа Dia?
3. Какие инструменты есть в программе Dia для создания блок-схем различных алгоритмических конструкций?
4. Какие еще существуют библиотеки элементов в программе Dia?

Практическое занятие №21 Кумир. Знакомство с исполнителем Робот. Простейшие и линейные алгоритмы

Цели занятия:

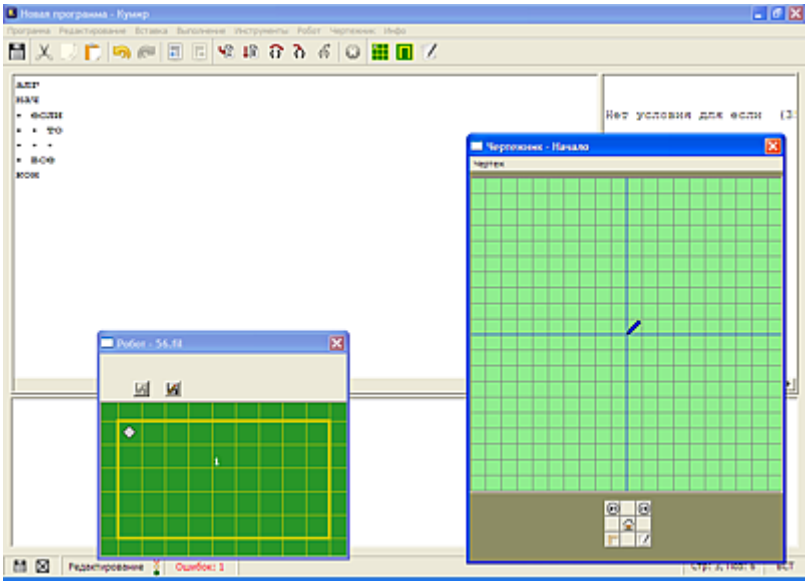
- познакомиться со средой программирования Кумир;
- научиться составлять линейные алгоритмы для исполнителя Робот.

*Исходные данные:*теоретический материал

Содержание и порядок выполнения задания

Изучить теоретическую часть

Выполнить задания



Теоретическая часть

КуМир (Комплект Учебных МИРов) - система программирования, предназначенная для поддержки начальных курсов информатики и программирования.

Особенности системы КуМир

- В системе КуМир используется алгоритмический язык с русской лексикой и встроенными исполнителями Робот и Чертёжник.
- При вводе программы КуМир осуществляет постоянный полный контроль ее правильности, сообщая на полях программы об всех обнаруженных ошибках.
- При выполнении программы в пошаговом режиме КуМир выводит на поля результаты операций присваивания и значения логических выражений. Это позволяет ускорить процесс освоения азов программирования.
- Кумир работает в операционных системах Windows, MacOS и GNU/Linux.

Система Кумир разработана в ФГУ ФНЦ НИИСИ РАН по заказу Российской Академии Наук и распространяется свободно на условиях лицензии GNU 2.0.

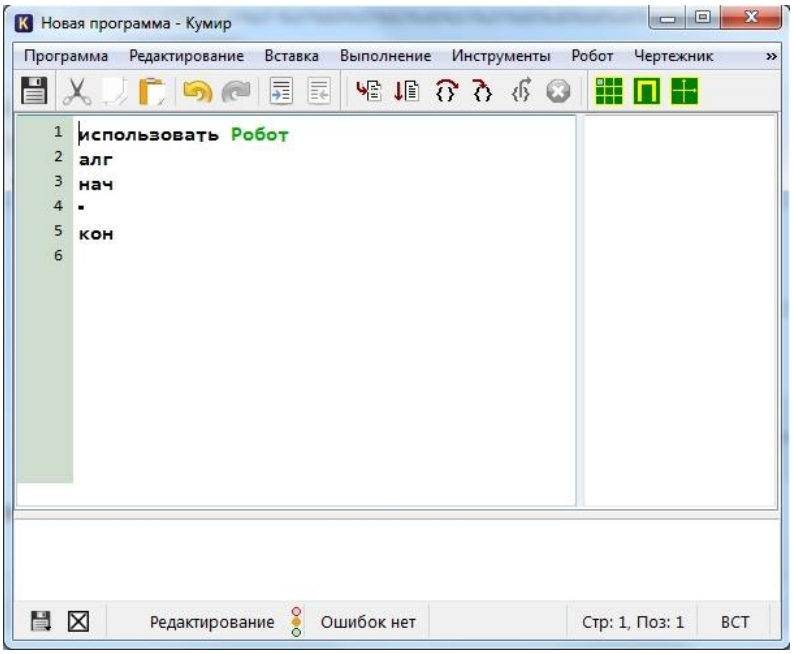
Исполнитель Робот

Кто такой исполнитель Робот?

Представьте себе клетчатое поле (как лист из тетради в клеточку) на котором находится некий объект, который мы назовем Робот. Используя специальные команды, мы можем этим Роботом управлять — перемещать его по клеткам, закрашивать клетки. И в большинстве случаев наша задача будет заключаться в том, чтобы написать такую программу для Робота, выполняя которую он будет закрашивать определенные клетки.

Настройка среды Кумир для исполнителя Робот.

Первым делом мы должны раскомментировать первую строку нашей программы, убрав символ **|**



Таким образом, программа станет выглядеть так:

использовать Робот
алг
нач
кон

Удалив символ |, мы тем самым указали Кумиру на то, что будем работать с исполнителем Робот. Если этого не сделать, то при написании программы мы столкнемся с ошибкой «Нет такого алгоритма». Поэтому очень важно при создании новой программы раскомментировать первую строку. Теперь все готово для дальнейшей работы.

Но перед началом, нам необходимо задать стартовую обстановку Робота и познакомиться с простыми командами исполнителя Робот.

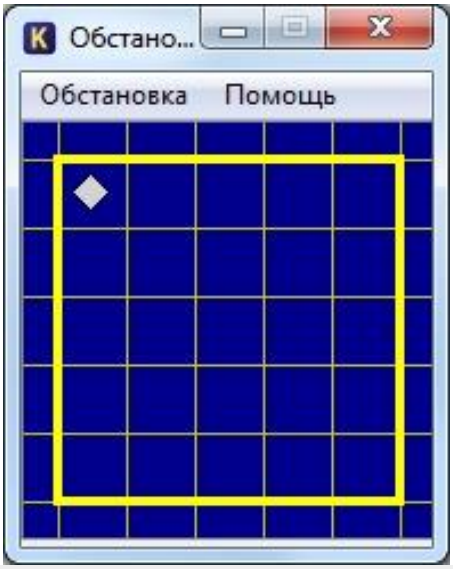
Стартовая обстановка Робота

Перед началом выполнения программы необходимо задать исполнителю Робот стартовую обстановку. Это значит установить Робота в нужную позицию, расставить стены, закрасить нужные клетки и т. п. Этот шаг очень важен. Если его проигнорировать, то программа может работать неправильно или вообще завершится аварийно.

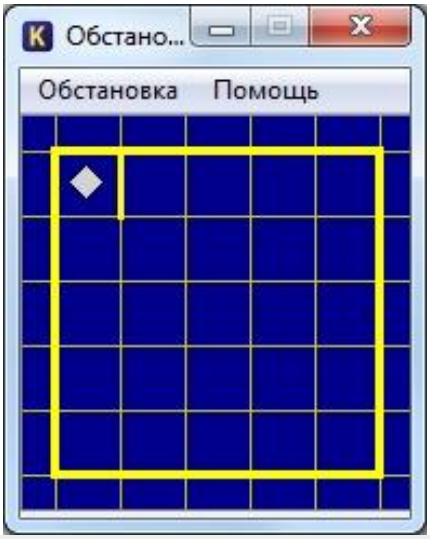
Что такое стартовая обстановка?

Наш Робот находится в некой среде — это клетчатое поле, размер которого известен. Так же на этом поле могут находится стены и закрашенные клетки, а сам

Робот может находится в любой клетке. Так вот — **стартовая обстановка** задает положение Робота на поле и расположение всех остальных элементов — стен, закрашенных клеток. И перед тем, как писать алгоритм для Робота необходимо задать стартовую обстановку. Насколько это важно давайте рассмотрим на примере. Пусть есть две стартовые обстановки:



Стартовая обстановка 1



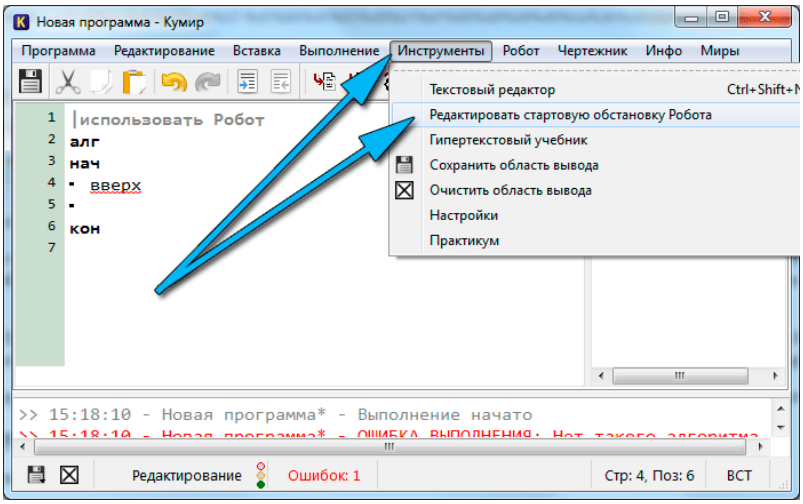
Стартовая обстановка 2

Отличаются они только тем, что в стартовой обстановке 2 справа от Робота находится стена.

Если наша программа начнется с команды, которая переместит Робота на одну клетку вправо (о простых командах Робота), то в первом случае (стартовая обстановка 1) Робот выполнит эту команду, а во втором программа завершится аварийно, так как Робот не может ходить сквозь стены. Получается, что одна и та же программа в первом случае работает, а во втором приводит к ошибке. Именно поэтому так важно задавать стартовую обстановку для Робота.

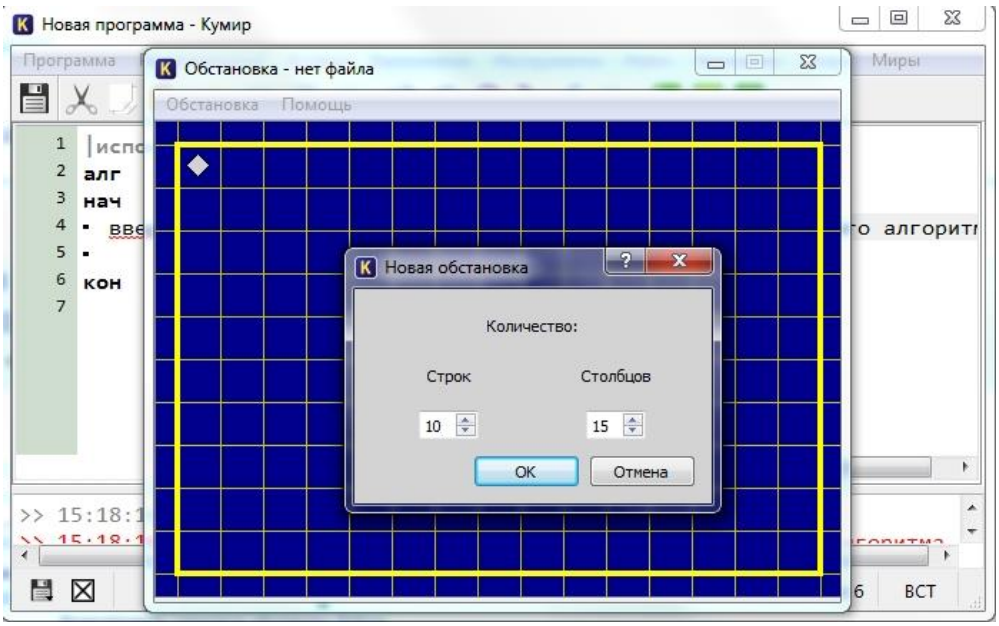
Как задать стартовую обстановку?

Запустив среду Кумир в меню***Инструменты***выбираем пункт***Редактировать*** стартовую обстановку Робота



Откроется окно с синим фоном. Это и есть **стартовая обстановка Робота**. И мы ее можем изменить.

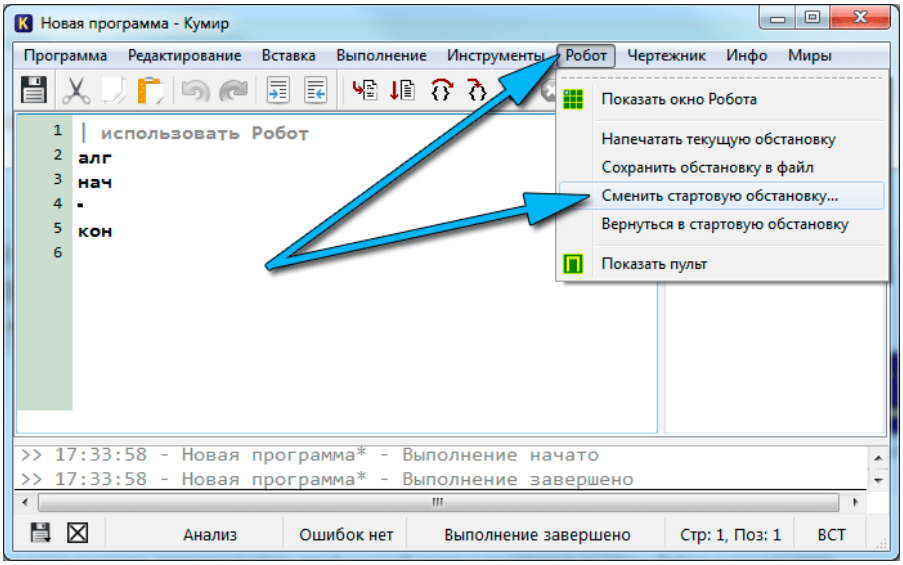
По-умолчанию, размер окна 10 на 15 клеток. Если нам необходимо изменить количество строк и столбцов, то щелкаем ***Обстановка*** -> ***Новая обстановка*** и задаем необходимые значения



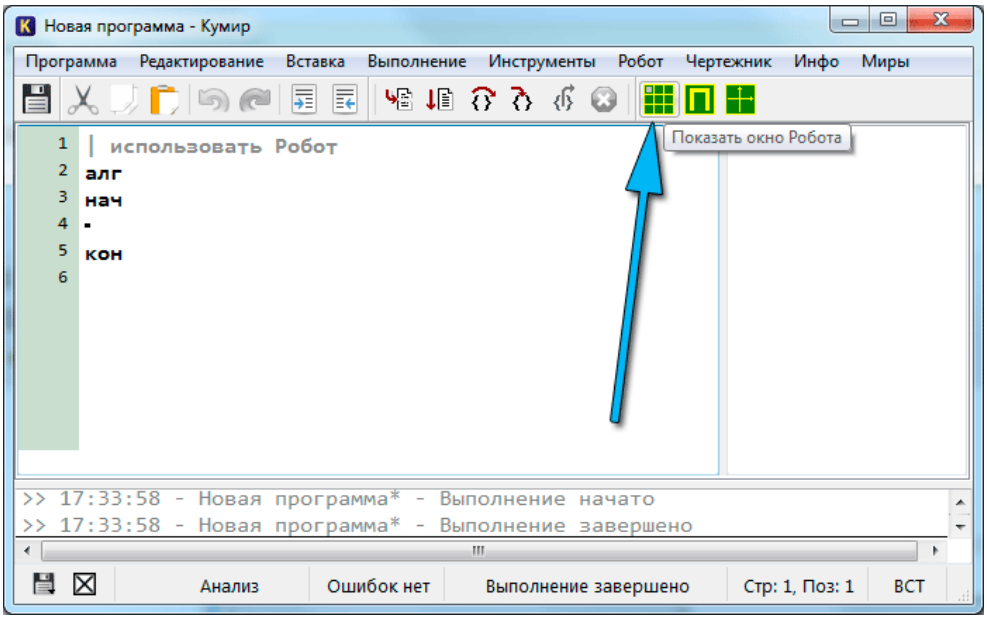
Далее,

- чтобы переместить Робота в новую позицию, щелкаем по нему левой кнопкой мыши и не отпуская ее тащим Робота в нужное место.
- чтобы добавить/удалить стену, щелкаем левой кнопкой мыши по границе клетки.
- чтобы закрасить/очистить клетку, щелкаем по ней левой кнопкой мыши
- чтобы добавить или убрать точку в клетку щелкаем по клетке, удерживая клавишу ***Ctrl***

После того, как мы задали нужную стартовую обстановку, ее необходимо сохранить (***Обстановка*** -> ***Сохранить*** или ***Обстановка*** -> ***Сохранить как***). После этого закрываем окно **Обстановка** и в основном окне программы выбираем ***Робот*** -> ***Сменить стартовую обстановку***



Находим сохраненную ранее обстановку и загружаем ее. После этого убедимся, что загрузили правильную стартовую обстановку, щелкнув по кнопке ***Показать окно Робота***



Если в окне с зеленым фоном (текущая обстановка Робота) вы увидите вашу обстановку, то можно переходить к написанию алгоритма, используя простые команды Робота.

Исполнитель Робот. Простые команды.

У нашего Робота тоже есть система команд. Сегодня мы рассмотрим простые команды Робота. Всего их 5:

вверх

вниз

влево

вправо

закрасить

Результат выполнения этих команд понятен из их названия:

вверх — переместить Робота на одну клетку вверх

вниз — переместить Робота на одну клетку вниз

влево — переместить Робота на одну клетку влево

вправо — переместить Робота на одну клетку вправо

закрасить — закрасить текущую клетку (клетку в которой находится Робот).

Эти команды можно писать с клавиатуры, а можно использовать горячие клавиши (нажав их команды будут вставляться автоматически):

вверх — Escape, Up (стрелка вверх)

вниз — Escape, Down (стрелка вниз)

влево — Escape, Left (стрелка влево)

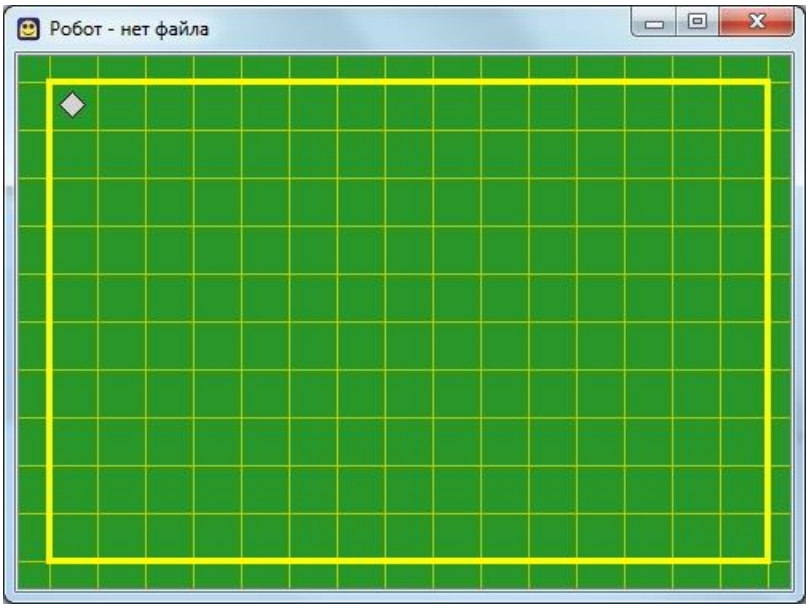
вправо — Escape, Right (стрелка вправо)

закрасить — Escape, Space (пробел)

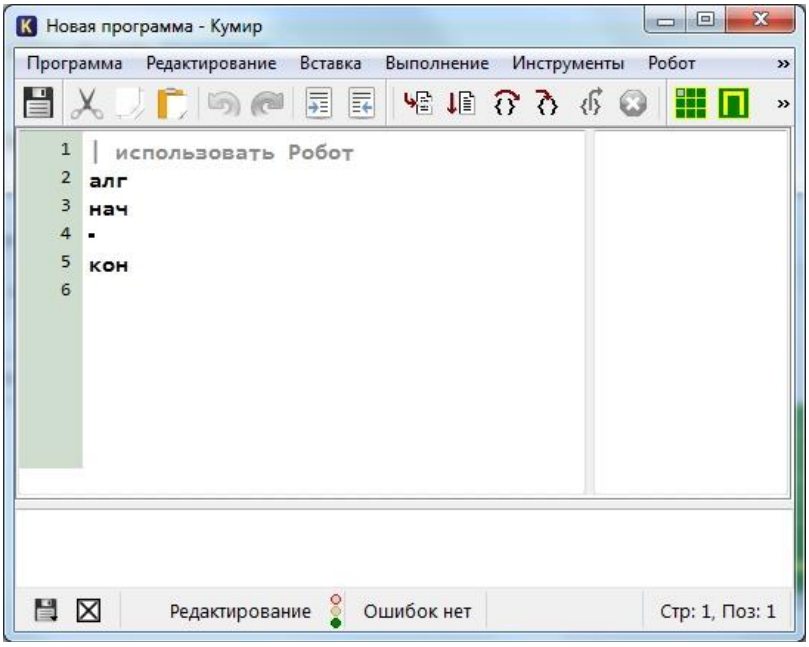
Обратите внимание, что набирать нужную комбинацию горячих клавиш нужно не привычным нам способом! Мы привыкли нажимать клавиши одновременно, а здесь их нужно нажимать последовательно.

Теперь мы готовы написать первый алгоритм для Робота. Предлагаю начать с простого — нарисуем квадрат со стороной 3 клетки. Поехали!

Запускаем Кумир, настраиваем его. Можно начинать писать программу? Конечно нет! Мы же не задали стартовую обстановку! Делаем это. Предлагаю использовать вот такую:



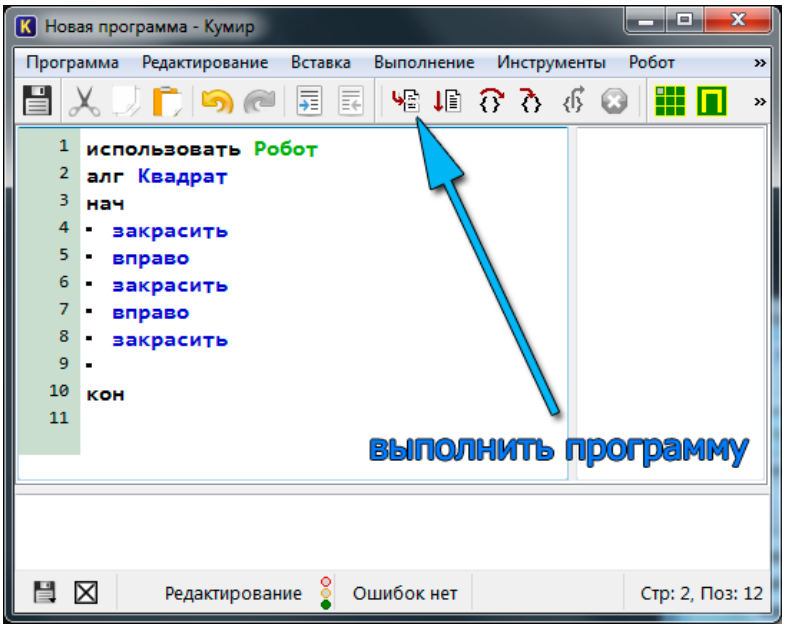
Вот теперь все готово. Начинаем писать программу. Пока она выглядит так



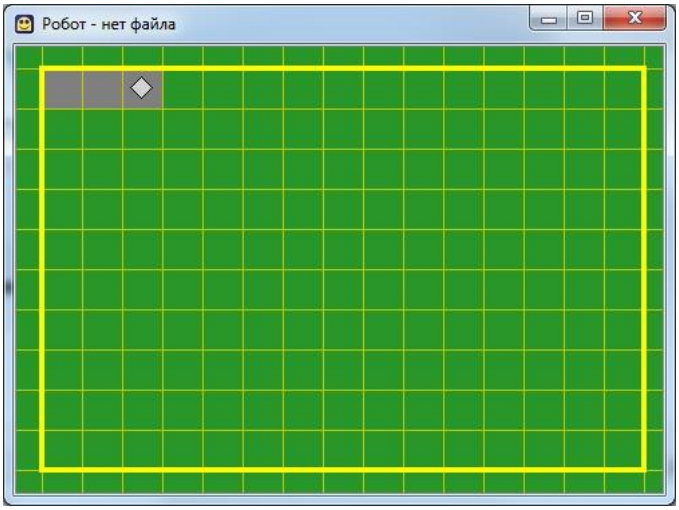
Удаляем символ «|» и называем наш алгоритм «Квадрат»

Предлагаю рисовать квадрат, двигаясь по часовой стрелке. Для начала закрасим текущую клетку, дав команду**закрасить**. Потом делаем шаг вправо и опять закрашиваем клетку. И еще раз шаг вправо и закрасить.

Попробуем запустить программу и посмотреть что же получилось. Для запуска нажимаем **F9** или же кнопку на панели инструментов



В результате мы должны увидеть вот такую картину



Если такое окно Робота у вас не появилось, то на панели инструментов щелкните «**Показать окно Робота**» или в меню Робот выберите пункт "**Показать окно Робота"**. Продолжаем дальше.

Теперь мы будем двигаться вниз и закрашивать правую сторону квадрата:

вниз

закрасить

вниз

закрасить

Потом пойдем влево, закрашивая нижнюю границу квадрата

влево

закрасить

влево

закрасить

У нас осталась одна незакрашенная клетка. Закрасим ее

вверх

закрасить

Все готово! В итоге наша программа выглядит так:

использовать Робот

алг Квадрат

нач

закрасить

вправо

закрасить

вправо

закрасить

вниз

закрасить

вниз

закрасить

влево

закрасить

влево

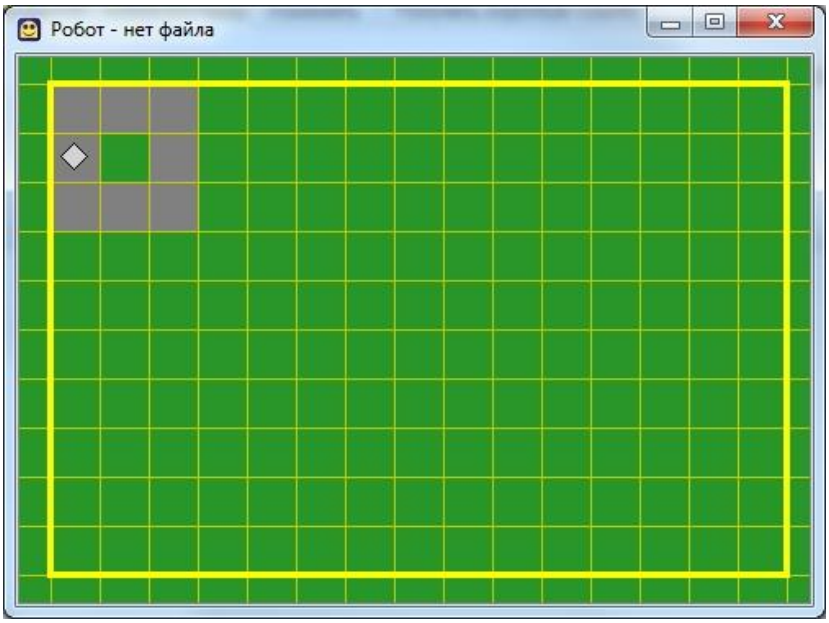
закрасить

вверх

закрасить

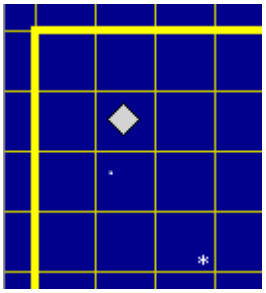
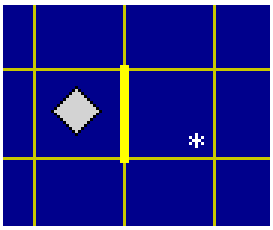
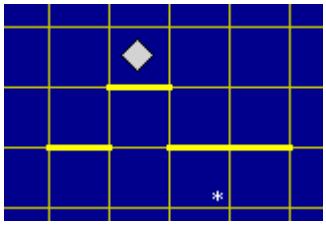
кон

А результат ее работы вот так

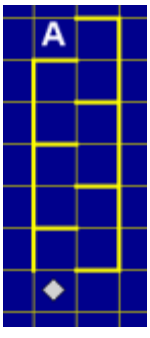


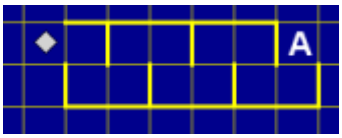
Задание 1

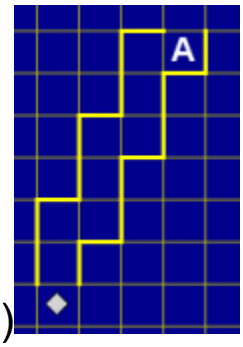
1. Используя команды исполнителя, напишите алгоритм для перевода Робота в точку отмеченную *.

A)  B)  C) 

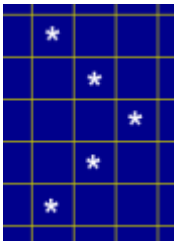
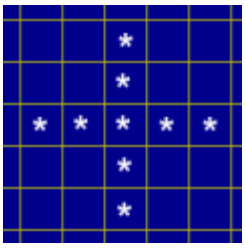
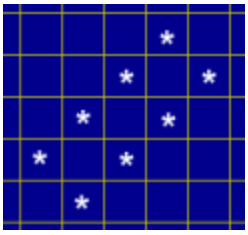
Задание 2. Необходимо провести Робота по лабиринту из начальногоположения (◊) в точку А.

A) 

в) 

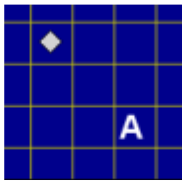
с) 

Задание 3. Составьте программу закрашивания клеток поля, отмеченных звездочкой. Начальное положение Робота где-то в центре поля.

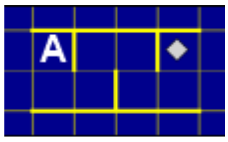
A)  в)  с) 

Задание 4. Составьте программу закрашивания прямоугольника 3×4, считая, что Робот находится где-то в центре поля.

Задание 5. Необходимо перевести Робота из начального положения (.) в точку А за минимальное число шагов любым из возможных способов.

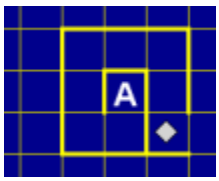


Задание 6. Необходимо перевести Робота по лабиринту из начального положения (.) в точку А.



Задание 7. Необходимо перевести Робота из начального положения (◊) в точку А за минимальное число шагов любым из возможных способов.

Задание 8.Необходимо перевести Робота по лабиринту из начального положения (◊) в точку А.



Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки:

1 Среда исполнителя что это?

2 Что такое система команд исполнителя?

3 Что такое цикл?

Практическое занятие №22 Кумир Робот. Циклические и условные алгоритмы

Цель занятия: изучить циклический и условный алгоритм на примере Кумир Робот

*Исходные данные:*теоретический материал

Содержание и порядок выполнения задания

Изучить теоретическую часть

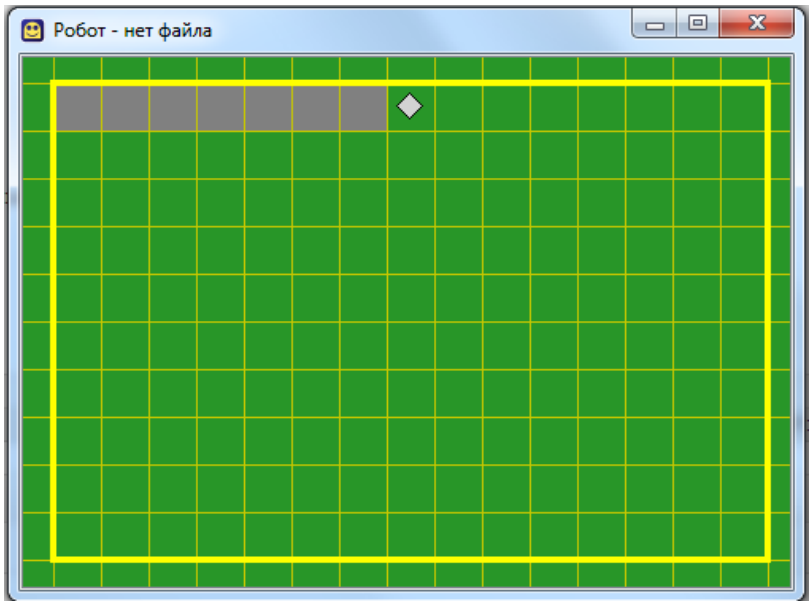
Выполнить задания

Теоретическая часть

У исполнителя Робот тоже есть возможность записывать циклы. Причем, **циклы бывают разные**: **цикл со счетчиком** и **цикл с параметром**.

Давайте рассмотрим это на примере.

Закрасим 7 клеток, как на рисунке. Рекомендую почитать про стартовую обстановку Робота и про его простые команды.



Изначально Робот находился в левой верхней клетке.

Давайте для начала решим задачу линейно. В этом случае мы будет закрашивать текущую клетку и перемещаться на 1 клетку вправо и программа будет выглядеть так:

использовать Робот

алг

нач

закрасить

вправо

закрасить

вправо

закрасить

вправо

закрасить

вправо

закрасить

вправо

закрасить

вправо

кон

Как видим, команды закрасить и вправо повторяются 7 раз. Давайте теперь перепишем программу с использованием цикла. Кстати, чтобы вставить цикл в свою программу можно в меню **Вставка** выбрать пункт **нц-раз-кц** или нажать одну из комбинаций клавиш **Esc, Р** (русская буква Р) или **Esc, Н** (латинская буква Н). Причем клавиши надо нажимать последовательно — сначала Esc, отпустить ее и только потом Р или Н.

Так вот, наша **программа с циклом** будет выглядеть так:

использовать Робот

алг

нач

нц 7 раз

закрасить

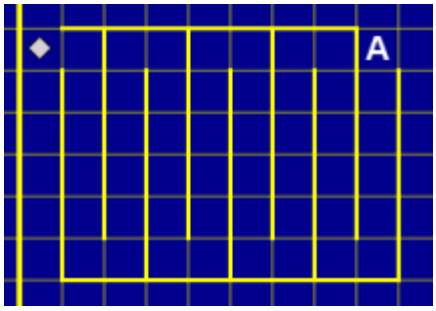
вправо

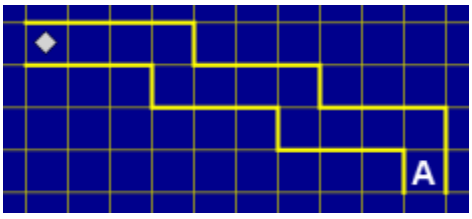
кц

кон

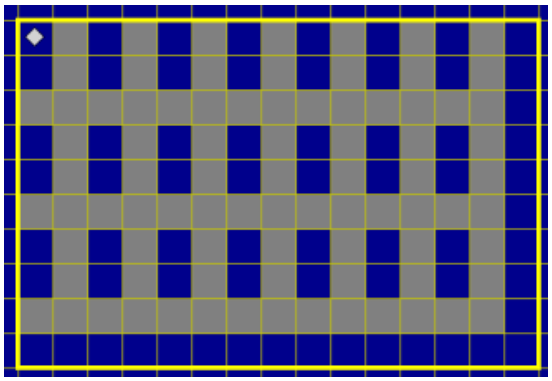
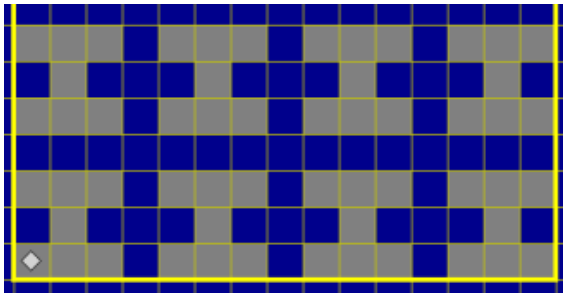
Если мы ее запустим, то увидим, что в результате получится тоже самое — 7 закрашенных клеток

Задание №1: Необходимо провести Робота по лабиринту из начального положения (◊) в точку А.

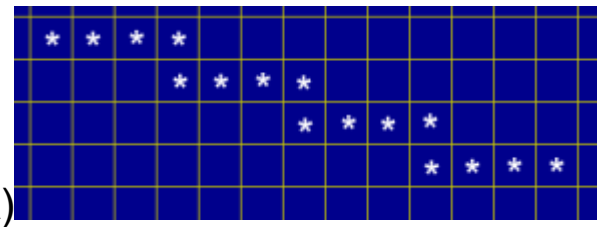
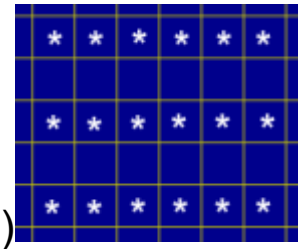
А) 

в) 

Задание №2: Составьте программу рисования узора. Начальное положение Робота отмечено символом ◊.

А)  В) 

Задание №3: Составьте программу закрашивания клеток поля, отмеченных звездочкой.

А)  в) 

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки

1. Какие циклы бывают в исполнителе Робот?
2. Алгоритмическая конструкция, отображающая естественный, последовательный порядок действий, называется…
3. В результате выполнения программы

```
использовать Робот
алг
нач
. закрасить
. вверх
. закрасить
. влево
```

```
. закрасить
. вниз
. закрасить
. вправо
кон
```

Робот закрасит

Практическое занятие №23 Кумир Робот Составление алгоритмов

*Цель занятия***.** Познакомиться с возможностью использования вспомогательногоалгоритма для Робота, оптимизировать решение с помощью вспомогательного алгоритма.

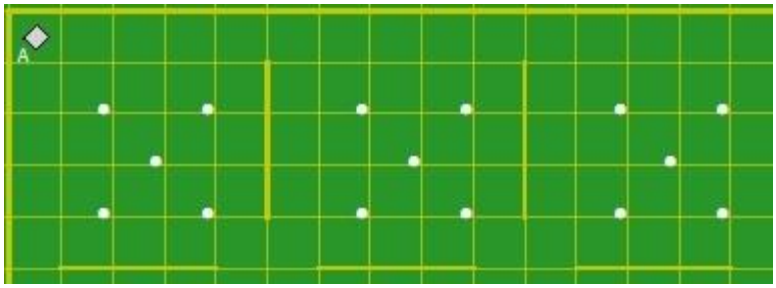
Содержание и порядок выполнения работы:

Изучите теоретическую часть

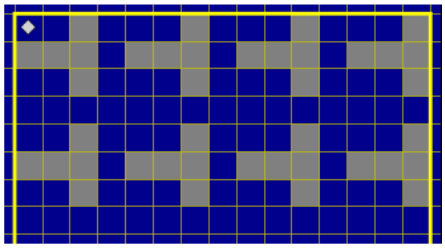
Выполните задания

Теоретическая часть

Задание 1. Закрасьте помеченные клетки. Используйте вспомогательный алгоритм для закраски одного квадрата.

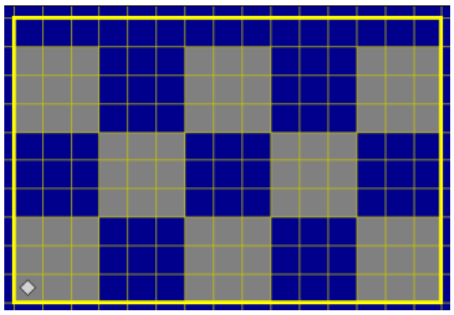


Задание 2. Закрасить указанные клетки с использованием вспомогательного алгоритма для закрашивания колонки, состоящей из 5 клеток. Начальное положение Робота отмечено символом ◊.



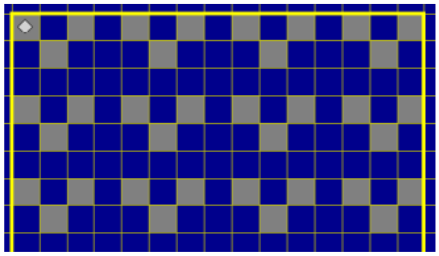
Задание 3.

Закрасить указанные клетки с использованием вспомогательного алгоритма. Начальное положение Робота отмечено символом ◊.



Задание 4.

Закрасить указанные клетки с использованием вспомогательного алгоритма. Начальное положение Робота отмечено символом ◊



Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки

1. Как называется алгоритмическая конструкция, в которой в зависимости от условий может выполняться либо одно, либо другое действие?
2. Дайте определение понятию «вспомогательный алгоритм»
3. Петя составил алгоритм, а Коля стер в нем одну команду:

алг прогулка

дано їна поле Робота стен нет

надо ї Робот погулял и вернулся в исходное положение

нач

| вверх

| вправо

| ???

| вниз

| влево

| влево

кон

Какую команду стер Коля?

Практическое занятие №24 Введение в язык программирования Python

Цель занятия:

1. Познакомиться со средой разработки Python.

2. Изучить основные типы данных, команды ввода и вывода данных.

Исходные материалы: теоретический материал, компьютер, программа Python

Содержание и порядок выполнения:

1. Изучить теоретическую часть;

2. Выполнить задания.

Теоретическая часть

Python– это объектно-ориентированный, интерпретируемый, переносимый язык сверхвысокого уровня. Программирование на Python позволяет получать быстро и качественно необходимые программные модули.

Python (в русском языке распространено название питóн) — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций. Язык обладает чётким и последовательным синтаксисом, продуманной модульностью и масштабируемостью, благодаря чему исходный код написанных на Python программ легко читаем.

Python — активно развивающийся язык программирования, новые версии с добавлением и изменением языковых свойств выходят примерно раз в два с половиной года. Он находит применение во множестве сфер человеческой деятельности.

Python – не самый «молодой» язык программирования, но и не слишком старый. К моменту его создания уже существовали такие языки как «Паскаль» или

«Си». А потому при создании «питона» авторы старались взять лучшее из различных платформ для разработчиков. Фактически Python представляет собой своеобразный «джем» удачных решений более чем из 8 различных языков.

В комплекте вместе с интерпретатором Python идет IDLE (интегрированная среда разработки). По своей сути она подобна интерпретатору, запущенному в интерактивном режиме с расширенным набором возможностей (подсветка синтаксиса, просмотр объектов, отладка и т.п.).

Для запуска IDLE в Windows необходимо перейти в папку Python в меню “Пуск” и найти там ярлык с именем “IDLE (Python 3.X XX-bit)”.

После установки программы запустите интерактивную графическую среду IDLE и дождитесь появления приглашения для ввода команд:

Type "copyright", "credits" or "license ()" for more information.

>>>

Три знака «больше» (>>>) называются *приглашением*. После приглашения можно вводить различные команды.

В самом начале обучения Python можно представить как обычный интерактивный калькулятор. В интерактивном режиме IDLE найдем значения следующих математических выражений. После завершения набора выражения нажмите клавишу**Enter** для завершения ввода и вывода результата на экран.

```
>>>3.0+6
9.0
>>>4+9
13
>>>1-5
- 4
>>> _+6
2
>>>
```

Приглашение возникнет снова. Это значит, что оболочка Python готова к выполнению дальнейших команд.

Нижним подчеркиванием в предыдущем примере обозначается последний полученный результат.

Любая Python-программа состоит из последовательности допустимых символов, записанных в определенном порядке и по определенным правилам.

Программа включает в себя:

комментарии;

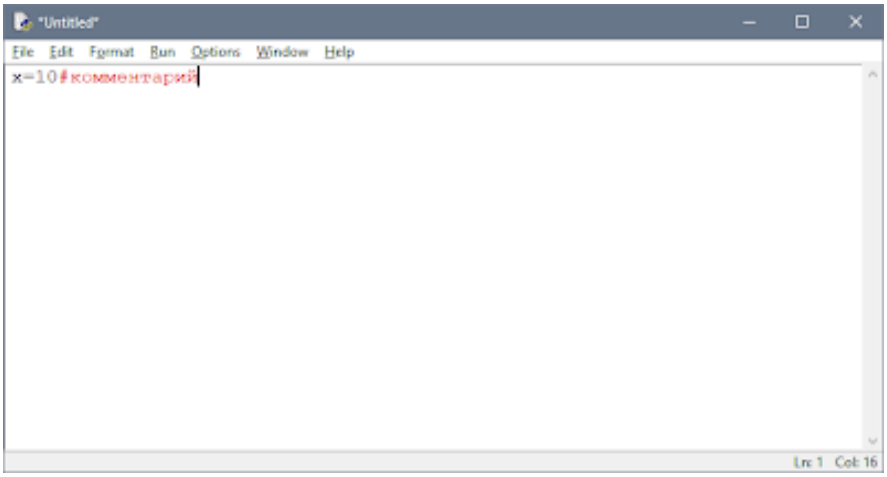
команды;

знаки пунктуации;

идентификаторы;

ключевые слова.

Комментарии в Python обозначаются предваряющим их символом # и продолжаются до конца строки (т.е. в Python все комментарии являются однострочными), при этом не допускается использование перед символом # кавычек:



Знаки пунктуации

В алфавит Python входит достаточное количество знаков пунктуации, которые используются для различных целей. Например, знаки "+" или "*" могут использоваться для сложения и умножения, а знак запятой "," - для разделения параметров функций.

Идентификаторы

Идентификаторы в Python - это имена используемые для обозначения переменной, функции, класса, модуля или другого объекта.

Ключевые слова

Некоторые слова имеют в Python специальное назначение и представляют собой управляющие конструкции языка.

Ключевые слова в Python:

['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']

Типы данных

None (неопределенное значение переменной)

Логические переменные (Boolean Type)

Числа (Numeric Type)

int – целое число

float – число с плавающей точкой

complex – комплексное число

Списки (Sequence Type)

list – список

tuple – кортеж

range – диапазон

Строки (Text Sequence *Type*)

str

Ввод и вывод данных

Ввод данных осуществляется при помощи команды **input(**список ввода**)**:

```
a = input()
print(a)
```

В скобках функции можно указать сообщение - комментарий к вводимым данным:

```
a = input ("Введите количество: ")
```

Команда input() по умолчанию воспринимает входные данные как строку символов. Поэтому, чтобы ввести целочисленное значение, следует указать тип данных int():

```
a = int (input())
```

Для ввода вещественных чисел применяется команда

```
a=float(input())
```

Вывод данных осуществляется при помощи команды **print**(список вывода):

```
a = 1
b = 2
print(a)
print(a + b)
print("сумма = ", a + b)
```

Существует возможность записи команд в одну строку, разделяя их через **;**. Однако не следует часто использовать такой способ, это снижает удобочитаемость:

```
a = 1; b = 2; print(a)
print (a + b)
print ("сумма = ", a + b)
```

Для команды **print** может задаваться так называемый сепаратор — разделитель между элементами вывода:

```
x=2
y=5
print ( x, "+", y, "=", x+y, sep = " " )
```

Результат отобразится с пробелами между элементами: 2 + 5 = 7

Простые арифметические операции над числами

x + y	Сложение
x - y	Вычитание
x * y	Умножение
x / y	Деление
//	Деление с округлением вниз
**	Возведение в степень
%	Остаток от деления
abs(x)	Модуль числа

```
>>>5/3
1.66666666666666667
>>> 5//3
1
>>>5%3
2
>>>5**2
25
>>>
```

Обратите внимание, что при записи числа 1.66666666666666667 используется не запятая, а точка.

Если один из операторов является вещественным числом, то в результате получится вещественное число

```
Python 3.4.1: example_prost_math.py - F:/Лабораторные Python/example_prost_math.py
File Edit Format Run Options Windows Help
#простейшие математические операции
x=5
y=6
print('x = ',x)
print('y = ',y)
z=x+y
print('z = ',z)
z=x-y
print('z = ',z)
z=x*y
print('z = ',z)
z=x/y
print('z = ',z)
```

Пример программы на Python

```
Python 3.4.1 Shell
File Edit Shell Debug Options Windows Help
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
x =  5
y =  6
z =  11
z =  -1
z =  30
z =  0.8333333333333334
>>>
```

Результат выполнения программы с применением простых арифметических операций

Порядок выполнения операций

Операции — это любые действия, которые совершаются с помощью операторов. Математические операции выполняются по очереди в зависимости от их приоритета (если не задать другую очередность с помощью скобок).

Умножение и деление имеют более высокий приоритет, чем сложение и вычитание, и это значит, что они будут выполняться первыми. Иначе говоря, при вычислении математического выражения Python сначала умножит и разделит числа, а затем перейдет к сложению и вычитанию.

Например, 5 + 30 * 20

в этом выражении сперва будут перемножены числа 30 и 20, а затем к их произведению будет прибавлено число 5.

>>> 5 + 30 * 20

605

По сути, это выражение означает «умножить 30 на 20 и прибавить к результату 5». Получается 605. Однако мы можем изменить порядок операций, заключив первые два числа в скобки. Вот так:

```
>>> (5 + 30) * 20
700
```

В результате получилось 700, а не 605, поскольку Python выполняет операции в скобках прежде, чем операции вне скобок. Другими словами, это выражение означает «прибавить 5 к 30 и умножить результат на 20».

Скобки могут быть *вложенными*, то есть внутри скобок могут стоять еще одни скобки:

```
>>> ((5 + 30) * 20) / 10
70.0
```

В этом примере Python сперва вычислит выражение во внутренних скобках, затем во внешних и в самом конце выполнит стоящую за скобками операцию деления.

Иначе говоря, это выражение означает «прибавить 5 к 30, затем умножить результат на 20, потом разделить результат на 10». Вот что при этом происходит:

сложение 5 и 30 дает 35;

умножение 35 на 20 дает 700;

деление 700 на 10 дает окончательный результат — 70.

Если бы мы не использовали скобки, результат вышел бы другим:

```
>>> 5 + 30 * 20 / 10
65.0
```

В этом случае сперва 30 умножается на 20 (получается 600), затем 600 делится на 10 (выходит 60) и, наконец, к 60 прибавляется 5, что дает в итоге 65.

! *Запомните, что умножение и деление всегда выполняются прежде, чем сложение и вычитание, если не менять порядок вычислений с помощью скобок.*

Для форматированного вывода используется **format**:

Строковый метод format() возвращает отформатированную версию строки, заменяя идентификаторы в фигурных скобках {}. Идентификаторы могут быть позиционными, числовыми индексами, ключами словарей, именами переменных.

Синтаксис команды **format**:

поле замены := "{" [имя поля] ["!" преобразование] [":" спецификация] "}"

имя поля := arg_name ("." имя атрибута | "[" индекс "]")*

преобразование := "r" (внутреннее представление) | "s" (человеческое представление)

спецификация := см. ниже

Аргументов в format() может быть больше, чем идентификаторов в строке. В таком случае оставшиеся игнорируются.

Идентификаторы могут быть либо индексами аргументов, либо ключами:

example_format.py - K:\Лабораторные Python\example_format.py (3.7.1)

File Edit Format Run Options Window Help

```
x=11
print(x)#вывод без форматирования
print ("{:4}".format(x))#перед значением перемнной x будет присутствовать 2 пробела,
                         #так как число 11 занимает 2 знакоместа
```

Python 3.7.1 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:05:16) [MSC v.1915 32 bit (Inte
l)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============= RESTART: K:\Лабораторные Python\example_format.py =============
11
  11
>>> |
```

В результате выведется число 11, а перед ним два пробела, так как указано использовать для вывода четыре знакоместа.

Или с несколькими аргументами:

example_format1.py - K:\Лабораторные Python\example_format1.py (3.7.1)

File Edit Format Run Options Window Help

```
x=2
print ("{:4d}{:4d}{:4d}".format (x,x,x))
```

Python 3.7.1 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:05:16) [MSC v.1915 32 bit (Inte
l)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============= RESTART: K:\Лабораторные Python\example_format1.py =============
   2   2   2
>>>
```

В итоге каждое из значений выводится из расчета 4 знакоместа.

Спецификация формата:

спецификация	:= [[fill]align][sign][#][0][width][,][.precision][type]
заполнитель	:= символ кроме '{' или '}'
выравнивание	:= "<" | ">" | "=" | "^"
знак	:= "+" | "-" | " "
ширина	:= integer
точность	:= integer
тип	:= "b" | "c" | "d" | "e" | "E" | "f" | "F" | "g" | "G" | "n" | "o" | "s" | "x" | "X" | "%"

Тип	Значение
'd', 'i', 'u'	Десятичное число.
'o'	Число в восьмеричной системе счисления.
'x'	Число в шестнадцатеричной системе счисления (буквы в нижнем регистре).
'X'	Число в шестнадцатеричной системе счисления (буквы в верхнем регистре).
'e'	Число с плавающей точкой с экспонентой (экспонента в нижнем регистре).
'E'	Число с плавающей точкой с экспонентой (экспонента в верхнем регистре).
'f', 'F'	Число с плавающей точкой (обычный формат).
'g'	Число с плавающей точкой. с экспонентой (экспонента в нижнем регистре), если она меньше, чем -4 или точности, иначе обычный формат.
'G'	Число с плавающей точкой. с экспонентой (экспонента в верхнем регистре), если она меньше, чем -4 или точности, иначе обычный формат.
'c'	Символ (строка из одного символа или число - код символа).
's'	Строка.
'%'	Число умножается на 100, отображается число с плавающей точкой, а за ним знак %.

Для форматирования вещественных чисел с плавающей точкой используется следующая команда:

print('{0:.2f}'.format(вещественное число))

format_chisla.py - K:/Лабораторные Python/format_chisla.py (3.7.1)

File Edit Format Run Options Window Help

```
x=10
y=7
print("{0:.2f}".format(x/y))
```

В результате выведется число с двумя знаками после запятой.

Python 3.7.1 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct
1)] on win32
Type "help", "copyright", "credits"
>>>
============== RESTART: K:/Лабораторн
1.43
>>>
```

Пример

Напишите программу, которая запрашивала бы у пользователя:

Вариант 0

- ФИО ("Ваши фамилия, имя, отчество?")

- возраст ("Сколько Вам лет?")

- место жительства ("Где вы живете?")

После этого выводила бы три строки:

"Ваше имя"

"Ваш возраст"

"Вы живете в"

Решение

```
a=input('Введите ваши фамилию, имя, отчество ')
b=input('Сколько вам лет? ')
c=input('Где вы живёте? ')
print('Ваше имя ',a)
print('Ваш возраст ',b)
print('Вы живете в ',c)
```

```
Введите ваши фамилию, имя, отчество Иванов Иван Иванович
Сколько вам лет? 15
Где вы живёте? Уссурийск
Ваше имя  Иванов Иван Иванович
Ваш возраст  15
Вы живете в  Уссурийск
```

Задания для самостоятельной работы (по вариантам)

Напишите программу, которая запрашивала бы у пользователя:

Вариант 1

Имя, Фамилия, Возраст, Место жительства

- фамилия, имя ("Ваши фамилия, имя?")
- возраст ("Сколько Вам лет?")
- место жительства ("Где вы живете?")

После этого выводила бы три строки:

"Ваши фамилия, имя"

"Ваш возраст"

"Вы живете в"

Вариант 2

Имя, Дата рождения, Образование

- имя ("Ваше, имя?")
- дата рождения ("Ваша дата рождения?")
- образование ("Где Вы учитесь?")

После этого выводила бы три строки:

"Ваше имя"

"Дата рождения"

"Вы учитесь в "

Вариант 3

Фамилия, Место жительства

- Фамилия("Ваша фамилия?")
- место жительства ("Где Вы живете?")

После этого выводила бы две строки:

"Ваша фамилия"

"Вы живете в"

Вариант 4

Фамилия, Место рождения, любимая музыка

- Фамилия, ("Ваша фамилия?")
- место рождения ("Где Вы родились?")
- музыка("Какая музыка нравится? ")

После этого выводила бы три строки:

"Ваши имя, фамилия"

"Вы родились в"

"Ваша любимая музыка "

Вариант 5

Имя, Фамилия, ФИО мамы, ФИО отца

- ФИО (например, "Ваши фамилия, имя, отчество?")

- возраст ("Сколько Вам лет?")

- место жительства ("Где Вы живете?")

После этого выводила бы три строки:

"Ваши имя, фамилия, отчество"

"Ваш возраст"

"Вы живете в"

Вариант 6

Имя, Любимый предмет в школе, Номер класса

- имя ("Ваше имя?")

- любимый предмет ("Какой Ваш любимый предмет в школе?")

- номер класса ("В каком классе Вы учитесь?")

После этого выводила бы три строки:

"Ваше имя"

"Ваш любимый предмет в школе"

"Вы учитесь в классе номер"

Вариант 8

Имя, Фамилия, Отчество, Хобби

- ФИО (например, "Ваши фамилия, имя, отчество?")

- хобби ("Чем Вы увлекаетесь?")

После этого выводила бы две строки:

"Ваши имя, фамилия, отчество"

"Ваше хобби"

Вариант 9

Имя, Фамилия, любимый спорт

- Фамилия, имя ("Ваши фамилия, имя?")

- образование ("В какой школе Вы учитесь?")

- ФИО Вашего руководителя по информатики ("ФИО Вашего руководителя по информатики?")

После этого выводила бы три строки:

"Ваши имя, фамилия"

"Вы учитесь в школе номер: "

"ФИО Вашего руководителя по информатике "

Вариант 10

Имя, Фамилия, Любимый предмет в школе (в институте), ФИО классного руководителя (куратора)

- Фамилия, имя ("Ваши фамилия, имя?")

- любимый предмет в школе ("Какой Ваш любимый предмет в школе?")

- ФИО классного руководителя ("ФИО Вашего классного руководителя?")

После этого выводила бы три строки:

"Ваши имя, фамилия"

"Ваш любимый предмет в школе "

"ФИО Вашего классного руководителя"

Вариант 11

Имя, Фамилия, Возраст, Дата рождения

- Фамилия, имя ("Ваши фамилия, имя?")

- возраст ("Сколько Вам лет?")

- дата рождения ("Когда Вы родились?")

После этого выводила бы три строки:

"Ваши имя, фамилия"

"Ваш возраст"

"Дата Вашего рождения"

Вариант 12

Имя, Фамилия, Место жительства, Месторождения

- Фамилия, имя ("Ваши фамилия, имя?")

- место рождения ("Где Вы родились?")

- место жительства ("Где Вы живете?")

После этого выводила бы три строки:

"Ваши имя, фамилия"

"Вы родились в"

"Вы живете в"

Вариант 13

Имя, Фамилия, Возраст, Номер телефона

- Фамилия, имя ("Ваши фамилия, имя?")

- возраст ("Сколько тебе лет?")

- номер телефона ("Номер Вашего телефона?")

После этого выводила бы три строки:

"Ваши имя, фамилия"

"Ваш возраст"

"Ваш номер телефона"

Вариант 14

Имя, Фамилия, Страна, Край, Город

- Фамилия, имя ("Ваши фамилия, имя?")

- страна ("В какой стране Вы живете?")

- город ("В каком городе Вы живете?")

После этого выводила бы три строки:

"Ваши имя, фамилия"

"Вы живете в стране"

"Вы живете в крае"

"Вы живете в городе"

Вариант 15

Имя, Фамилия, ФИО Вашего классного руководителя

- Фамилия, имя ("Ваши фамилия, имя?")

- ФИО Вашего классного руководителя ("ФИО Вашего классного руководителя?")

После этого выводила бы три строки:

"Ваши имя, фамилия"

"ФИО Вашего руководителя по информатике"

"ФИО Вашего классного руководителя"

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия;

2. Цель занятия;

3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»;

4. Список используемых источников;

5. Выводы и предложения;

6. Дата и подпись курсанта и преподавателя.

Вопросы для самопроверки

1.Как в языке Python называются указания компьютеру, определяющие, какие операции выполнит компьютер над данными?

2.Определите порядок выполнения операций в указанной инструкции?

```
a = 3 - 5 * 4 ** (-3 + 2)
```

3. Символ # в Python обозначает **…**

4.Операция 3**4 - это

5.345 - **…** тип данных.

6.Операция 46%10 – это …

7.Функция input() – предназначена для …

Практическое занятие №25 Математические операции в Python

Цель занятия:

1. Познакомиться с основными математическими операциями в Python

Исходные материалы: теоретический материал, программа Python

Содержание и порядок выполнения:

1. Изучить теоретическую часть;

2. Выполнить задания.

Теоретическая часть

Язык Python, благодаря наличию огромного количества библиотек для решения разного рода вычислительных задач, сегодня является конкурентом таким пакетам как Matlab и Octave. Запущенный в интерактивном режиме, он, фактически, превращается в мощный калькулятор. В этом практическом занятии речь пойдет об

арифметических операциях, доступных в данном языке Арифметические операции изучим применительно к числам.

Если в качестве операндов некоторого арифметического выражения используются только целые числа, то результат тоже будет целое число. Исключением является операция деления, результатом которой является вещественное число. При совместном использовании целочисленных и вещественных переменных, результат будет вещественным.

Целые числа (int)

Числа в Python 3 поддерживают набор самых обычных математических операций:

x + y	Сложение
x - y	Вычитание
x * y	Умножение
x / y	Деление
x // y	Получение целой части от деления
x % y	Остаток от деления
-x	Смена знака числа
abs(x)	Модуль числа
divmod(x, y)	Пара (x // y, x % y)
x ** y	Возведение в степень
pow(x, y[, z])	x : Число, которое требуется возвести в степень. y : Число, являющееся степенью, в которую нужно возвести первый аргумент. Если число отрицательное или одно из чисел "x" или "y" не целые, то аргумент "z" не принимается. z : Число, на которое требуется произвести деление по модулю. Если число указано, ожидается, что "x" и "y" положительны и имеют тип int.

Пример применения выше описанных операций над целыми числами

```
x = 5
y = 2
z = 3
x+y = 7
x-y = 3
x*y = 10
x/y = 2.5
x//y = 2
x%y = 1
-x= -5
abs(-x) = 5
divmod(x,y) = (2, 1)
x**y = 25
pow(x,y,z) = 1
```

Вещественные числа (float)

Вещественные числа поддерживают те же операции, что и целые. Однако (из-за представления чисел в компьютере) вещественные числа неточны, и это может привести к ошибкам.

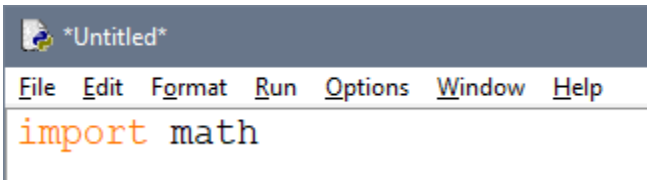
Пример применения вышеописанных операций над вещественными числами

```
x = 5.5
y = 2.3
x+y = 7.8
x-y = 3.2
x*y = 12.649999999999999
x/y = 2.3913043478260874
x//y = 2.0
x%y = 0.9000000000000004
-x= -5.5
abs(-x) = 5.5
divmod(x,y) = (2.0, 0.9000000000000004)
x**y = 50.44686540422945
```

Библиотека (модуль) math

В стандартную поставку Python входит библиотека math, в которой содержится большое количество часто используемых математических функций.

Для работы с данным модулем его предварительно нужно импортировать.



Рассмотрим наиболее часто используемые функции модуля math

math.ceil(x)	Возвращает ближайшее целое число большее, чем x
math.fabs(x)	Возвращает абсолютное значение числа x
math.factorial(x)	Вычисляет факториал x
math.floor(x)	Возвращает ближайшее целое число меньшее, чем x
math.exp(x)	Вычисляет e**x
math.log2(x)	Логарифм по основанию 2
math.log10(x)	Логарифм по основанию 10
math.log(x[, base])	По умолчанию вычисляет логарифм по основанию e, дополнительно можно указать основание логарифма
math.pow(x, y)	Вычисляет значение x в степени y
math.sqrt(x)	Корень квадратный от x

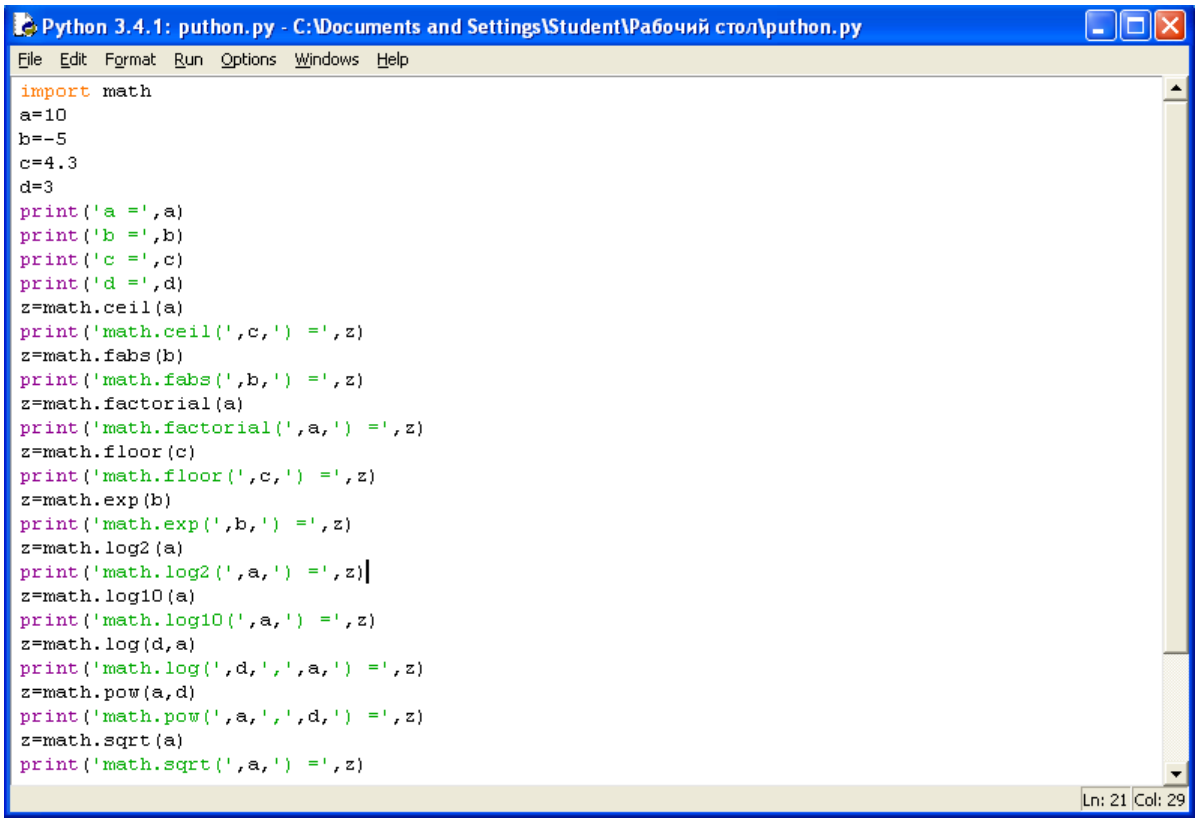
Пример применения вышеописанных функций над числами

В программе определены 4 переменные - a, b, c, d, каждая из которых является либо целым числом, либо вещественным, либо отрицательным.

Командой print() выводится значение каждой переменной на экран при выполнении программы.

В переменную z помещается результат выполнения функции модуля math.

Затем командой print() выводится сообщение в виде используемой функции и её аргумента и результат её выполнения.



Пример программы на Python

```
Python 3.4.1 Shell
File  Edit  Shell  Debug  Options  Windows  Help
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit
(Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
a = 10
b = -5
c = 4.3
d = 3
math.ceil( 4.3 ) = 10
math.fabs( -5 ) = 5.0
math.factorial( 10 ) = 3628800
math.floor( 4.3 ) = 4
math.exp( -5 ) = 0.006737946999085467
math.log2( 10 ) = 3.321928094887362
math.log10( 10 ) = 1.0
math.log( 3 , 10 ) = 0.47712125471966244
math.pow( 10 , 3 ) = 1000.0
math.sqrt( 10 ) = 3.1622776601683795
>>>
                                                                Ln: 19 Col: 4
```

Результат выполнения программы с применением функций модуля math

Тригонометрические функции модуля math

math.cos(x)	Возвращает cos числа X
math.sin(x)	Возвращает sin числа X
math.tan(x)	Возвращает tan числа X
math.acos(x)	Возвращает acos числа X
math.asin(x)	Возвращает asin числа X
math.atan(x)	Возвращает atan числа X

Пример применения вышеописанных функций над числами

В программе определена переменная x, содержащая целое число.

Значение переменной выводится командой print() на экран.

В переменную z помещается результат выполнения тригонометрической функции модуля math.

Затем командой print() выводится сообщение в виде используемой функции и её аргумента и результат её выполнения.

```
import math
x=1
print('x =',x)

z=math.cos(x)
print('math.cos(',x,') =',z)

z=math.sin(x)
print('math.sin(',x,') =',z)

z=math.tan(x)
print('math.tan(',x,') =',z)

z=math.acos(x)
print('math.acos(',x,') =',z)

z=math.asin(x)
print('math.asin(',x,') =',z)

z=math.atan(x)
print('math.atan(',x,') =',z)
```

Пример программы с использованием тригонометрических функций модуля math

```
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
x = 1
math.cos( 1 ) = 0.5403023058681398
math.sin( 1 ) = 0.8414709848078965
math.tan( 1 ) = 1.5574077246549023
math.acos( 1 ) = 0.0
math.asin( 1 ) = 1.5707963267948966
math.atan( 1 ) = 0.7853981633974483
>>>
```

Результат выполнения программы с применениемтригонометрических функций модуля math

Константы:

- **math.pi** - число Pi.
- **math.e** - число e (экспонента).

Пример

Напишите программу, которая бы вычисляла заданное арифметическое выражение при заданных переменных. Ввод переменных осуществляется с клавиатуры. Вывести результат с 2-мя знаками после запятой.

Вариант 0

$$Z = \frac{9\pi t + 10\cos(x)}{\sqrt{t} - |\sin(t)|} * e^{x}$$

x=10; t=1

Решение

Сначала импортируем модуль math. Для этого воспользуемся командой importmath.

Затем следует ввести значения двух переменных целого типа x и t.

Для ввода данных используется команда input, но так как в условии даны целые числа, то нужно сначала определить тип переменных: x=int(), t=int().

Определив тип переменных, следует их ввести, для этого в скобках команды int() нужно написать команду input().

Для переменной x это выглядит так: x=int(input("сообщение при вводе значения")).

Для переменной t аналогично: t=int(input("сообщение при вводе значения")).

Следующий шаг - это составление арифметического выражения, результат которого поместим в переменную z.

Сначала составим числитель. Выглядеть он будет так: 9*math.pi*t+10*math.cos(x).

Затем нужно составить знаменатель, при этом обратим внимание на то, что числитель делится на знаменатель, поэтому и числитель, и знаменатель нужно поместить в скобки (), а между ними написать знак деления /.

Выглядеть это будет так: (9*math.pi*t+10*math.cos(x))/(math.sqrt(t)-math.fabs(math.sin(t))).

Последним шагом является умножение дроби на экспоненту в степени x.

Так как умножается вся дробь, то следует составленное выражение поместить в скобки (), а уже потом написать функцию math.pow(math.e,x).

В результате выражение будет иметь вид:

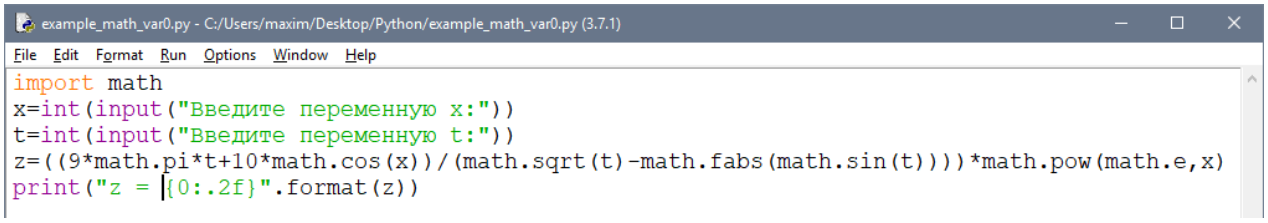
z=((9*math.pi*t+10*math.cos(x))/(math.sqrt(t)-math.fabs(math.sin(t))))*math.pow(math.e,x).

При составлении данного выражения следует обратить внимание на количество открывающихся и закрывающихся скобок.

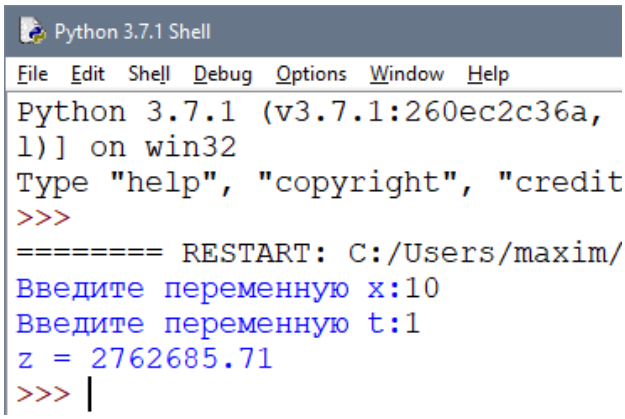
Командой print() выведем значение переменной, отформатировав его командой format.

Сам формат записывается в апострофах в фигурных скобках {}.

В задаче требуется вывести число с двумя знаками после запятой, значит вид формата будет выглядеть следующим образом: {0:.2f}, где 2 - это количество знаков после запятой, а f указывает на то, что форматируется вещественное число. При этом перед 2 нужно поставить точку, указав тем самым на то, что форматируем именно дробную часть числа.



Результат



ВАРИАНТЫ ЗАДАНИЙ

Вариант 1.

Вычислить. $a = \dfrac{\sqrt{|x-1|} - \sqrt[3]{|y|}}{1 + \dfrac{x^2}{2} + \dfrac{z^2}{4}}$, и $b = x(arctgZ + e)^{-(y+3)}$; если *x = 2; y = 6; z = 7.*

Вариант 2.

Вычислить $a = \dfrac{3 + e^{y-1}}{1 + x(y - tgZ)}$, и $b = 1 + |y - x| + \cdot \dfrac{(y-x)^2}{2} + \dfrac{|y-z|^3}{3}$;

если *x = 5; y = 1; z = 4.*

Вариант 3.

Вычислить $a = (1+y)\dfrac{x + y/(x^2+4)}{e^{-x-2} + 1/(z^2+4)}$, и $b = \dfrac{1 + \cos(y-2)}{x^{4/2} + \sin^2 z}$; если *x =1; y=12; z = 6.*

Вариант 4.

Вычислить $a = y + \dfrac{x}{z^2 + \left|\dfrac{x^2}{y + x^3/3}\right|}$, и $b = (1 + \arcsin\ x^2 + tg^2\dfrac{z}{2})$;

если *x = 11; y = 5; z = 10.*

Вариант 5.

Вычислить $a = \dfrac{2\cos(x - n/6)}{1/2 + \sin^2 y}$, и $b = 1 + \dfrac{z^2}{3 + z^2/5}$; если *x = 15; y = 7; z = 3.*

Вариант 6.

Вычислить $a = \dfrac{1 + \sin^2(x-y)}{2 + |x - 2x/(1 + x^2y^2)|} + z$, и $b = \cos^2(arctg\dfrac{1}{z})$; если

x = 3; y = 4; z = 5.

Вариант 7.

Вычислить $a = \ln\left|(y - \sqrt{|x|})(x - \dfrac{y}{z + x^2/4})\right|$, и $b = x - \dfrac{x^2}{3!} + \dfrac{x^5}{5}$;

если *x =5; y =3; z =17.*

Вариант 8.

Вычислить $a = \left|\dfrac{\sin^3|3x^3 + 5y^2 + 15z|}{\sqrt{(12x^3 + 6y^2 - z)^2 + 3.14}}\right|$, и $b = tg(7x^2 + e^{3y^2 - z})$; если

x = 7; y = 4; z = 8.

Вариант 9.

Вычислить $a=\sqrt{\dfrac{|\sin 8y|+17x}{(1-\sin 4\cdot y\cdot\cos(z^2+18))^2}}$, и $b=y-\sqrt{\dfrac{3|z|^2}{3+|tg3x^2-\sin 5y|}}$; если

x = 17; y = 16; z = 15.

Вариант 10.

Вычислить $a=l^{(2x^2-y)}+\dfrac{3\arccos y^2}{(z^2+xy)^2}$, и $b=tg\left(\dfrac{|x-y|^2}{3Z+\cos x^2}\right)$; если

x = 6; y = 7; z = 8.

Вариант 11.

Вычислить $a=10\cdot\ln\left|(x-\sqrt{\cos y})(z-\dfrac{y^2}{x-x^2/2})\right|$; и $b=2x^3+\dfrac{|y^2-3z^3|}{e^{3x+y^2}}$;

если *x = 3; y = 4; z = 5.*

Вариант 12.

Вычислить $a=\dfrac{\cos^2 x+5\sin^3 y}{\ln|2z+z^3|}$; и $b=\arcsin\left(\dfrac{4x^2+5y^3}{\sqrt{2z-y^2}}\right)/\ln(\sqrt{7x-7x/5y^2})$;

если *x = 4; y = 5; z = 6.*

Вариант 13.

Вычислить $a=tg\left((x+y)^2-\sqrt{\dfrac{\cos^2(z)}{tgz^2}}\right)$, и $b=\ln(x^2+\dfrac{e^{-3y^2}}{\sin^2 y})$; если

x = 5; y = 6; z = 7.

Вариант 14.

Вычислить $a=\left(\dfrac{y}{\sin x}+\dfrac{y}{(\sin^2 x-3\cos z)}\right)\cdot e^{5x^2}$, и $b=\dfrac{5-e^{z-2}}{y+x^2|z^2-tgz|}$;

если *x = 8; y = 9; z = 10.*

Вариант 15.

Вычислить $a = z + \left|y^2 - x^2\right| + y(\arcsin(z - e^{(z+5)}));$

и $b = \sqrt[3]{\dfrac{x - \cos^2(y+z)}{5 + \ln(y - 5x/\left| xy - \sqrt{7}\right|)}}$; если *x = 9; y = 10; z = 11*

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия;
2. Цель занятия;
3. Вариант задания;
4. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»;
5. Список используемых источников;
6. Выводы и предложения;
7. Дата и подпись курсанта и преподавателя;

Вопросы для самопроверки

1 Перечислите математические операции для целых чисел.

2 Перечислите часто используемые функции модуля math.

3 Перечислите тригонометрические функции модуля math.

Практическое занятие №26 Структура ветвление в Python

Цель занятия:

1. Познакомиться со структурой ветвление (if, if-else, if-elif-else);

2. Научиться работать с числами и строками используя структуру ветвление.

Исходные данные: теоретический материал, программа Python

Содержание и порядок выполнения:

1. Изучить теоретическую часть;

2. Выполнить задания.

Теоретическая часть

Условный оператор ветвления if, if-else, if-elif-else

Оператор ветвления if позволяет выполнить определенный набор инструкций в зависимости от некоторого условия. Возможны следующие варианты использования.

1. Конструкция if

Синтаксис оператора if выглядит так:

if логическое выражение:

команда_1

команда_2

...

команда_n

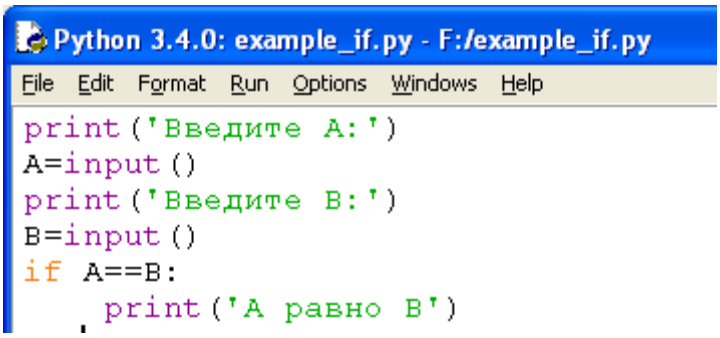
После оператора if записывается логическое выражение.

Логическое выражение — конструкция языка программирования, результатом вычисления которой является «истина» или «ложь».

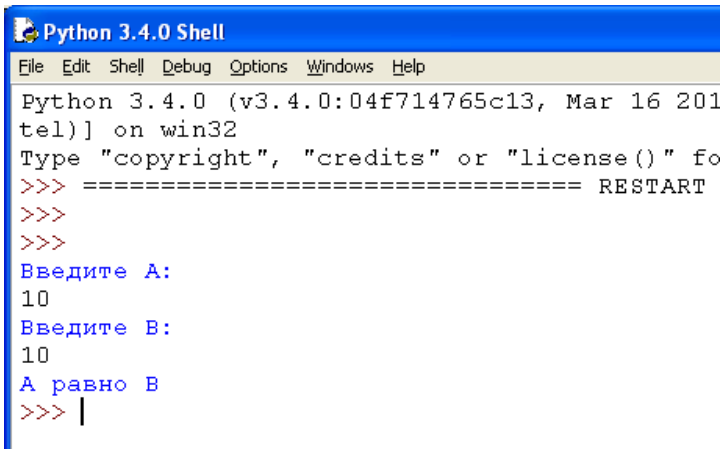
Если это выражение истинно, то выполняются инструкции, определяемые данным оператором. Выражение является истинным, если его результатом является число не равное нулю, непустой объект, либо логическое True. После выражения нужно поставить двоеточие “:”.

ВАЖНО: блок кода, который необходимо выполнить, в случае истинности выражения, отделяется **четырьмя** пробелами слева!

Программа запрашивает у пользователя два числа, затем сравнивает их и если числа равны, то есть логическое выражение A==B истинно, то выводится соответствующее сообщение.



Пример программы на Python



Результат выполнения программы с использованием условного оператора if

2. Конструкция if – else

Бывают случаи, когда необходимо предусмотреть альтернативный вариант выполнения программы. Т.е. при истинном условии нужно выполнить один набор инструкций, при ложном – другой. Для этого используется конструкция if – else.

Синтаксис оператора if – else выглядит так:

if логическое выражение:

команда_1

команда_2

...

команда_n

else:

команда_1

команда_2

...

команда_n

Программа запрашивает у пользователя два числа, затем сравнивает их и если числа равны, то есть логическое выражение A==B истинно, то выводится соответствующее сообщение. В противном случае выводится сообщение, что числа не равны.

```
Python 3.4.0: example_if.py - F:/example_if.py
File Edit Format Run Options Windows Help
print('Введите A:')
A=input()
print('Введите B:')
B=input()
if A==B:
    print('A равно B')
else:
    print('A не равно B')
```

Пример программы на Python

```
>>> ================================ RESTART
>>>
Введите A:
10
Введите B:
5
A не равно B
>>> |
```

Результат выполнения программы с использованием условного оператора if-else

3. Конструкция if – elif – else

Для реализации выбора из нескольких альтернатив можно использовать конструкцию if – elif – else.

Синтаксис оператора if – elif – else выглядит так:

if логическое выражение_1:

команда_1

команда_2

...

команда_n

elif логическое выражение_2:

команда_1

команда_2

...

команда_n

elif логическое выражение_3:

команда_1

команда_2

...

команда_n

else:

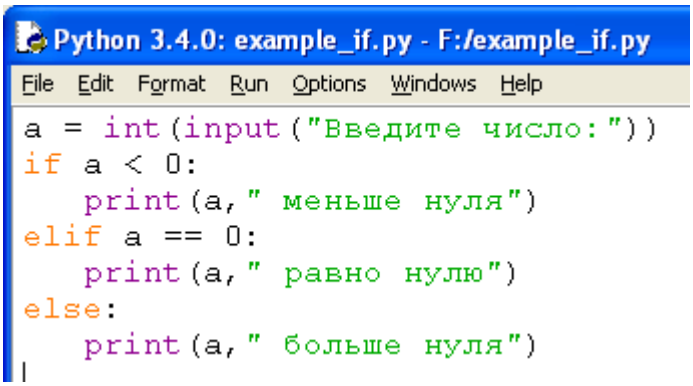
команда_1

команда_2

...

команда_n

Программа запрашивает число у пользователя и сравнивает его с нулём a<0. Если оно меньше нуля, то выводится сообщение об этом. Если первое логическое выражение не истинно, то программа переходит ко второму - a==0. Если оно истинно, то программа выведет сообщение, что число равно нулю, в противном случае, если оба вышеуказанных логических выражения оказались ложными, то программа выведет сообщение, что введённое число больше нуля.



Пример программы на Python

```
Введите число:41
41  больше нуля
>>> ================================ RESTART =
>>>
Введите число:-5
-5  меньше нуля
>>> ================================ RESTART =
>>>
Введите число:0
0  равно нулю
>>>
```

Результат выполнения программы с использованием условного оператора if-elif-else

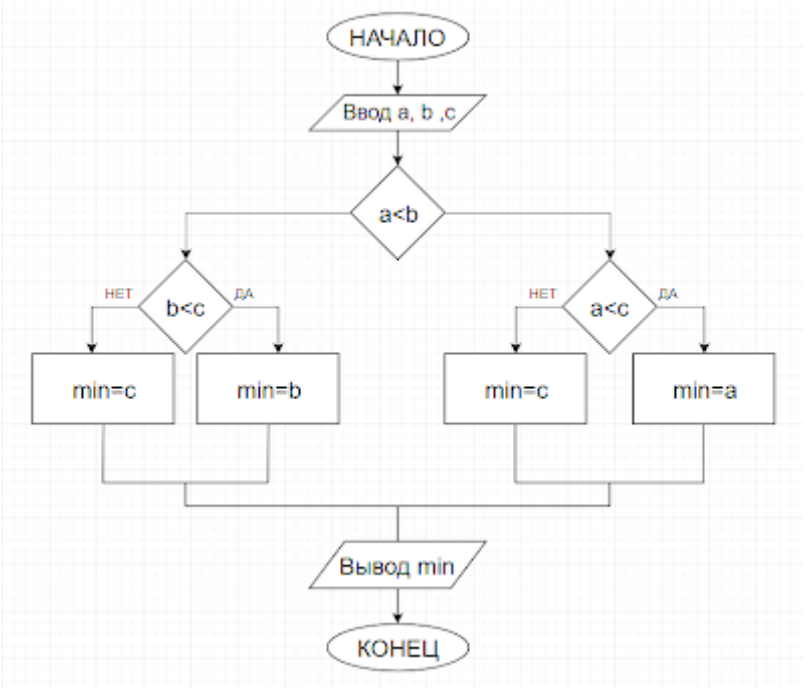
Пример

Вариант 0

Дано 3 числа. Найти минимальное среди них и вывести на экран.

Решение

Для простоты построим блок-схему задачи.



Командами

a=input('')

b=input('')

c=input('')

введём три числа, присвоив значения переменным a, b, c.

Условной конструкцией if-else проверим на истинность логическое выражение a<b. Если оно истинно, то переходим на проверку логического выражения a<c. Если оно истинно, то переменной "y" присвоим значение переменной "a", т.е. "a" будет минимальным, а иначе "y" присвоится значение переменной "c".

Если в начале логическое выражение a<b оказалось ложным, то переходим на проверку другого логического выражения b<c.

Если оно истинно, то "y" присвоится значение переменной "b", иначе "c".

Командой print() выводим минимальное значение.

Пример программы:

```
#нахождение минимального из 3-х чисел
a=input('Введите целое число \n')
b=input('Введите целое число \n')
c=input('Введите целое число \n')
if a<b:
    if a<c:
        y=a
    else:
        y=c
else:
    if b<c:
        y=b
    else:
        y=c
print('Минимальное:',y)
```

Результат выполнения программы:

```
Введите целое число
2
Введите целое число
5
Введите целое число
1
Минимальное: 1
```

Задания для самостоятельной работы (по вариантам)

Вариант 1

Даны три целых числа. Выбрать из них те, которые принадлежат интервалу [1,3].

Вариант 2

Дан номер года (положительное целое число). Определить количество дней в этом году, учитывая, что обычный год насчитывает 365 дней, а високосный — 366 дней. Високосным считается год, делящийся на 4, за исключением тех годов, которые делятся на 100 и не делятся на 400 (например, годы 300, 1300 и 1900 не являются високосными, а 1200 и 2000 — являются).

Вариант 3

Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 3% предоставляется в том случае, если сумма покупки больше 500 руб., в 5% - если сумма больше 1000 руб.

Вариант 4

Написать программу, которая бы по введенному номеру единицы измерения (1 — килограмм, 2 — миллиграмм, 3 — грамм, 4 — тонна, 5 — центнер) и массе М выдавала соответствующее значение массы в килограммах.

Вариант 5

Найти косинус минимального из 4 заданных чисел.

Вариант 6

Вывести на экран синус максимального из 3 заданных чисел.

Вариант 7

Даны три стороны одного треугольника и три стороны другого треугольника. Определить, будут ли эти треугольники равновеликими, т. е. имеют ли они равные площади. Если это не так, то вывести «Foul!!!»

Вариант 8

Составьте программу подсчёта площади равнобедренного треугольника. Если площадь треугольника чётная, разделить её на 2, в противном случае вывести сообщение «Не могу делить на 2!»

Вариант 9

Составить программу, которая по данному числу (1-12) выводит название соответствующего ему месяца на английском языке.

Вариант 10

Составить программу, осуществляющую перевод величин из радианной меры в градусную или наоборот. Программа должна запрашивать, какой перевод нужно осуществить, и выполнять указанное действие.

Вариант 11

Дано три числа. Найти количество положительных чисел среди них;

Вариант 12

Если действительные числа x и y – одного знака, найти их среднее геометрическое, в противном случае найти их среднее арифметическое.

Вариант 13

Определить, существует ли прямоугольный треугольник со сторонами x,y,z. Если – да, вычислить его площадь.

Вариант 14

Определить, существует ли треугольник с длинами сторон a, b, c. Если – да, вычислить его площадь по формуле Герона.

Формула Герона имеет вид:

S=p(p-a)(p-b)(p-c), где p=12(a+b+c)

Вариант 15

Вычислить значение функции f(x), если

$$f(x)=\begin{cases} 0{,}5-\sqrt[4]{|x|}\ , & x\geq 0 \\ \dfrac{\sin^2 x^2}{|x+1|}\ , & x<0 \end{cases}$$

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия;
2. Цель занятия;
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»;
4. Список используемых источников;
5. Выводы и предложения;
6. Дата и подпись курсанта и преподавателя.

Вопросы для самопроверки

1. Какой вычислительный процесс называется разветвляющимся?
2. Опишите синтаксис оператора if.
3. Дайте определение понятию «логическое выражение»
4. Опишите конструкцию if-elif-else.
5. Что позволяет выполнить оператор ветвления if.

Практическое занятие №27 Работа с циклами в Python

Цель занятия:

1. Познакомиться с циклическими конструкциями

Исходные данные: теоретический материал, программа Python

Содержание и порядок выполнения:

1. Изучить теоретическую часть;
2. Выполнить задания.

Теоретическая часть

В Python существуют два типа цикличных выражений:

- Цикл while

- Цикл for

1. Цикл while в Python

Инструкция while в Python повторяет указанный блок кода до тех пор, пока указанное в цикле логическое выражение будет оставаться истинным.

Синтаксис цикла while:

while логическое выражение:

команда 1

команда 2

...

команда n

После ключевого слова while указывается условное выражение, и пока это выражение возвращает значение True, будет выполняться блок инструкций, который идет далее.

Все инструкции, которые относятся к циклу while, располагаются на последующих строках и должны иметь отступ от начала строки (4 пробела).

```
#! Программа по вычислению факториала
number = int(input("Введите число: "))
i = 1
factorial = 1
while i <= number:
    factorial *= i
    i += 1
print("Факториал числа", number, "равен", factorial)
```

Пример программы на Python

```
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Введите число: 5
Факториал числа 5 равен 120
>>>
```

Результат выполнения программы с использованием циклического оператора while

2.Цикл for в Python:

Цикл for в Python обладает способностью перебирать элементы любого комплексного типа данных (например, строки или списка).

Синтаксис цикла for:

for int **in** range():

команда 1

команда 2

...

команда

Переменной int присваивается значение первого элемента функции range(), после чего выполняются команды. Затем переменной int присваивается следующее по порядку значение и так далее до тех пор, пока не будут перебраны все элементы функции range().

Функция range() является универсальной функцией Python для создания списков (list) содержащих арифметическую прогрессию. Чаще всего она используется в циклах for.

range(старт, стоп, шаг) - так выглядит стандартный вызов функции range() в Python. По умолчанию старт равняется нулю, шаг единице.

Вариант 0

1. Найти сумму n элементов следующего ряда чисел: 1 -0.5 0.25 -0.125 ... n. Количество элементов (n) вводится с клавиатуры. Вывести на экран каждый член ряда и его сумму. Решить задачу используя циклическую конструкцию for.

Решение:

В данном случае ряд чисел состоит из элементов, где каждый следующий меньше предыдущего в два раза по модулю и имеет обратный знак. Значит, чтобы получить следующий элемент, надо предыдущий разделить на -2.

Какой-либо переменной надо присвоить значение первого элемента ряда (в данном случае это 1). Далее в цикле добавлять ее значение к переменной, в которой накапливается сумма, после чего присваивать ей значение следующего элемента ряда, разделив текущее значение на -2. Цикл должен выполняться n раз.

```
n=int(input('Введите количество элементов последоваетльности: '))
x=1
s=0
print(x)
for i in range(n):
    s+=x
    x/=-2
    print(x)
print('Сумма ряда:',s)
```

Пример программы с циклом for

```
Введите количество элементов последоваетльности: 5
1
-0.5
0.25
-0.125
0.0625
-0.03125
Сумма ряда: 0.6875
```

Результат выполнения программы

2. Дано целое число, не меньшее 2. Выведите его наименьший натуральный делитель, отличный от 1.

Решение:

Для начала введём целое число командой int(input(текст сообщения)).

Затем зададим переменной i значение 2. Переменная i выполняет роль счётчика. Если задать ей значение 1, то условие задачи не будет выполнено, а результатом всегда будет 1.

В цикле while в качестве логического выражения используется команда n%i сравниваемая с нулём. Таким образом, если остаток от деления введённого числа на текущее значение i не равно нулю, то счётчик увеличивается на 1, а если равно нулю цикл заканчивается и командой print() выводится сообщение и значение i.

```
n = int(input('Введите целое число не меньшее 2\n'))
i = 2
while n%i != 0:
    i+=1
print('наименьший натуральный делитель:',i)
```

Пример программы с циклом while

```
Введите целое число не меньшее 2
49
наименьший натуральный делитель: 7
```

Результат выполнения программы

Вариант 1

1. Дано вещественное число – цена 1 кг конфет. Вывести стоимость 1, 2, … 10 кг конфет. Решить задачу используя циклическую конструкцию for.

2. Дана непустая последовательность целых чисел, оканчивающаяся нулем. Найти: а) сумму всех чисел последовательности; б) количество всех чисел последовательности

Решить задачу используя циклическую конструкцию while.

Вариант2

1. Даны два числа A и B (A<B). Найти сумму всех целых чисел от A до B включительно. Решить задачу используя циклическую конструкцию for.

2. Дана последовательность отрицательных целых чисел, оканчивающаяся положительным числом. Найти среднее арифметическое всех чисел последовательности (без учета положительным числа).

Решить задачу используя циклическую конструкцию while.

Вариант 3

1. Даны два числа A и B (A<B). Найти сумму квадратов всех целых чисел от A до B включительно. Решить задачу используя циклическую конструкцию for.

2. Дана последовательность из n целых чисел. Первое число в последовательности чётное. Найти сумму всех идущих подряд в начале последовательности чётных чисел. Условный оператор не использовать

Решить задачу используя циклическую конструкцию while.

Вариант 4

1. Найти среднее арифметическое всех целых чисел от a до 200 (значения a и b вводятся с клавиатуры; a≤ 200). Решить задачу используя циклическую конструкцию for.

2. Дана последовательность из n вещественных чисел, начинающаяся с положительного числа. Определить, какое количество положительных чисел записано в начале последовательности. Условный оператор не использовать.

Решить задачу используя циклическую конструкцию while.

Вариант 5

1. Найти сумму всех целых чисел от a до b (значения a и b вводятся с клавиатуры; b ≥a). Решить задачу используя циклическую конструкцию for.

2.Дано целое число N (> 0), являющееся некоторой степенью числа 2: N = 2^K. Найти целое число K — показатель этой степени.

Решить задачу используя циклическую конструкцию while.

Вариант 6

1. Найти сумму квадратов всех целых чисел от a до 50 (значение a вводится с клавиатуры; 0 ≤a≤50). Решить задачу используя циклическую конструкцию for.

2.Дано целое число N (> 1). Найти наименьшее целое число K, при котором выполняется неравенство 5^K > N.

Решить задачу используя циклическую конструкцию while.

Вариант 7

1. Дана непустая последовательность целых чисел, оканчивающаяся нулем.

Найти:

а) сумму всех чисел последовательности;

б) количество всех чисел последовательности.

Решить задачу используя циклическую конструкцию for.

2.Дано целое число N (> 1). Найти наибольшее целое число K, при котором выполняется неравенство 2^K > N.

Решить задачу используя циклическую конструкцию while.

Вариант 8

1. Дана последовательность из n вещественных чисел. Первое число в последовательности нечетное. Найти сумму всех идущих подряд в начале последовательности нечетных чисел. Условный оператор не использовать. Решить задачу используя циклическую конструкцию for.

2.Дано целое число N (> 0). Используя операции деления нацело и взятия остатка от деления, найти количество и сумму его цифр.

Решить задачу используя циклическую конструкцию while.

Вариант 9

1. Среди чисел 1, 4, 9, 16, 25, ... найти первое число, большее n. Решить задачу используя циклическую конструкцию for.

2.Среди чисел 1, 5 10, 16, 23, ... найти первое число, большее n. Условный оператор не использовать.

Решить задачу используя циклическую конструкцию while.

Вариант 10

1. Известны оценки по физике каждого из 20 учеников класса. Определить среднюю оценку. Решить задачу используя циклическую конструкцию for.

2.Дано число A (> 1). Вывести наибольшее из целых чисел K, для которых сумма 1 + 1/2 + … + 1/K будет меньше A, и саму эту сумму.

Решить задачу используя циклическую конструкцию while.

Вариант 11

1. Известно сопротивление каждого из элементов электрической цепи. Все элементы соединены последовательно. Определить общее сопротивление цепи. Решить задачу используя циклическую конструкцию for.

2.Дано целое число N (> 0). Найти наибольшее целое число K, квадрат которого не превосходит N: K^2 ≤ N. Функцию извлечения квадратного корня не использовать.

Решить задачу используя циклическую конструкцию while.

Вариант 12

1. Известны оценки по физике каждого ученика двух классов. Определить среднюю оценку в каждом классе. Количество учащихся в каждом классе одинаковое. Решить задачу используя циклическую конструкцию for.

2. Выведите на экран для числа 2 его степени от 0 до 20

Решить задачу используя циклическую конструкцию while.

Вариант 13

1. В области 12 районов. Известны количество жителей (в тысячах человек) и площадь (в км2) каждого района. Определить среднюю плотность населения по области в целом. Решить задачу используя циклическую конструкцию for.

2.Мой богатый дядюшка подарил мне один доллар в мой первый день рождения. В каждый день рождения он удваивал свой подарок и прибавлял к нему столько долларов, сколько лет мне исполнилось. Написать программу, указывающую, к какому дню рождения подарок превысит 100$.

Решить задачу используя циклическую конструкцию while.

Вариант 14

1. Одноклеточная амеба каждые 3 часа делится на 2 клетки. Определить, сколько клеток будет через 3, 6, 9, ..., 24 часа, если первоначально была одна амеба. Решить задачу используя циклическую конструкцию for.

2. Вывести таблицу значений функции y = -0.5x + x. Значения аргумента (x) задаются минимумом, максимумом и шагом. Например, если минимум задан как 1, максимум равен 3, а шаг 0.5. То надо вывести на экран изменение x от 1 до 3 с шагом 0.5 (1, 1.5, 2, 2.5, 3) и значения функции (y) при каждом значении x.

Решить задачу используя циклическую конструкцию while.

Вариант 15

1. Начав тренировки, лыжник в первый день пробежал 10 км. Каждый следующий день он увеличивал пробег на 10% от пробега предыдущего дня. Определить:

а) пробег лыжника за второй, третий, ..., десятый день тренировок;

б) какой суммарный путь он пробежал за первые 7 дней тренировок.

Решить задачу используя циклическую конструкцию for.

2. Найти сумму и произведение цифр, введенного целого числа. Например, если введено число 325, то сумма его цифр равна 10 (3+2+5), а произведение 30 (3*2*5).

Решить задачу используя циклическую конструкцию while.

Выводы и предложения о проделанной работе

Содержание отчета:

1.Наименование практического занятия;

2.Цель занятия;

3.Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»;

4.Список используемых источников;

5.Выводы и предложения;

6.Дата и подпись курсанта и преподавателя.

Вопросы для самопроверки

1. Какие типы цикличных выражений существуют в Python?
2. Какой способностью обладает цикл for в Python?
3. Напишите синтаксис цикла while

Практическое занятие №28 Работа со строками в Python

Цель занятия:

1. Познакомиться с методами работы со строками

Исходные данные: теоретический материал, программа Python

Содержание и порядок выполнения:

1. Изучить теоретическую часть;
2. Выполнить задания.

Теоретическая часть

Строка — базовый тип, представляющий из себя неизменяемую последовательность символов; str от «string» — «строка».

Функции и методы работы со строками

Функция или метод	Назначение

S1 + S2	Конкатенация (сложение строк)
S1 * 3	Повторение строки
S[i]	Обращение по индексу
S[i:j:step]	Извлечение среза
len(S)	Длина строки
S.join(список)	Соединение строк из последовательности str через разделитель, заданный строкой
S1.count(S[, i, j])	количество вхождений подстроки s в строку s1. Результатом является число. Можно указать позицию начала поиска i и окончания поиска *j*
S.find(str, [start],[end])	Поиск подстроки в строке. Возвращает номер первого вхождения или -1
S.index(str, [start],[end])	Поиск подстроки в строке. Возвращает номер первого вхождения или вызывает ValueError
S.rindex(str, [start],[end])	Поиск подстроки в строке. Возвращает номер последнего вхождения или вызывает ValueError
S.replace(шаблон, замена)	Замена шаблона
S.split(символ)	Разбиение строки по разделителю
S.upper()	Преобразование строки к верхнему регистру
S.lower()	Преобразование строки к нижнему регистру

Ниже приведена программа, демонстрирующая использование функций и методов работы со строками.

example_string.py - K:\Лабораторные Python\example_string.py (3.7.1)

File Edit Format Run Options Window Help

```
s1="Пропаганда"
s2="Сенсация"
s3="Сенсация*Сенсация*Сенсация*Сенсация"
s4='OxOxOxAx'
print('s1 = ',s1)
print('s2 = ',s2)
print('s3 = ',s3)
print('s4 = ',s4)
print('s1+s2 = ',s1+s2) #сложение двух строк
print('s1*3 = ',s1*3)   #умножение строки на 3, т.е.строка выведется 3 раза
print('s1[2] = ',s1[2]) #вывод элемента строки s1 с индексом 2
print('s1[2,4] = ',s1[2:4]) #извлечение среза строки s1 начиная с индекса 2
                            #и заканчивая индексом 4
print('s3.count = ',s3.count(s2))   #количество вхождений подстроки s2 в S3,
                                    #в результате выведтеся число
print('s1.find(''a'') = ',s1.find('a'))  #поиск подстроки 'a' в строке s1
                                         #результатом будет номер первого вхождения
print('s1.index(''п'') = ',s1.index('п'))#поиск подстроки 'п' в строке s1
                                         #результатом будет номер первого вхождения
print('s1.rindex(''д'') = ',s1.rindex('a'))#поиск подстроки 'a' в строке s1
                                           #возвращает номер последнего вхождения
print('s4.replace(''Ox'',''Ax'',2) = ',s4.replace('Ox','Ax',2))#замена шаблона. Строка 'Ox' - это шаблон
                                            #строка 'Ax' - это замена
                                            #в строке 4 последоваетльность 'Ox' будет заменена
                                            #на 'Ax' с шагом 2
print('s3.split(''*'') = ',s3.split('*'))#разбиение по разделителю *
print('s1.upper = ',s1.upper())#перевод символов в верхний регистр
print('s1.lower = ',s1.lower())#перевод символов в нижний регистр
```

Ln: 20 Col: 40

Пример программы на Python

```
Python 3.7.1 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:05:16) [MSC v.19
1)] on win32
Type "help", "copyright", "credits" or "license()" for more infor
>>>
============= RESTART: K:\Лабораторные Python\example_string.py =
s1 =  Пропаганда
s2 =  Сенсация
s3 =  Сенсация*Сенсация*Сенсация*Сенсация
s4 =  ОхОхОхАх
s1+s2 =  ПропагандаСенсация
s1*3 =  ПропагандаПропагандаПропаганда
s1[2] =  о
s1[2,4] =  оп
s3.count =  4
s1.find(а) =  4
s1.index(п) =  3
s1.rindex(д) =  9
s4.replace(Ох,Ах,2) =  АхАхОхАх
s3.split(*) =  ['Сенсация', 'Сенсация', 'Сенсация', 'Сенсация']
s1.upper =  ПРОПАГАНДА
s1.lower =  пропаганда
```

Результат выполнения программы с использованием функций и методов работы со строками

Пример

Вариант 0

Проверить, будет ли строка читаться одинаково справа налево и слева направо (т. е. является ли она палиндромом).

Решение

Сначала введём строку командой: s=input('Введите строку ').

Затем определим логическую переменную flag и присвоим ей значение 1: flag=1.

Для начала в введённой строке нужно удалить пробелы. Для этого воспользуемся циклической конструкцией for, которая выполнится столько раз, какую имеет длину строка. Длину строки определим функцией len(s).

В теле цикла будем проверять следующее условие: s[i]!=' '. Данное логическое выражение будет истинно в том случае, если i-ый элемент строки не будет равен пробелу, тогда выполнится команда следующая после двоеточия: string+=s[i].

К строке string, которая была объявлена в начале программы, будет добавляться посимвольно строка s, но уже без пробелов.

Для проверки строки на "палиндром" воспользуемся циклической конструкцией for.

Длина половины строкинаходится делением нацело на 2.Если количество символов нечетно,то стоящий в середине не учитывается,т.к. его сравниваемая пара - он сам.

Количество повторов цикла равно длине половины строки. Длину строки определим функцией len(s), где аргумент введённая нами строка s. Зная длину строки, можно вычислить количество повторов цикла. Для этого целочисленно разделим длину строки на 2: len(s)//2.

Для задания диапазона для цикла используем функцию range(), в которой аргументом будет являться половина длины строки: range(len(s//2)).

for i in range(len(s//2)).

Если символ с индексом i не равен "симметричному" символу с конца строки (который находится путем индексации с конца)

if s[i] != s[-1-i],

то переменной flag присваивается значение 0 и происходит выход из цикла командой break.

Далее, при помощи условной конструкции if-else в зависимости от значения flag либо - 0, либо -1 выводится сообщение, что строка палиндром, либо нет.

```
s=input('Введите строку \n')
flag=1
string=''
for i in range(len(s)):
    if s[i]!=' ':
        string+=s[i]
print(string)
for i in range(len(s)//2):
    if string[i]!=string[-i-1]:
        flag=0
        break
if flag: print('Палиндром')
else: print('не палиндром')
```

Пример программы на Python

```
Введите строку
а роза упала на лапу азора
арозаупаланалапуазора
Палиндром
```

Результат выполнения программы

Задания для самостоятельной работы (по вариантам)

Вариант 1

Дана строка, содержащая русскоязычный текст. Найти количество слов, начинающихся с буквы "е".

Вариант 2

В строке заменить все двоеточия (:) знаком процента (%). Подсчитать количество замен.

Вариант 3

В строке удалить символ точку (.) и подсчитать количество удаленных символов.

Вариант 4

В строке заменить букву(а) буквой (о). Подсчитать количество замен.Подсчитать, сколько символов в строке.

Вариант 5

В строке заменить все заглавные буквы строчными.

Вариант 6

В строке удалить все буквы "а" и подсчитать количество удаленных символов.

Вариант 7

Дана строка. Преобразовать ее, заменив звездочками все буквы "п", встречающиеся среди первых n/2 символов. Здесь n - длина строки.

Вариант 8

Дана строка, заканчивающаяся точкой. Подсчитать, сколько слов в строке.

Вариант 9

Определить, сколько раз в тексте встречается заданное слово.

Вариант 10

Дана строка-предложение на английском языке. Преобразовать строку так, чтобы каждое слово начиналось с заглавной буквы.

Вариант 11

Дана строка. Подсчитать самую длинную последовательность подряд идущих букв «н». Преобразовать ее, заменив точками все восклицательные знаки.

Вариант 12

Дана строка. Вывести все слова, оканчивающиеся на букву "я".

Вариант 13

Дана строка символов, среди которых есть одна открывающаяся и одна закрывающаяся скобки. Вывести на экран все символы, расположенные внутри этих скобок.

Вариант 14

Дана строка. Вывести все слова, начинающиеся на букву "а" и слова оканчивающиеся на букву "я".

Вариант 15

Дана строка текста. Подсчитать количество букв «т» в строке

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия;
2. Цель занятия;
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»;
4. Список используемых источников;
5. Выводы и предложения;
6. Дата и подпись курсанта и преподавателя.

Вопросы для самопроверки

1. Что называется строкой в Python
2. Перечислите функции и методы работы со строками.
3. Для чего служит функция S.split**?**
4. Для чего используется функцияrange()?

Практическое занятие №29Работа со списками, кортежами, словарями. Операции над списками в Python

Цель занятия: Изучение текстовых списков, кортежей и словарей в Python.

Исходные данные: теоретический материал, программа Python

Содержание и порядок выполнения:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

Часто нам нужно держать много однообразных данных в одном файле, например, список учеников колледжа или номера телефонов в справочнике. В Python такие наборы данных можно организовывать в списки, кортежи и словари.

Список (list) – это структура, состоящая из элементов, расположенных в определенном порядке. Каждому элементу соответствует номер (или индекс), по которому к нему можно обратиться. Для создания списка вквадратных скобках ([]) через запятую перечисляются все его элементы. Например, создадим список членов своей семьи:

```
>>>myfamily=['father','mother','sister','brother']
```

Программа на Python	Результат вывода на экран

`myfamily = ['father', 'mother', 'sister', 'brother']`
`for item in myfamily:`
`  print('Hello', item)`
	Hello father
	Hello mother
	Hello sister
	Hello brother

В данном случае наш список будет храниться в переменной myfamily.

Когда список создан, можно написать программу для работы с этим списком. К примеру, напишем приветствие для каждого из членов семьи, используя цикл:

Каждый элемент списка имеет свой номер (индекс). Нумерация элементов начинается с нуля:

```
a = ["Андрей", "Вера", "Даша", "Коля", "Юра"]
```

0	1	2	3	4
Андрей	Вера	Даша	Коля	Юра
a[0]	a[1]	a[2]	a[3]	a[4]

С каждым элементом списка можно работать отдельно:

```
a = ["Андрей", "Вера", "Даша", "Коля", "Юра"]
print(a[2])
```

Вывод: **Даша**

Список может содержать разные типы объектов. В один и тот же список одновременно можно включать строки, числа, объекты других типов данных:

```
objects=[1,2.6,'Hello',True]
```

Списки можно складывать, тогда новый список будет содержать элементы из обоих списков:

x = [1, 2, 3, 4]

y = [5, 6, 7, 8]

z = x + y print (z)

Результат: [1, 2, 3, 4, 5, 6, 7, 8]

Со списками можно делать много разных операций:

x in A	Проверить, содержится ли элемент **x** в списке **A**. Возвращает True или False	`A = [1, 2, 3, 4, 5, 6, 7, 8]`
		`print (2 in A)`
		`Результат:`
		`True`

min(A)	Найти наименьший элемент в списке **A.**	`A = [1, 2, 3, 4, 5, 6, 7, 8]`
		`print (min(A))`
		`Результат:`
		`1`

max(A)	Найти наибольший элемент в списке **A.**	`A = [1, 2, 3, 4, 5, 6, 7, 8]`
		`print (max(A))`
		`Результат:`
		`8`

Функция нахождения длины списка len(a):

a = ["**Яблоко**", "**Банан**", "**Груша**"]

x = len(a)

print(x)

Результат:3

Сортировка списка. Функция sorted(a):

По возрастанию:

animals = ["кот", "еж", "собака", "барсук"]

animals = **sorted**(animals)

print(animals)

Вывод:

['**барсук**', '**еж**', '**кот**', '**собака**']

Поубыванию:

a = [5, 65, 14, 700, 8]

a = **sorted**(a, reverse = True)

print(a)

Вывод:

[**700, 65, 14, 8, 5**]

Способы вывода списков

1)*С помощью функции*print():

b = [17, 409, 88]

s=[‘f’,’g’,’r’,’a’]

print(b)

print(s)

Вывод:

[**17, 409, 88**]

[**f,g,r,a**]

2) *Вывод каждого элемента списка по отдельности*:

a = ["Андрей", "Вера", "Даша", "Коля", "Юра"]

fori in range(5):

print(a[i])

Вывод:

Андрей

Вера

Даша

Коля

Юра

3)Вывод каждого элемента списка поотдельности в одной строке:

a = ["Андрей", "Вера", "Даша", "Коля", "Юра"]

fori in range(5):

print(a[i], end = " ")

Вывод

Андрей Вера Даша Коля Юра

4) Вывод элементов списка без обращения к индексам элементов:

fruits = ["Яблоко", "Банан", "Груша"]

for x in fruits:

print(x, end = " ")

Вывод

Яблоко Банан Груша

Кортеж (**tuple**), как и список, представляет собой последовательность элементов. Однако хранящиеся в нем элементы нельзя изменять, добавлять или удалять. Для создания кортежа используются круглые скобки, в которые помещаются его значения, разделенные запятыми:

```
user=('Timur',23,1.10.1998)
print(user)
```

В кортежах удобно хранить свойства объектов, например, имя, возраст, дату рождения. Если вдруг кортеж состоит из одного элемента, то после единственного элемента кортежа необходимо поставить запятую:

```
user=('Tom',)
```

Словари (**dictionary**) – это структура данных, в которой каждый элемент вместо индекса имеет уникальный ключ. Элементы словаря можно изменять. Для создания словаря используются фигурные скобки ({}):

```
dictionary={ключ1:значение1,ключ2:значение2,...}
```

Создадим словарь под именемmycollege:

```
mycollege = {'4kurs':'Анна, Кирилл, Pavel',
'3kurs':'Игорь,Олег,Ольга'}
```

В этом словаре в качестве ключей используются названия классов, а в качестве значений – имена тех, кто учится в этих классах.

В словарь можно добавить значение, пометив его новым ключом:

```
mycollege['2kurs']='Elena,Алиса,Дима'
print(mycollege)
```

Результат:

```
{'4kurs': 'Анна, Кирилл, Pavel', '3kurs':
'Игорь,Олег,Ольга','2kurs':'Elena,Алиса,Дима}
```

Чтобы изменить значение элемента, нужно придать его ключу новое значение:

```
mycollege = {'4kurs':'Анна, Кирилл, Pavel',
'3kurs':'Игорь,Олег,Ольга'}
mycollege['3kurs']='Matvei,Таня,Света'
```

```
print(mycollege)
```

Результат:

```
{'4kurs':Анна,Кирилл,Pavel','3kurs':'Matvei,Таня,Света'}
```

Используя цикл`for`,можно вывести на экран только ключи словаря:

```
foriinmycollege:
    print(i)
```

Результат:

```
4 kurs
3 kurs
```

Или вывести только значения словаря:

```
foriinmycollege:
    print(mycollege[i])
```

Результат:

```
Анна,Кирилл,Pavel
Matvei, Таня, Света
```

Задания для самостоятельного выполнения:

Задание №1

Создать список своих любимых фильмов. Вывести список тремя способами: а) в строчку; б) в столбик; в) в строчку через запятую.

Задание №2

Ввести с клавиатуры список фамилий 10 -ти писателей. Отсортировать список в алфавитном порядке и вывести его.

Задание №3

Создать список своих любимых песен. Отсортировать список в алфавитном порядке и вывести его. Найти длину списка и вывести результат на экран.

Задание №4

Создайте словарь «Рыбы», а его элементы разделите на 3 вида:

«речные», «озерные» и «морские рыбы». Выведите на экран сначала только ключи, а потом элементы словаря.

Задание №5

Добавить в данный словарь «Рыбы» ключ «океанические», заполнить его и вывести на экран обновленный словарь.

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки

1. Что такое список, кортеж, словарь в Питоне?
2. Как создаются списки?
3. Какие операции можно производить со списками?
4. Какие существуют способы вывода списка?
5. Как записывается функция определения длины списка?
6. Как отсортировать список?

Практическое занятие №30 Одномерные массивы как списками. Заполнение и обработка одномерных массивов в Python

Цель занятия: Изучение одномерных массивов в Python.

Исходные данные: теоретический материал, программа Python

Содержание и порядок выполнения:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть:

Одномерные массивы (списки) в Python — это определенное количество элементов одного типа, которые имеют общее имя, и каждый элемент имеет свой индекс — порядковый номер.

Часто для работы содномерными массивами используются списки.

Список (list) – это структура данных для хранения объектов различных типов.

Списки являются упорядоченными последовательностями, которые состоят из различных типов данных, заключающихся в квадратные скобки [ ] и отделяющиеся друг от друга с помощью запятой.

Создание списков на Python.

Создать список можно несколькими способами

1. Получение списка через присваивание конкретных значений.

Так выглядит в коде Python пустой список:

s = [] # Пустой список

Примеры создания списков со значениями:

```
l=[5,75,-4,7,-51]# список целых чисел
l=[1.13,5.34,12.63,4.6,34.0,12.8]# список из вещественных чисел
l=["Оля", "Владимир", "Михаил", "Дарья"]# список из строк
l=["Москва", "Иванов", 12, 124]   # смешанный список
l=[[0, 0, 0], [1, 0, 1], [1, 1, 0]]   # список, состоящий из списков
l=['s', 'p', ['isok'], 2] # список из значений и списка
```

Списки можно складывать (конкатенировать) с помощью знака «+»:

```
l=[1, 3]+[4,23]+[5]
print('l=[1, 3]+[4,23]+[5] =',l)
```

Результат:

```
>>>
l=[1, 3]+[4,23]+[5] = [1, 3, 4, 23, 5]
>>>
```

2. Создание списка при помощи функции Split().

Используя функцию split в Python можно получить из строки список.

stroka ="Привет, страна"

lst=stroka.split(",")

```
stroka ="Здравствуй, Дедушка Мороз"  #stroka - строка
lst=stroka.split(",") #lst - список
print('stroka = ',stroka)
print('lst=stroka.split(","):',lst)
```

Результат:

```
============== RESTART: C:/Users/maxim/Desktop/ex_list_
stroka =  Здравствуй, Дедушка Мороз
lst=stroka.split(","): ['Здравствуй', ' Дедушка Мороз']
```

3. Генераторы списков.

В Python создать список можно также при помощи генераторов.

Первый способ.

Сложение одинаковых списков заменяется умножением:

Список из 10 элементов, заполненный единицами

l = [1]*10

Второйспособ.

Пример 1.

l = [i for i in range(10)]

Пример 2.

c = [c * 3 for c in 'list']

print (c) # ['lll', 'iii', 'sss', 'ttt']

```
Создание списка из строки.
l = list (строка):
 ['с', 'т', 'р', 'о', 'к', 'а']

Создание списка при помощи функции Split().
stroka=" Hello, friend "
lst=stroka.split(","):
 ['Hello', ' friend']

Генераторы списков.
Первый способ.
l = [1]*10:
 [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Второй способ. Пример 1.
l = [i for i in range(10)]:
 [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Второй способ. Пример 2.
c=[c*3 for c in "list"]:
 ['lll', 'iii', 'sss', 'ttt']
```

Примеры использования генераторов списка.

Пример 1.

Заполнить список квадратами чисел от 0 до 9, используя генератор списка.

Решение:

l = [i*i for i in range(10)]

Пример 2.

Заполнить список числами, где каждое последующее число больше на 2.

l = [(i+1)+i for i in range(10)]

print(l)

```
Заполнить список квадратами чисел от 0 до 9, используя генератор списка.
l = [i*i for i in range(10)]:
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

Заполнить список числами, где каждое последующее число больше на 2.
l = [(i+1)+i for i in range(10)]:
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
```

Модуль random предоставляет функции для генерации случайных чисел, букв, случайного выбора элементов последовательности.

random.randint(A, B) - случайное целое число N, A ≤ N ≤ B.

random.random() - случайное число от 0 до 1.

Случайные числа в списке:

10 чисел, сгенерированных случайным образом в диапазоне (10,80)

from random import randint

l = [randint(10,80) for x in range(10)]

10 чисел, сгенерированных случайным образом в диапазоне (0,1)

l = [random() for i in range(10)]

```
from random import *
l = [randint(10,80) for i in range(10)]
print('10 чисел, сгенерированных случайным образом в диапазоне (10,80).')
print('l = [randint(10,80) for x in range(10)]:')
print(l)
print()

l = [random() for i in range(10)]
print('10 чисел сгенерированных в диапазоне от 0 до 1.')
print('l = [random() for i in range(10):')
for i in range(len(l)):
    print ('{:.2f}'.format(l[i]), end = " " )
```

```
10 чисел, сгенерированных случайным образом в диапазоне (10,80).
l = [randint(10,80) for x in range(10)]:
[70, 33, 79, 61, 34, 27, 11, 55, 52, 31]

10 чисел сгенерированных в диапазоне от 0 до 1.
l = [random() for i in range(10):
0.66 0.97 0.87 0.57 0.54 0.83 0.57 0.65 0.04 0.07
```

4. Ввод списка (массива) в языке Python.

Для ввода элементов списка используется цикл for и команда range ():

fori in range(N):

  x[i] = int(input())

Более простой вариант ввода списка:

x = [int(input()) for i in range(N)]

```
print('Ввод списка. Пример 1:')
x=[]
for i in range(4):
    x.append(int(input()))
print(x)

x=[]
print('Ввод списка. Пример 2:')
x = [ int(input()) for i in range(4) ]
print(x)
```

```
Ввод списка. Пример 1:
45
4
85
2
[45, 4, 85, 2]
Ввод списка. Пример 2:
4
5
7
8
[4, 5, 7, 8]
```

Функция int здесь используется для того, чтобы строка, введенная пользователем, преобразовывалась в целые числа.

5.Вывод списка (массива) в языке Python.

Вывод целого списка (массива):

print (L)

Поэлементный вывод списка (массива):

for i inrange(N):

  print(L[i], end = " ")

```
Вывод целого списка (массива)
[1, 56, 6, 3, 6, 7, 3, 37, 7, 37, 37]

Поэлементный вывод списка (массива)
1 56 6 3 6 7 3 37 7 37 37
```

2. Методы списков.

Метод	Что делает
list.append(x)	Добавляет элемент в конец списка
list.extend(L)	Расширяет список list, добавляя в конец все элементы списка L
list.insert(i, x)	Вставляет перед i-ым элементом значение x
list.remove(x)	Удаляет первый элемент в списке, имеющий значение x. ValueError, если такого элемента не существует
list.pop([i])	Удаляет i-ый элемент и возвращает его. Если индекс не указан, удаляется последний элемент
list.index(x, [start [, end]])	Возвращает положение первого элемента со значением x (при этом поиск ведется от start до end)
list.count(x)	Возвращает количество элементов со значением x
list.reverse()	Разворачивает список
list.copy()	Поверхностная копия списка
list.clear()	Очищает список

Ниже приведена программа, демонстрирующая методы работы списков.

```
a=[0,2,2,2,4] #список a
b=[5,6,7,2,9] #список b
print('Исходный список a:',a)
print('Исходный список b:',b)
x=99
y=5

a.append(x)
print('a.append(x):',a)

a.extend(b)
print('a.extend(b):',a)

a.insert(3,x)
print('a.insert(3,x):',a)

a.remove(x)
print('a.remove(x):',a)

print('a.pop(5):',a.pop(5))
print(a)

print('a.index(y,0,len(a)):',a.index(y,0,len(a)))

print('a.count(2):',a.count(2))

a.reverse()
print('a.reverse():',a)

z=a.copy()
print('z=a.copy():',z)

z.clear()
print('z.clear():')
print('z =',z)
```

Пример программы на Python

```
Исходный список a: [0, 2, 2, 2, 4]
Исходный список b: [5, 6, 7, 2, 9]
a.append(x): [0, 2, 2, 2, 4, 99]
a.extend(b): [0, 2, 2, 2, 4, 99, 5, 6, 7, 2, 9]
a.insert(3,x): [0, 2, 2, 99, 2, 4, 99, 5, 6, 7, 2, 9]
a.remove(x): [0, 2, 2, 2, 4, 99, 5, 6, 7, 2, 9]
a.pop(5): 99
[0, 2, 2, 2, 4, 5, 6, 7, 2, 9]
a.index(y,0,len(a)): 5
a.count(2): 4
a.reverse(): [9, 2, 7, 6, 5, 4, 2, 2, 2, 0]
z=a.copy(): [9, 2, 7, 6, 5, 4, 2, 2, 2, 0]
z.clear():
z = []
```

Результат выполнения программы

Вариант 0

1. Из массива X длиной n, среди элементов которого есть положительные, отрицательные и равные нулю, сформировать новый массив Y, взяв в него только те элементы из X, которые больше по модулю заданного числа M. Вывести на экран число M, данный и полученные массивы.

Решение:

```
n=int(input('Введите длину массива\n'))
m=int(input('Введите число M\n'))
x=[]
y=[]
for i in range(n):
    print('Введите ',i,'элемент:')
    x.append(int(input()))
for i in range(n):
    if abs(x[i])>m:
        y.append(x[i])
print('Введённое число M:',m)
print('Массив X:',x)
print('Массив Y:',y)
```

```
Введите длину массива
5
Введите число M
20
Введите  0 элемент:
21
Введите  1 элемент:
22
Введите  2 элемент:
5
Введите  3 элемент:
6
Введите  4 элемент:
8
Введённое число M: 20
Массив X: [21, 22, 5, 6, 8]
Массив Y: [21, 22]
```

2. В массиве целых чисел все отрицательные элементы заменить на положительные. Вывести исходный массив и полученный.

Решение:

```
n=int(input('Введите длину массива:'))
a=[]
for i in range(n):
    print('Введите',i,'элемент:')
    a.append(int(input()))
print('Исходный массив:',a)
for i in range(n):
    if a[i]<0:
        a[i]=-a[i]
print('Полученный массив:',a)
```

```
Введите длину массива:5
Введите 0 элемент:
-5
Введите 1 элемент:
-4
Введите 2 элемент:
-6
Введите 3 элемент:
5
Введите 4 элемент:
-7
Исходный массив: [-5, -4, -6, 5, -7]
Полученный массив: [5, 4, 6, 5, 7]
```

Варианты заданий для самостоятельного выполнения:

Вариант 1

1. Дан одномерный массив, состоящий из N целочисленных элементов. Ввести массив с клавиатуры. Найти максимальный элемент. Вывести массив на экран в обратном порядке.

2. В массиве действительных чисел все нулевые элементы заменить на среднее арифметическое всех элементов массива.

Вариант 2

1. Дан одномерный массив, состоящий из N целочисленных элементов. Ввести массив с клавиатуры. Найти минимальный элемент. Вывести индекс минимального элемента на экран.

2. Дан массив целых чисел. Переписать все положительные элементы во второй массив, а остальные - в третий.

Вариант 3

1. В одномерном числовом массиве D длиной n вычислить сумму элементов с нечетными индексами. Вывести на экран массив D, полученную сумму.

2. Дан одномерный массив из 8 элементов. Заменить все элементы массива меньшие 15 их удвоенными значениями. Вывести на экран монитора преобразованный массив.

Вариант 4

1. Дан массив целых чисел. Найти максимальный элемент массива и его порядковый номер.

2. Дан одномерный массив целого типа. Получить другой массив, состоящий только из нечетных чисел исходного массива или сообщить, что таких чисел нет. Полученный массив вывести в порядке убывания элементов.

Вариант 5

1. Дан одномерный массив из 10 целых чисел. Вывести пары отрицательных чисел, стоящих рядом.

2. Дан целочисленный массив размера 10. Создать новый массив, удалив все одинаковые элементы, оставив их 1 раз.

Вариант 6

1. Дан одномерный массив из 10 целых чисел. Найти максимальный элемент и сравнить с ним остальные элементы. Вывести количество меньших максимального и больших максимального элемента.

2. Одномерный массив из 10-и целых чисел заполнить с клавиатуры, определить сумму тех чисел, которые >5.

Вариант 7

1. Дан массив целых чисел. Найти сумму элементов с четными номерами и произведение элементов с нечетными номерами. Вывести сумму и произведение.

2. Переставить в одномерном массиве минимальный элемент и максимальный.

Вариант 8

1. Найдите сумму и произведение элементов списка. Результаты вывести на экран.

2. В массиве действительных чисел все нулевые элементы заменить на среднее арифметическое всех элементов массива.

Вариант 9

1. Дан одномерный массив, состоящий из N вещественных элементов. Ввести массив с клавиатуры. Найти и вывести минимальный по модулю элемент. Вывести массив на экран в обратном порядке.

2. Даны массивы А и В одинакового размера 10. Вывести исходные массивы. Поменять местами их содержимое и вывести в начале элементы преобразованного массива А, а затем — элементы преобразованного массива В.

Вариант 10

1. Определите, есть ли в списке повторяющиеся элементы, если да, то вывести на экран это значение, иначе сообщение об их отсутствии.

2. Дан одномерный массив из 15 элементов. Элементам массива меньше 10 присвоить нулевые значения, а элементам больше 20 присвоить 1. Вывести на экран монитора первоначальный и преобразованный массивы в строчку.

Вариант 11

1 Найти наибольший элемент списка, который делиться на 2 без остатка и вывести его на экран.

2. Дан одномерный массив целого типа. Получить другой массив, состоящий только из четных чисел исходного массива, меньше 10, или сообщить, что таких чисел нет. Полученный массив вывести в порядке возрастания элементов.

Вариант 12

1. Найти наименьший нечетный элемент списка и вывести его на экран.

2. Даны массивы А и В одинакового размера 10. Поменять местами их содержимое и вывести вначале элементы преобразованного массива А, а затем — элементы преобразованного массива В.

Вариант 13

1. Дан одномерный массив целых чисел. Проверить, есть ли в нем одинаковые элементы. Вывести эти элементы и их индексы.

2. Дан одномерный массив из 8 элементов. Заменить все элементы массива меньшие 15 их удвоенными значениями. Вывести на экран монитора преобразованный массив.

Вариант 14

1. Найти максимальный элемент численного массива и поменять его местами с минимальным.

2. Программа заполняет одномерный массив из 10 целых чисел числами, считанными с клавиатуры. Определить среднее арифметическое всех чисел массива. Заменить элементы массива большие среднего арифметического на 1.

Вариант 15

1. Определите, есть ли в списке повторяющиеся элементы, если да, то вывести на экран эти значения.

2. Дан одномерный массив целого типа. Получить другой массив, состоящий только из нечетных чисел исходного массива или сообщить, что таких чисел нет. Полученный массив вывести в порядке убывания элементов

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия
2. Цель занятия
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»
4. Список используемых источников
5. Выводы и предложения
6. Дата и подпись курсанта и преподавателя

Вопросы для самопроверки

1. Что такое массивы?
2. Как создаются списки?
3. Как происходит ввод списка (массива)?
4. Как происходит вывод списка (массива)?

Практическое занятие №31 Работа с двумерными массивами

Цель занятия:

1. Изучение двумерных массивов в Python.

2. Знать - способ описания двумерного массива, способы ввода элементов двумерного массива;

3. Уметь - вводить массивы, получать списки через присваивание конкретных значений, применять функции;

4. Владеть - основными навыками создания программ обработки двумерных массивов.

Исходные данные: теоретический материал, программа Python

Содержание и порядок выполнения:

Изучить теоретическую часть

Выполнить задания

Теоретическая часть

Матрицами называются массивы элементов, представленные в виде прямоугольных таблиц, для которых определены правила математических действий. Элементами матрицы могут являться числа, алгебраические символы или математические функции.

Для работы с матрицами в Python также используются списки. Каждый элемент списка-матрицы содержит вложенный список.

Таким образом, получается структура из вложенных списков, количество которых определяет количество столбцов матрицы, а число элементов внутри каждого вложенного списка указывает на количество строк в исходной матрице.

1. Создание списка

Пусть даны два числа: количество строк n и количество столбцов m. Необходимо создать список размером n×m, заполненный нулями.

Очевидное решение оказывается неверным:

A = [[0] *m]*n

В этом легко убедиться, если присвоить элементу A[0][0] значение 1, а потом вывести значение другого элемента A[1][0] — оно тоже будет равно 1! Дело в том, что [0] * m возвращает ссылку на список из m нулей. Но последующее повторение этого элемента создает список из n элементов, которые являются ссылкой на один и тот же список (точно так же, как выполнение операции B = A для списков не создает новый список), поэтому все строки результирующего списка на самом деле являются одной и той же строкой.

Таким образом, двумерный список нельзя создавать при помощи операции повторения одной строки.

Первый способ.

Сначала создадим список из n элементов (для начала просто из n нулей). Затем сделаем каждый элемент списка ссылкой на другой одномерный список из m элементов:

```
A = [0] * n
for i in range(n):
    A[i] = [0] * m
```

```
n=3
m=3
A=[0]*n
for i in range(n):
    A[i]=[0]*m
print('A:',A)
```

```
A: [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
>>>
```

Второй способ.

Создать пустой список, потом n раз добавить в него новый элемент, являющийся списком-строкой:

```
A = []
for i in range(n):
    A.append([0] * m)
```

```
n=3
m=4
A = []
for i in range(n):
    A.append([0]*m)
print(A)
```

```
A: [[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
>>>
```

2. Ввод вложенного списка (двумерного массива)

Пример:

```
n=5
A = []
for i in range(n):
    b = input()
        for i in range(len(row)):
            row[i] = int(row[i])
A.append(row)
```

```
ex_array_primer.py - E:/ex_array_primer.py (3.4.
File  Edit  Format  Run  Options  Window  Help
n=3
A = []
for i in range(n):
    B = []
    for i in range(n):
        B.append(int(input()))
    A.append(B)
```

```
Python 3.4.3 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.4.3 (v3.4.3:9b73f1c3e601,
tel)] on win32
Type "copyright", "credits" or "l
>>> ==============================
>>>
1
2
3
4
5
6
7
8
9
>>>
```

3. Вывод вложенного списка (двумерного массива)

Для обработки и вывода списка как правило используется два вложенных цикла. Первый цикл по номеру строки, второй цикл по элементам внутри строки. Например, вывести двумерный числовой список на экран построчно, разделяя числа пробелами внутри одной строки, можно так:

```
for i in range(n):
    for j in range(n]):
        print(A[i][j], end = ' ')
    print()
```

```
n=3
A = []
#ввод массива
for i in range(n):
    B = []
    for i in range(n):
        B.append(int(input()))
    A.append(B)
#вывод массива
for i in range(n):
    for j in range(n):
        print(A[i][j], end=' ')
    print()
```

```
1
2
3
4
5
6
7
8
9
1 2 3
4 5 6
7 8 9
```

То же самое, но циклы не по индексу, а по значениям списка:

```
for row in A:
    for elem in row:
        print(elem, end = ' ')
    print()
```

```
A=[[1,2,3,4],[5,6,7,8]]
for row in A:
    for elem in row:
        print(elem, end = ' ')
    print()
```

```
Python 3.7.1 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.7.1 (v3.7.1:260ec2c36a
1)] on win32
Type "help", "copyright", "cred
>>>
==================== RESTART: K
1 2 3 4
5 6 7 8
>>>
```

Для вывода одной строки можно воспользоваться методом join. Использовав этот метод в цикле for можно

```
for row in A:
    print(' '.join(list(map(str, row))))
```

4. Обработка и вывод вложенных списков

Часто в задачах приходится хранить прямоугольные таблицы с данными. Такие таблицы называются матрицами или двумерными массивами. В языке программирования Питон таблицу можно представить в виде списка строк, каждый

элемент которого является в свою очередь списком, например, чисел. Например, создать числовую таблицу из двух строк и трех столбцов можно так:

A = [[1, 2, 3], [4, 5, 6]]

Здесь первая строка списка A[0] является списком из чисел [1, 2, 3].

Тоесть

A[0][0]= 1,

A[0][1]= 2,

A[0][2]= 3,

A[1][0]=4,

A[1][1]=5,

A[1][2]=6.

Используем два вложенных цикла для подсчета суммы всех чисел в списке:

```
S = 0
for i in range(len(A)):
    for j in range(len(A[i])):
        S += A[i][j]
```

Или то же самое с циклом не по индексу, а по значениям строк:

```
S = 0
for row in A:
    for elem in row:
        S += elem
```

```
A=[[1, 2, 3,4],[ 5, 6,7,8]]
#вывод при помощи цикла for и метода join
print('Массив A:')
for i in A:
    print(' '.join(list(map(str, i))))
#Пример 1. Подсчёт суммы всех элементов
S = 0
for i in range(len(A)):
    for j in range(len(A[i])):
        S += A[i][j]
print('Пример 1. Сумма элементов:',S)
#Пример 2. Подсчёт суммы всех элементов
S = 0
for row in A:
    for elem in row:
        S += elem
print('Пример 2. Сумма элементов:',S)
```

```
Массив A:
1 2 3 4
5 6 7 8
Пример 1. Сумма элементов: 36
Пример 2. Сумма элементов: 36
```

5. Пример сложной обработки массива

Пусть дана квадратная матрица из n строк и n столбцов. Необходимо элементам, находящимся на главной диагонали, проходящей из левого верхнего угла в правый нижний (то есть тем элементам A[i][j], для которых i==j) присвоить значение 0, элементам, находящимся выше главной диагонали – значение 1, элементам, находящимся ниже главной диагонали – значение 2. То есть получить такой массив (пример для n==3):

0 1 1

2 0 1

2 2 0

Рассмотрим несколько способов решения этой задачи.

Первый способ.

Элементы, которые лежат выше главной диагонали – это элементы A[i][j], для которых i<j, а для элементов ниже главной диагонали i>j. Таким образом, мы можем сравнивать значения i и j и по ним определять значение A[i][j]. Получаемследующийалгоритм:

```
for i in range(n):
    for j in range(n):
        if i< j:
            A[i][j] = 0
        elifi> j:
            A[i][j] = 2
        else:
            A[i][j] = 1
```

Ниже приведён пример программы, в котором квадратная матрица 3x3 заполняется элементами со значением 9, а затем элементам, находящимся на главной диагонали, проходящей из левого верхнего угла в правый нижний (то есть тем элементам A[i][j], для которых i==j) присваивается значение 0, элементам, находящимся выше главной диагонали – значение 1, элементам, находящимся ниже главной диагонали – значение 2.

```
n=3
A=[]
#заполняем массив 9-ми
for i in range(n):
    A.append([9]*n)
#вывод исходного массива
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()
#заменяем элементы главной диагонали, выше и ниже неё
for i in range(n):
    for j in range(n):
        if i < j:
            A[i][j] = 1
        elif i > j:
            A[i][j] = 2
        else:
            A[i][j] = 0
#вывод изменённого массива
print()
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()
```

```
9 9 9
9 9 9
9 9 9

0 1 1
2 0 1
2 2 0
>>>
```

Второй способ.

Данный алгоритм плох, поскольку выполняет одну или две инструкции if для обработки каждого элемента. Если мы усложним алгоритм, то мы сможем обойтись вообще без условных инструкций.

Сначала заполним главную диагональ, для чего нам понадобится один цикл:

for i in range(n):

A[i][i] = 1

Затем заполним значением 0 все элементы выше главной диагонали, для чего нам понадобится в каждой из строк с номером i присвоить значение элементам A[i][j] для j=i+1, ..., n-1. Здесь нам понадобятся вложенные циклы:

foriinrange(n):

forjinrange(i + 1, n):

A[i][j] = 0

Аналогично присваиваем значение 2 элементам A[i][j] для j=0, ..., i-1:

```
for i in range(n):
    for j in range(0, i):
        A[i][j] = 2
```

Можно также внешние циклы объединить в один и получить еще одно, более компактное решение:

```
for i in range(n):
    for j in range(0, i):
        A[i][j] = 2
    A[i][i] = 1
    for j in range(i + 1, n):
        A[i][j] = 0
```

```
n=3
A=[]
#заполняем массив 9-ми
for i in range(n):
    A.append([9]*n)
#вывод исходного массива
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()
#заменяем элементы главной диагонали, выше и ниже неё
for i in range(n):
    for j in range(0, i):
        A[i][j] = 2
    A[i][i] = 0
    for j in range(i + 1, n):
        A[i][j] = 1
#вывод изменённого массива
print()
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()
```

```
9 9 9
9 9 9
9 9 9

0 1 1
2 0 1
2 2 0
>>>
```

Третий способ.

А вот такое решение использует операцию повторения списков для построения очередной строки списка. i-я строка списка состоит из i чисел 2, затем идет одно число 1, затем идет n-i-1 число 0:

for i in range(n):

A[i] = [2] * i + [1] + [0] * (n - i - 1)

```
n=3
A=[]
#заполняем массив 9-ми
for i in range(n):
    A.append([9]*n)
#вывод исходного массива
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()
#заменяем элементы главной диагонали, выше и ниже неё
for i in range(n):
    A[i] = [2] * i + [0] + [1] * (n - i - 1)
#вывод изменённого массива
print()
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()
```

```
9 9 9
9 9 9
9 9 9

0 1 1
2 0 1
2 2 0
>>>
```

Вариант 0

1. Дан двумерный массив размером 3x3. Определить максимальное значение среди элементов третьего столбца массива; максимальное значение среди элементов второй строки массива. Вывести полученные значения.

Решение:

```
n=3
a=[]
for i in range(n):
    b = []
    for j in range(n):
        print('Введите [',i,',',j,'] элемент')
        b.append(int(input()))
    a.append(b)
#вывод массива
for i in range(n):
    for j in range(n):
        print(a[i][j], end=' ')
    print()

#максимальное значение среди элементов третьего столбца
maximum=a[0][2]
for i in range(n):
    for j in range(n):
        if maximum<a[i][2]:
            maximum=a[i][2]
print('Максимальный в 3 столбце:',maximum)

#максимальное значение среди элементов второй строки
maximum=a[1][0]
for i in range(n):
    for j in range(n):
        if maximum<a[1][j]:
            maximum=a[1][j]
print('Максимальный во второй строке:',maximum)
```

```
Введите [ 0 , 0 ] элемент
1
Введите [ 0 , 1 ] элемент
2
Введите [ 0 , 2 ] элемент
3
Введите [ 1 , 0 ] элемент
4
Введите [ 1 , 1 ] элемент
5
Введите [ 1 , 2 ] элемент
6
Введите [ 2 , 0 ] элемент
7
Введите [ 2 , 1 ] элемент
8
Введите [ 2 , 2 ] элемент
9
1 2 3
4 5 6
7 8 9
Максимальный в 3 столбце: 9
Максимальный во второй строке: 6
```

2. Дан двумерный массив размером mxn. Сформировать новый массив заменив положительные элементы единицами, а отрицательные нулями. Вывести оба массива.

Решение:

```
m=int(input('Введите количество строк'))
n=int(input('Введите количство столбцов'))
a=[]
for i in range(m):
    b = []
    for j in range(n):
        print('Введите [',i,',',j,'] элемент')
        b.append(int(input()))
    a.append(b)
#вывод массива
print('Исходный массив:')
for i in range(m):
    for j in range(n):
        print(a[i][j], end=' ')
    print()

for i in range(m):
    for j in range(n):
        if a[i][j]<0: a[i][j]=0
        elif a[i][j]>0: a[i][j]=1

#вывод массива
print('Изменённый массив:')
for i in range(m):
    for j in range(n):
        print(a[i][j], end=' ')
    print()
```

```
Введите количество строк:3
Введите количество столбцов:4
Введите [ 0 , 0 ] элемент
-1
Введите [ 0 , 1 ] элемент
5
Введите [ 0 , 2 ] элемент
4
Введите [ 0 , 3 ] элемент
-5
Введите [ 1 , 0 ] элемент
-2
Введите [ 1 , 1 ] элемент
-1
Введите [ 1 , 2 ] элемент
0
Введите [ 1 , 3 ] элемент
4
Введите [ 2 , 0 ] элемент
-5
Введите [ 2 , 1 ] элемент
4
Введите [ 2 , 2 ] элемент
5
Введите [ 2 , 3 ] элемент
-5
Исходный массив:
-1 5 4 -5
-2 -1 0 4
-5 4 5 -5
Полученный массив:
0 1 1 0
0 0 0 1
0 1 1 0
```

Варианты заданий для самостоятельного выполнения:

Вариант 1

1. Вычислить сумму и число положительных элементов матрицы A[N, N], находящихся над главной диагональю.

2. Дана матрица B[N, M]. Найти в каждой строке матрицы максимальный и минимальный элементы и поменять их с первым и последним элементами строки соответственно.

Вариант 2

1. Дана целая квадратная матрица n-го порядка. Определить, является ли она магическим квадратом, т. е. такой матрицей, в которой суммы элементов во всех строках и столбцах одинаковы.

2. Дана прямоугольная матрица A[N, N]. Переставить первый и последний столбцы местами и вывести на экран.

Вариант 3.

1. Определить, является ли заданная целая квадратная матрица n-го порядка симметричной (относительно главной диагонали).

2. Дана вещественная матрица размером n x m. Переставляя ее строки и столбцы, добиться того, чтобы наибольший элемент (или один из них) оказался в верхнем левом углу.

Вариант 4.

1. Дана прямоугольная матрица. Найти строку с наибольшей и строку с наименьшей суммой элементов. Вывести на печать найденные строки и суммы их элементов.

2. Дана квадратная матрица A[N, N], Записать на место отрицательных элементов матрицы нули, а на место положительных — единицы. Вывести на печать нижнюю треугольную матрицу в общепринятом виде.

Вариант 5.

1. Упорядочить по возрастанию элементы каждой строки матрицы размером n x m.

2. Дана действительная матрица размером n x m, все элементы которой различны. В каждой строке выбирается элемент с наименьшим значением. Если число четное, то заменяется нулем, нечетное - единицей. Вывести на экран новую матрицу.

Вариант 6.

1. Дана целочисленная квадратная матрица. Найти в каждой строке наибольший элемент и в каждом столбце наименьший. Вывести на экран.

2. Дана действительная квадратная матрица порядка N (N — нечетное), все элементы которой различны. Найти наибольший элемент среди стоящих на главной и побочной диагоналях и поменять его местами с элементом, стоящим на пересечении этих диагоналей.

Вариант 7.

1. Квадратная матрица, симметричная относительно главной диагонали, задана верхним треугольником в виде одномерного массива. Восстановить исходную матрицу и напечатать по строкам.

2. Для заданной квадратной матрицы сформировать одномерный массив из ее диагональных элементов. Найти след матрицы, просуммировав элементы одномерного массива. Преобразовать исходную матрицу по правилу: четные строки разделить на полученное значение, нечетные оставить без изменения.

Вариант 8.

1. Задана матрица порядка n и число к. Разделить элементы k-й строки на диагональный элемент, расположенный в этой строке.

2. Задана квадратная матрица. Получить транспонированную матрицу (перевернутую относительно главной диагонали) и вывести на экран.

Вариант 9.

1. Для целочисленной квадратной матрицы найти число элементов, кратных k, и наибольший из этих элементов.

2. В данной действительной квадратной матрице порядка n найти наибольший по модулю элемент. Получить квадратную матрицу порядка n — 1 путем отбрасывания из исходной матрицы строки и столбца, на пересечении которых расположен элемент с найденным значением.

Вариант 10.

1. Найти максимальный среди всех элементов тех строк заданной матрицы, которые упорядочены (либо по возрастанию, либо по убыванию).

2. Расположить столбцы матрицы D[M, N] в порядке возрастания элементов k-й строки (1 <= k <= M).

Вариант 11.

1. В данной действительной квадратной матрице порядка n найти сумму элементов строки, в которой расположен элемент с наименьшим значением. Предполагается, что такой элемент единственный.

2. Среди столбцов заданной целочисленной матрицы, содержащих только такие элементы, которые по модулю не больше 10, найти столбец с минимальным произведением элементов и поменять местами с соседним.

Вариант 12.

1. Для заданной квадратной матрицы найти такие k, что k-я строка матрицы совпадает с k-м столбцом.

2. Дана действительная матрица размером n x m. Требуется преобразовать матрицу: поэлементно вычесть последнюю строку из всех строк, кроме последней.

Вариант 13.

1. Определить наименьший элемент каждой четной строки матрицы A[M, N].

2. Найти наибольший и наименьший элементы прямоугольной матрицы и поменять их местами.

Вариант 14.

1. Задана квадратная матрица. Переставить строку с максимальным элементом на главной диагонали со строкой с заданным номером m.

2. Составить программу, которая заполняет квадратную матрицу порядка п натуральными числами 1, 2, 3, ..., n2, записывая их в нее «по спирали».

Например, для п = 5 получаем следующую матрицу:

1 2 3 4 5

16 17 18 19 6

15 24 25 20 7

14 23 22 21 8

14 12 11 10 9

Вариант 15.

1. Определить номера строк матрицы R[M, N], хотя бы один элемент которых равен c, и элементы этих строк умножить на d.

2. Среди тех строк целочисленной матрицы, которые содержат только нечетные элементы, найти строку с максимальной суммой модулей элементов.

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия;
2. Цель занятия;
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»;
4. Список используемых источников;
5. Выводы и предложения;
6. Дата и подпись курсанта и преподавателя.

Вопросы для самопроверки

1. Дайте определение понятию «матрицы».
2. Как создаются списки?
3. Как осуществляется вывод вложенного списка?

4. Как создать вложенный список (синтаксис)?

Практическое занятие №32 Функции и процедуры в Python

Цель занятия:

1. Изучение процедур и функций в Python.

2. Знать - синтаксис процедур и функций, процедура с параметром, локальные и глобальные переменные;

3. Уметь - применять синтаксис процедур и функций при составлении программы;

4. Владеть - основными навыками работы с функциями и процедурами.

Исходные данные: теоретический материал, программа Python

Содержание и порядок выполнения:

1. Изучить теоретическую часть;

2. Выполнить задания.

Теоретическая часть

Подпрограмма - это именованный фрагмент программы, к которому можно обратиться из другого места программы.

Подпрограммы делятся на две категории: процедуры и функции.

1. Процедуры.

Рассмотрим синтаксис процедуры:

def имя процедуры(Список параметров):

Система команд

Для определения процедуры используется ключевое слово def, затем указывается имя процедуры и в скобках её формальные параметры, если они присутствуют. После ставится двоеточие и со следующей строки с отступом в 4 пробела указываются команды.

Процедура — вспомогательный алгоритм, выполняющий некоторые действия.

Процедура должна быть определена к моменту её вызова. Определение процедуры начинается со служебного слова def.

Вызов процедуры осуществляется по ее имени, за которым следуют круглые скобки, например, Err().

В одной программе может быть сколько угодно много вызовов одной и той же процедуры.

Использование процедур сокращает код и повышает удобочитаемость.

Процедура с параметрами.

Как используются в Python параметры процедуры, рассмотрим на примере.

Пример.

Написать процедуру, которая печатает раз указанный символ (введенный с клавиатуры), каждый с новой строки.

```
def printChar(s):
   print (s)
sim = input('введите символ')
printChar(sim) # первый вызов, вывод введенного символа
printChar('*') # второй вызов, вывод *
```

```
def printChar(s):
    print (s)
sim = input('введите символ: ')
printChar(sim) # первый вызов, вывод введенного символа
printChar('*') # второй вызов, вывод *
```

```
>>>
введите символ: 41
41
*
```

Глобальная переменная — если ей присвоено значение в основной программе (вне процедуры).

Локальная переменная (внутренняя) известна только на уровне процедуры, обратиться к ней из основной программы и из других процедур нельзя.

Параметры процедуры — локальные переменные.

2. Примеры использования локальных и глобальных переменных.

Пример 1.

```
x = 3 # глобальная переменная
defpr(): # процедура без параметров
  print (x) # вывод значения глобальной переменной
pr()
```

```
x = 3 # глобальная переменная
def pr(): # процедура без параметров
  print (x) # вывод значения глобальной переменной
pr()
```

```
Python 3.4.1 Shell
File  Edit  Shell  Debug  Options  Windows  Help
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10
tel)] on win32
Type "copyright", "credits" or "license()" for mor
>>> ================================ RESTART =====
>>>
3
>>>
```

Пример 2.

x = 3 # глобальная переменная

defpr(a): # процедура с параметром

  print (a) # 4

pr(x) # передача параметра глобальной переменной (3)

```
Python 3.4.1: ex_procedure3.py - C:/Documents and Settings/St
File  Edit  Format  Run  Options  Windows  Help
x = 10  # глобальная переменная
def pr(a): # процедура с параметром
  print (a) # 4
pr(x) # передача параметра глобальной переменной (3)
```

```
Python 3.4.1 Shell
File  Edit  Shell  Debug  Options  Windows  Help
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:3
tel)] on win32
Type "copyright", "credits" or "license()" for more
>>> ================================ RESTART =======
>>>
10
>>>
```

Существует возможность изменить значение глобальной переменной (не создавая локальную). В процедуре с помощью слова global:

x = 3 # глобальная переменная

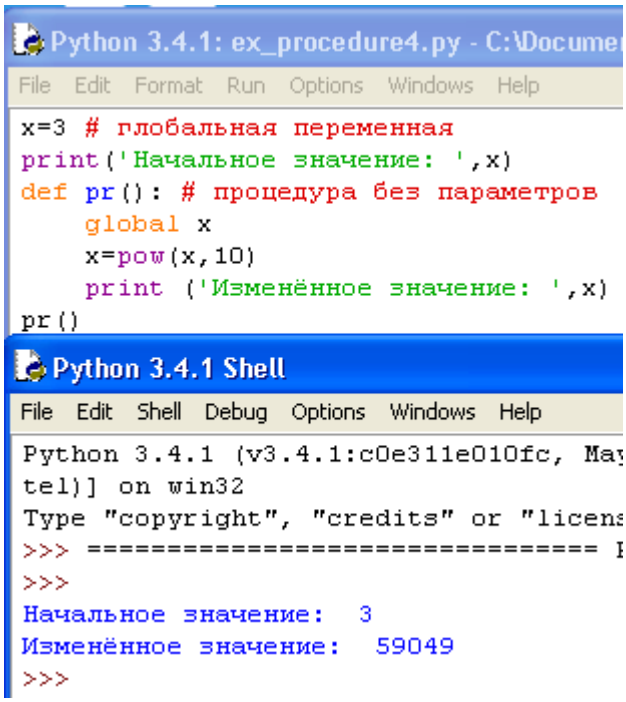
defpr(): # процедура без параметров

  global x

  x = pow(x,10)

  print (x) # вывод измененного значения глобальной переменной

pr()



3. Функции.

Функция - подпрограмма, к которому можно обратиться из другого места программы.

Для создания функции используется ключевое слово def, после которого указывается имя и список аргументов в круглых скобках. Тело функции выделяется также как тело условия (или цикла): четырьмя пробелами.

Рассмотрим синтаксис функции:

```
def имя функции(Список параметров):
    Система команд
    return выражение
```

Часть функций языка Python являются встроенными функциями, которые обеспечены синтаксисом самого языка. Например, int, input, randint.

Рассмотрим пример создания пользовательских функций.

Пример 1.

Вычислить сумму цифр числа.

```
defsumD(n): # определение функции с параметром
    sumD = 0
    while n!= 0:
        sumD += n % 10
        n = n // 10
```

returnsumD # возврат значения функции

основная программа

print (sumD(int(input())) # вызов функции с параметром

```
def sumD(n): # определение функции с параметром
    summa = 0
    while n!= 0:
        summa += n % 10
        n=n//10
    return summa # возврат значения функции
# основная программа
print (sumD(int(input()))) # вызов функции с параметром
```

```
Python 3.4.1 Shell
File  Edit  Shell  Debug  Options  Windows  Help
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22
tel)] on win32
Type "copyright", "credits" or "license()" for more infc
>>> ================================ RESTART ===========
>>>
123456789
45
>>>
```

Вариант 0.

1. Определить, являются ли три треугольника равновеликими. Длины сторон вводить с клавиатуры. Для подсчёта площади треугольника использовать формулу Герона. Вычисление площади оформить в виде функции с тремя параметрами.

Формула Герона:

$$S = \sqrt{p(p-a)(p-b)(p-c)},$$

$$где \quad p = \frac{a+b+c}{2}$$

Решение:

```
import math
def s(x,y,z):
    p=(x+y+z)/2
    s=math.sqrt(p*(p-x)*(p-y)*(p-z))
    return s
A=[]
for i in range(3):
    print('Введите стороны ',i,'-го треугольника:')
    a=int(input('a:'))
    b=int(input('b:'))
    c=int(input('c:'))
    A.append(s(a,b,c))
for i in range(3):
    print('Площадь ',i,'-го треугольника {:.2f}'.format(A[i]))
if A[0]==A[1]:
    if A[0]==A[2]:
        print('Треугольники равновеликие')
else: print('Треугольники не равновеликие')
```

```
Введите стороны  0 -го треугольника:
a:3
b:4
c:5
Введите стороны  1 -го треугольника:
a:6
b:7
c:8
Введите стороны  2 -го треугольника:
a:9
b:10
c:11
Площадь  0 -го треугольника 6.00
Площадь  1 -го треугольника 20.33
Площадь  2 -го треугольника 42.43
Треугольники не равновеликие
```

2. Ввести одномерный массив A длиной m. Поменять в нём местами первый и последний элементы. Длину массива и его элементы ввести с клавиатуры. В программе описать процедуру для замены элементов массива. Вывести исходные и полученные массивы.

Решение:

```
def zam(X):
    tmp=X[0]
    X[0]=X[len(X)-1]
    X[len(X)-1]=tmp
A=[]
m=int(input('Введите длину массива:'))
for i in range(m):
    print('Введите ',i,'элемент массива')
    A.append(int(input()))
print(A)
zam(A)
print(A)
```

```
Введите длину массива:5
Введите  0 элемент массива
0
Введите  1 элемент массива
1
Введите  2 элемент массива
2
Введите  3 элемент массива
3
Введите  4 элемент массива
4
[0, 1, 2, 3, 4]
[4, 1, 2, 3, 0]
```

Варианты заданий для самостоятельного выполнения

Вариант 1.

1. Составить программу для вычисления площади разных геометрических фигур.

2. Даны 3 различных массива целых чисел (размер каждого не превышает 15). В каждом массиве найти сумму элементов и среднеарифметическое значение.

Вариант 2.

1. Вычислить площадь правильного шестиугольника со стороной а, используя подпрограмму вычисления площади треугольника.

2. Пользователь вводит две стороны трех прямоугольников. Вывести их площади.

Вариант 3.

1. Даны катеты двух прямоугольных треугольников. Написать функцию вычисления длины гипотенузы этих треугольников. Сравнить и вывести какая из гипотенуз больше, а какая меньше.

2. Преобразовать строку так, чтобы буквы каждого слова в ней были отсортированы по алфавиту.

Вариант 4.

1. Даны две дроби A/B и C/D (A, B, C, D — натуральные числа). Составить программу деления дроби на дробь. Ответ должен быть несократимой дробью. Использовать подпрограмму алгоритма Евклида для определения НОД.

2. Задана окружность (x-a)2 + (y-b)2 = R2 и точки P(p1, p2), F(f1, f1), L(l1,l2). Выяснить и вывести на экран, сколько точек лежит внутри окружности. Проверку, лежит ли точка внутри окружности, оформить в виде процедуры.

Вариант 5.

1. Даны две дроби A/B и C/D (A, B, C, D — натуральные числа). Составить программу вычитания из первой дроби второй. Ответ должен быть несократимой дробью. Использовать подпрограмму алгоритма Евклида для определения НОД.

2. Напишите программу, которая выводит в одну строчку все делители переданного ей числа, разделяя их пробелами.

Вариант 6.

1. Составить программу нахождения наибольшего общего делителя (НОД) и наименьшего общего кратного (НОК) двух натуральных чисел НОК(A, B) = (A*B)/НОД(A,B). Использовать подпрограмму алгоритма Евклида для определения НОД.

2. Составить программу вычисления площади выпуклого четырехугольника, заданного длинами четырех сторон и диагонали.

Вариант 7.

1. Даны числа X, Y, Z, T — длины сторон четырехугольника. Вычислить его площадь, если угол между сторонами длиной X и У — прямой. Использовать две подпрограммы для вычисления площадей: прямоугольного треугольника и прямоугольника.

2. Напишите программу, которая переводит переданное ей неотрицательное целое число в 10-значный восьмеричный код, сохранив лидирующие нули.

Вариант 8.

1. Найти все натуральные числа, не превосходящие заданного n, которые делятся на каждую из своих цифр.

2. Ввести одномерный массив A длиной m. Поменять в нём местами первый и последний элементы. Длину массива и его элементы ввести с клавиатуры. В программе описать процедуру для замены элементов массива. Вывести исходные и полученные массивы.

Вариант 9.

1. Из заданного числа вычли сумму его цифр. Из результата вновь вычли сумму его цифр и т. д. Через сколько таких действий получится нуль?

2. Даны 3 различных массива целых чисел. В каждом массиве найти произведение элементов и среднеарифметическое значение.

Вариант 10.

1. На отрезке [100, N] (210 < N< 231) найти количество чисел, составленных из цифр a, b, c.

2. Составить программу, которая изменяет последовательность слов в строке на обратную.

Вариант 11.

1. Два простых числа называются «близнецами», если они отличаются друг от друга на 2 (например, 41 и 43). Напечатать все пары «близнецов» из отрезка [n, 2n], где n — заданное натуральное число, большее 2.

2. Даны две матрицы A и B. Написать программу, меняющую местами максимальные элементы этих матриц. Нахождение максимального элемента матрицы оформить в виде процедуры.

Вариант 12.

1. Два натуральных числа называются «дружественными», если каждое из них равно сумме всех делителей (кроме его самого) другого (например, числа 220 и 284). Найти все пары «дружественных» чисел, которые не больше данного числа N.

2. Даны длины сторон треугольника a, b, c. Найти медианы треугольника, сторонами которого являются медианы исходного треугольника. Для вычисления медианы проведенной к стороне a, использовать формулу Вычисление медианы оформить в виде процедуры.

Вариант 13.

1. Натуральное число, в записи которого n цифр, называется числом Армстронга, если сумма его цифр, возведенная в степень n, равна самому числу. Найти все числа Армстронга от 1 до к.

2. Три точки заданы своими координатами X(x1, x2), Y(y1, y2) и Z(z1, z2). Найти и напечатать координаты точки, для которой угол между осью абсцисс и лучом, соединяющим начало координат с точкой, минимальный. Вычисления оформить в виде процедуры.

Вариант 14.

1. Составить программу для нахождения чисел из интервала [M, N], имеющих наибольшее количество делителей.

2.Четыре точки заданы своими координатами X(x1, x2), Y(y1, y2), Z(z1, z2), P(p1, p2). Выяснить, какие из них находятся на максимальном расстоянии друг от друга и вывести на экран значение этого расстояния. Вычисление расстояния между двумя точками оформить в виде процедуры.

Вариант 15.

1. Найти все простые натуральные числа, не превосходящие n, двоичная запись которых представляет собой палиндром, т. е. читается одинаково слева направо и справа налево.

2. Четыре точки заданы своими координатами X(x1, x2, x3), Y(y1, y2, y3), Z(z1, z2, z3), T(t1,t2, t3). Выяснить, какие из них находятся на минимальном расстоянии друг от друга и вывести на экран значение этого расстояния. Вычисление расстояния между двумя точками оформить в виде процедуры.

Выводы и предложения о проделанной работе

Содержание отчета:

1. Наименование практического занятия;
2. Цель занятия;
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»;
4. Список используемых источников;
5. Выводы и предложения;
6. Дата и подпись курсанта и преподавателя.

Вопросы для самопроверки

1. Дайте определение понятию «подпрограмма».
2. На какие категории делятся подпрограммы?
3. Дайте определение понятию «процедура».
4. Что такое глобальная и локальная переменные?
5. Что такое функции. Синтаксис функции.

Практическое занятие №33 Дополнительные типы данных в Python

Цель занятия:

1. Познакомиться с понятиями множества, кортежи, словари

Исходные данные: теоретический материал, программа

Содержание и порядок выполнения:

1. Изучить теоретическую часть;
2. Выполнить задания.

Теоретическая часть

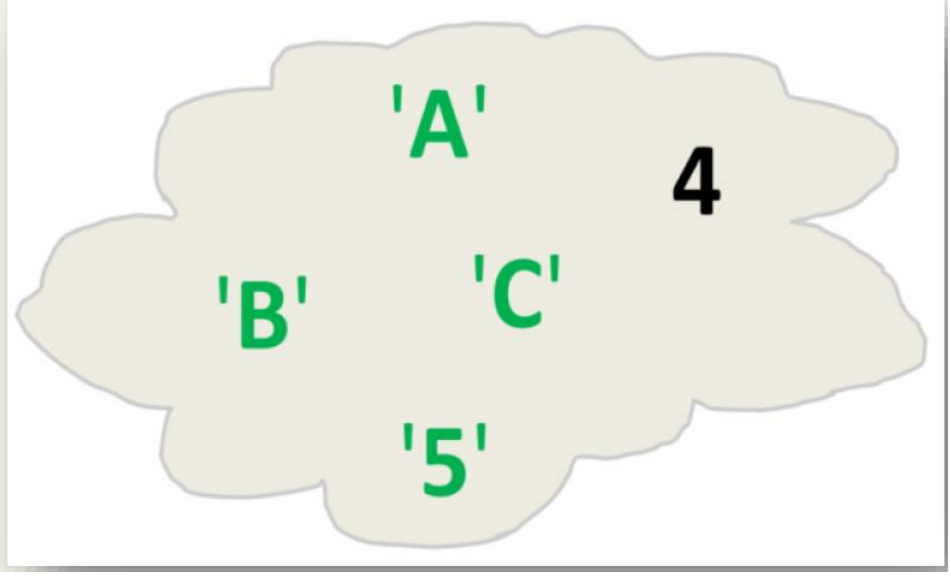
1 Множества

Математическое образование разработчика языка Python наложило свой отпечаток на типы данных (классы), которые присутствуют в языке.

Рассмотрим множество (set) в Python - неупорядоченную коллекцию неизменяемых, уникальных элементов. Создадим множество:

```
>>> v = {'A', 'C', 4, '5', 'B'}
>>> v
{'C', 'B', '5', 4, 'A'}
>>>
```

Заметим, что полученное множество отобразилось не в том порядке, в каком мы его создавали, т.к. множество - это неупорядоченная коллекция. Представим множество схематично:



Множества в Python обладают интересными свойствами:

```
>>> v = {'A', 'C', 4, '5', 'B', 4}
>>> v
{'C', 'B', '5', 4, 'A'}
>>>
```

Видим, что повторяющиеся элементы, которые мы добавили при создании множества, были удалены (элементы множества уникальны).

Рассмотрим способы создания множеств:

```
>>> set([3, 6, 3, 5])
{3, 5, 6}
>>>
```

Множества можно создавать на основе списков. Обратите внимание, что в момент создания множества из списка будут удалены повторяющиеся элементы. Это отличный способ очистить список от повторов:

```
>>> list(set([3, 6, 3, 5]))
```

2. Кортежи

Следующий тип данных (класс), который также уходит своими корнями в математику - кортеж (tuple). Кортеж условно можно назвать неизменяемым «списком», т.к. к нему применимы многие списковые функции, кроме изменения. Кортежи используются, когда мы хотим быть уверены, что элементы структуры данных не будут изменены в процессе работы программы. Вспомните проблему с псевдонимами у списков.

Некоторые операции над кортежами:

```
>>> ()      #  создание пустого кортежа
()
>>> (4)     # это не кортеж, а целочисленный объект!
4
>>> (4,)    # а вот это - кортеж, состоящий из одного элемента!
(4,)
>>> b = ('1', 2, '4')    # создаем кортеж
>>> b
('1', 2, '4')
>>> len(b)  # определяем длину кортежа
3
>>> t = tuple(range(10)) # создание кортежа с помощью функции range()
>>> t + b   # слияние кортежей
(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, '1', 2, '4')
>>> r = tuple([1, 5, 6, 7, 8, '1'])    # кортеж из списка
>>> r
(1, 5, 6, 7, 8, '1')
>>>
```

С помощью кортежей можно присваивать значения одновременно двум переменным:

```
>>> (x, y) = (10, 5)
>>> x
10
>>> y
5
>>> x, y = 1, 3 # если убрать круглые скобки, то результат не изменится
>>> x
1
>>> y
3
>>>
```

Поменять местами содержимое двух переменных:

```
>>> x, y = y, x
>>> x
3
>>> y
1
>>>
```

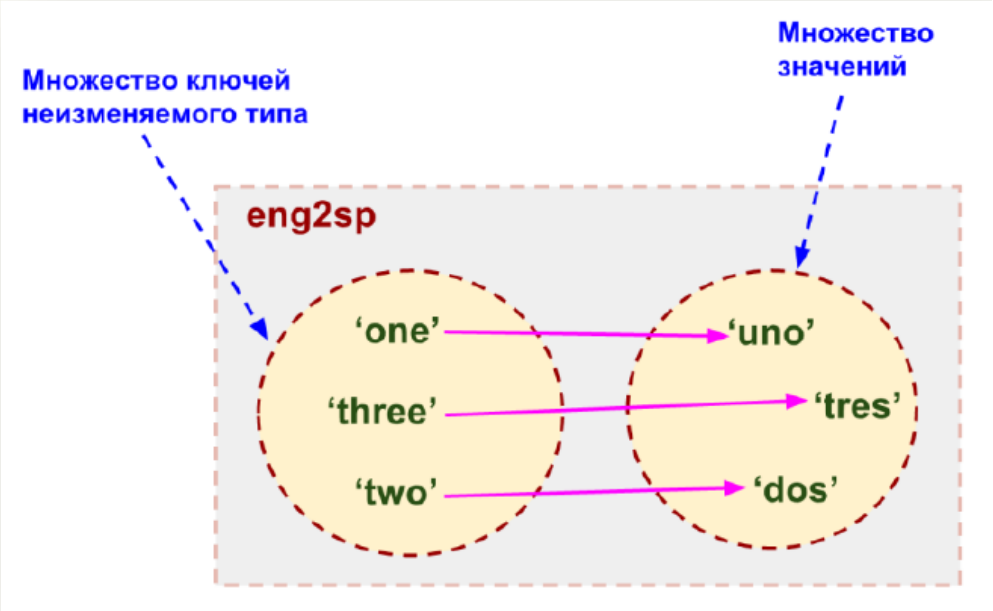
Мы сказали, что кортеж нельзя изменить, но можно изменить, например, список, входящий в кортеж:

```
>>> t = (1, [1, 3], '3')
>>> t[1]
[1, 3]
>>> t[1][0] = '1'
>>> t
(1, ['1', 3], '3')
>>>
```

3. Словари

Следующий тип данных (класс) - словарь (diet). Словарь в Python неупорядоченная изменяемая коллекция или, проще говоря, «список» с произвольными ключами, неизменяемого типа.

Пример создания словаря, который каждому слову на английском языка будет ставить в соответствие слово на испанском языке.




```
>>> eng2sp = dict()    # создаем пустой словарь
>>> eng2sp
{}
>>> eng2sp['one'] = 'uno' # добавляем 'uno' для элемента с индексом 'one'
>>> eng2sp
{'one': 'uno'}
>>> eng2sp['one']
'uno'
>>> eng2sp['two'] = 'dos'
>>> eng2sp['three'] = 'tres'
>>> eng2sp
{'three': 'tres', 'one': 'uno', 'two': 'dos'}
>>>
```

В качестве индексов словаря используются неизменяемые строки, могли бы воспользоваться кортежами, т.к. они тоже неизменяемые:

```
>>> e = {}
>>> e
{}
>>> e[(4, '6')] = '1'
>>> e
{(4, '6'): '1'}
>>>
```

Результирующий словарь e n g 2 s p отобразился в «перемешанном» виде, т.к. по аналогии с множествами, словари являются неупорядоченной коллекцией.

К словарям применим оператор i n :

```
>>> eng2sp
{'three': 'tres', 'one': 'uno', 'two': 'dos'}
>>> 'one' in eng2sp   # поиск по множеству КЛЮЧЕЙ
True
>>>
```

Часто словари используются, если требуется найти частоту встречаемости элементов в **последовательности (списке, строке, кортеже).**

Функция, которая возвращает словарь, содержащий статистику встречаемости элементов в последовательности:

```
def histogram(s):
    d = dict()
    for c in s:
        if c not in d:
            d[c] = 1
        else:
            d[c] = d[c] + 1 # или d[c] += 1
    return d
```

Результат вызова функции histogram для списка, строки, кортежа соответственно:

```
>>> histogram([2, 5, 6, 5, 4, 4, 4, 4, 3, 2, 2, 2, 2])
{2: 5, 3: 1, 4: 4, 5: 2, 6: 1}
>>> histogram("ywte3475eryt3478e477477474")
{'4': 6, '8': 1, 'e': 3, '3': 2, '7': 7, '5': 1, 'r': 1, 'y': 2, 'w':
1, 't': 2}
>>> histogram((5, 5, 5, 6, 5, 'r', 5))
{5: 5, 6: 1, 'r': 1}
>>>
```

Задание Сделать русско-английский словарь

В файле task6/en-ru.txt находятся строки англо-русского словаря в таком формате:

cat - кошка

dog - собака

home - домашняя папка, дом

mouse - мышь, манипулятор мышь

todo - делать, изготавливать

tomake - изготавливать

Здесь английское слово (выражение) и список русских слов (выражений) разделены двумя табуляциями и минусом: '\t-\t'.

Требуется создать русско-английский словарь и вывести его в файл ru-en.txt в таком формате:

делать - todo

дом - home

домашняя папка - home

изготавливать - to do, to make

кошка - cat

манипуляторомышь - mouse

мышь - mouse

собака - dog

Порядок строк в выходном файле должен быть словарным с *человеческой* точки зрения (так называемый *лексикографический* порядок слов). То есть выходные строки нужно отсортировать.

Выводы и предложения о проделанной работе

Содержаниеотчета:

1. Наименование практического занятия;

2. Цель занятия;
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»;
4. Список используемых источников;
5. Выводы и предложения;
6. Дата и подпись курсанта и преподавателя.

Вопросы для самопроверки

1. Дайте определение понятию «множества».
2. Дайте определение понятию «кортежи».
3. Дайте определение понятию «словари».

Практическое занятие №34Знакомство с графикой в Python

Цель занятия:

1. Познакомиться с понятиями графики

Исходные данные: теоретический материал, программа

Содержание и порядок выполнения:

1. Изучить теоретическую часть;
2. Выполнить задания.

Теоретическая часть

С помощью **графики в Python** можно рисовать фигуры и изображения, создавать анимацию, визуализировать математические вычисления в Python. В программах python можно использовать элементы графики в компьютерных играх.

Для работы с **графикой в Python** нужно импортировать модуль graphics.py.

Чтобы начать работу с **графикой в Python**, нужно создать окно для графики

Графический объект = GraphWin("Название окна для графики", ширина окна для графики в пикселях, высота окна для графики в пикселях)

GraphWin это ключевое слово, которое задаёт окно графической области, в котором будут отображаться графические объекты.

В качестве параметров этой функции указывается название окон для графики, ширина и высота окон в пикселях.

После запуска программы откроется окно для графики, где будут отображаться графические объекты.

Вся работа с графикой будет осуществляться нами через графические объекты.

Общая структура работы с графическими объектами в Python

Графический_объект.Вызов_команды()

Общая структура графической программы в Python.

```
# импортируем библиотеку graphics
fromgraphicsimport *
# создаём окно для графики
win = GraphWin("Окно для графики", 400, 400)
# ...рисуем все объекты...
win.getMouse() # ждём нажатия кнопки мыши
win.close() # закрываем окно для графики
```

В этой программе мы определили объект графическое окно win и открыли его с размерами 400 на 400 пикселей.

Команда win.getMouse() ожидает нажатие на любую кнопку мыши, наведённую на область окна win.

win.close() закрывает окно для графических объектов win.

С помощью модуля graphics.py в программах на Python можно отобразить точку, линию, окружность, прямоугольник, эллипс и многоугольник, вывести текст на экран.

Чтобы задать расположение объекта в графическом окне Python, необходимо указать его координаты в системе координат Python. Начало координат находится в левом верхнем углу окна для графики.

Положительное направление оси X определяется слева направо, оси Y определяется сверху вниз. Чем больше значение координаты X, тем правее точка, чем больше значение координаты Y, тем точка ниже. Чтобы нарисовать заданный объект obj в окне для графики win, нужно использовать процедуру obj.draw(win)

Перед тем, как рисовать графические объекты в заданном графическом окне, нужно их задать.

Для задания точки в Python используется функция Point(x, y)

obj = Point(x, y)

x, y – координаты точки.

Пример программы на Python, которая задаёт и отображает точку в графическом окне.

```
fromgraphicsimport * # импортируем библиотеку graphics
win = GraphWin("Окно для графики", 400, 400) # создаём окно для графики размером 400 на 400 пикселей
obj = Point(50, 50) # создаём точку в координатах (50, 50)
obj.draw(win) # отображаем точку в окне для графики
win.getMouse() # ждём нажатия кнопки мыши
win.close() # закрываем окно для графики
```

Для задания отрезка в Python используется функция Line(объект точка первого конца, объект точка второго конца)

```
obj = Line(Point(x1, y1), Point(x2, y2))
```

x1, y1 – координаты начала отрезка линии, x2, y2 – координаты конца отрезка линии.

Чтобы задать цвет рисования линий в Python используется команда obj.setOutline("цвет")

Пример программы на Python, которая отображает линию в графическом окне.

```
from graphics import *
win = GraphWin("Окнодляграфики", 400, 400)
obj = Line(Point(50, 50), Point(350, 350))
obj.setOutline("blue")
obj.draw(win)
win.getMouse()
win.close()
```

Для отображения окружности в Python используется

```
obj = Circle(Point(x, y), R)
```

x, y – координаты центра окружности,

R – радиус окружности.

Пример программы на Python, которая отображает окружность в графическом окне.

```
from graphics import *
win = GraphWin("Окнодляграфики", 400, 400)
obj = Circle(Point(200, 200), 50)
obj.draw(win)
win.getMouse()
win.close()
```

Дляотображенияпрямоугольника вPython используетсяпроцедура

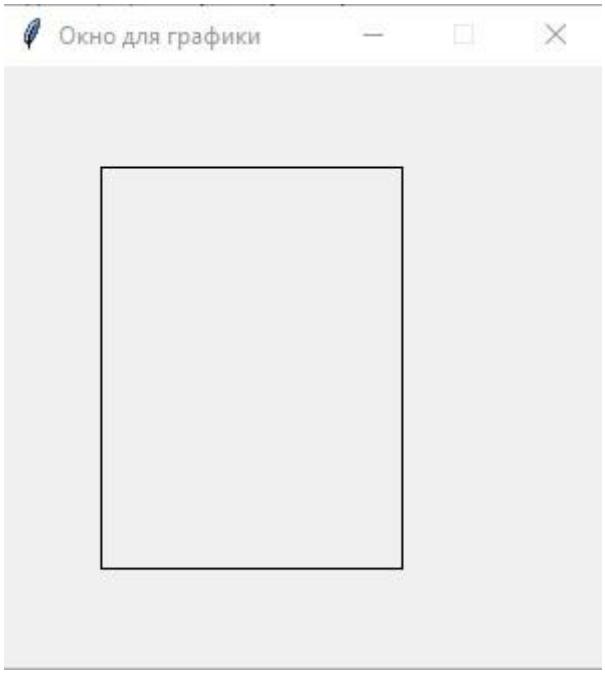
obj = Rectangle(Point(x1, y1), Point(x2, y2))

x1, y1 – координатылевоговерхнегоуглапрямоугольника,

x2, y2 – координатыправогонижнегоуглапрямоугольника

ПримерпрограммынаPython, котораяотображаетпрямоугольниквграфическомокне.

```
from graphics import *
win = GraphWin("Окнодляграфики", 300, 300)
obj = Rectangle(Point(50, 50), Point(200, 250))
obj.draw(win)
win.getMouse()
win.close()
```



Для отображения эллипса в Python используется процедура

obj = Oval(Point(x1, y1), Point(x2, y2))

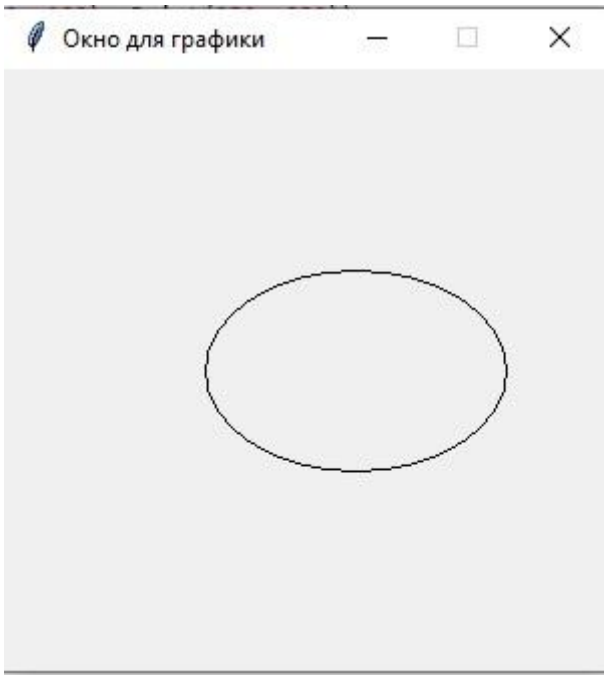
x1, y1 – координаты первого фокуса эллипса,

x2, y2 – координаты второго фокуса эллипса.

Пример программы на Python, которая отображает эллипс в графическом окне.

```
from graphics import *
win = GraphWin("Окнодляграфики", 300, 300)
obj = Oval(Point(100, 100), Point(250, 200))
```

```
obj.draw(win)
win.getMouse()
win.close()
```



Для отображения многоугольника в Python используется процедура

obj = Polygon(Point(x1, y1), Point(x2, y2),…, Point(xn, yn))

x1, y1, x2, y2,…, xn, yn – координаты вершин многоугольника.

Пример программы на Python, которая отображает пятиугольник в графическом окне.

```
from graphics import *
win = GraphWin("Окнодляграфики", 400, 400)
obj = Polygon(Point(10, 10), Point(300, 50), Point(200, 300), Point(150, 150), Point(70, 70))
obj.draw(win)
win.getMouse()
win.close()
```

Определение цвета закрашивания графического объекта в Python

Чтобы задать цвет закрашивания графического объекта в python используется команда obj.setFill("цвет")

Пример программы на Python, которая рисует закрашенную синюю окружность

```
from graphics import *
win = GraphWin("Окнодляграфики", 400, 400)
obj = Circle(Point(200, 200), 50)
obj.setFill("blue")
obj.draw(win)
win.getMouse()
win.close()
```

Для редактирования границ объектов в Python используются процедуры setOutline(“цвет границы”) иsetWidth(ширина границы).

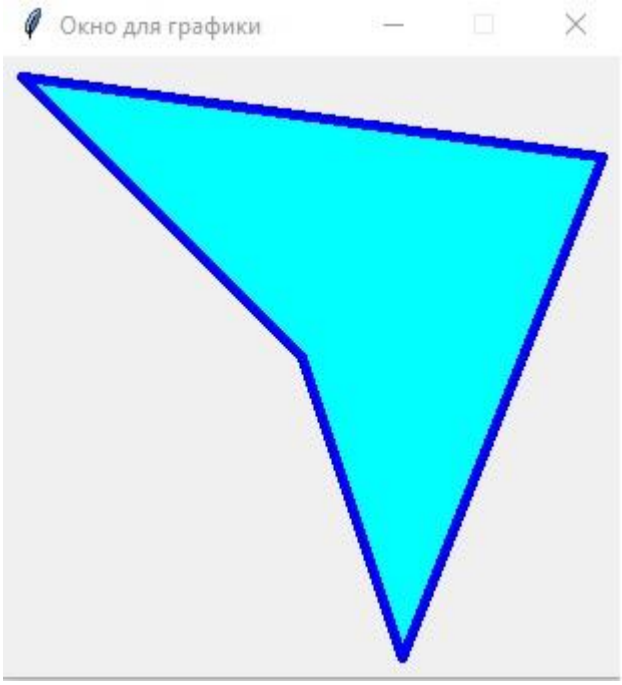
obj.setOutline("blue") – объект obj отображается с границей синего цвета.

obj.setWidth(5) – объект obj отображается с шириной границы 5 пикселей.

По умолчанию графический объект в Python будет отображаться с границами чёрного цвета шириной 1 пиксель.

Пример программы на Python, которая отображает фигуру с синей границей и заливкой в графическом окне.

```
from graphics import *
win = GraphWin("Окнодляграфики", 310, 310)
obj = Polygon(Point(10, 10), Point(300, 50), Point(200, 300), Point(150, 150), Point(70,
70))
obj.setOutline("blue")
obj.setWidth(5)
obj.setFill("cyan")
obj.draw(win)
win.getMouse()
win.close()
```



Чтобы переместить графический объект в Python, используется процедура move(dx, dy), которая перемещает объект на dx пикселей вправо и dy пикселей вниз.

obj.move(50, 50) смещает объект obj на 50 пикселей вправо и 50 пикселей вниз.

Для клонирования объектов используется процедура clone()

newObj = obj.clone()

С помощью этой команды создаётся новый графический объект newObj, который идентичен объекту obj.

Для удаления фигур с экрана используется процедура undraw() Объект удаляется с графического окна, но не удаляется из памяти.

obj.undraw()

Пример программы на Python, которая удаляет, перемещает и копирует объект в графическом окне.

```
fromgraphicsimport *
win = GraphWin("Окно для графики", 400, 400)
obj = Polygon(Point(30, 10), Point(30, 50), Point(20, 30), Point(15, 30), Point(7, 7))
obj.setOutline("blue")
obj.setWidth(2)
obj.setFill("cyan")
obj.draw(win)
```

```
win.getMouse()
obj.undraw()
win.getMouse()
obj.draw(win)
obj.move(100, 100)
win.getMouse()
shape = obj.clone()
shape.move(-100, -100)
shape.draw(win)
win.getMouse()
win.close()
```

Для создания текста в графическом окне в Python используется команда

текстовый объект=Text(координаты точки размещения текста, “Текст”)

msg = Text(Point(50, 100), “Hello World!”)

На экран в точке с координатами (50, 100) выведется текст со строкой Hello World!

Для изменения размера текста используется команда текстовый объект.setSize(размер текста)

msg.setSize(12)

Цвет текста изменяется с помощью метода setTextColor(цвет)

msg.setTextColor(“black”)

Текст в графическом объекте можно заменить с помощью метода setText(“Текст”)

msg.setText(“Другой текст”)

Стиль текста изменяется с помощью процедуры setStyle(стиль)

msg.setStyle(“bold”)

Стиль normal изменяет стиль текста на обычный, bold меняет стиль на полужирный, italic меняет стиль на курсив, bolditalic меняет стиль текста на полужирный курсив.

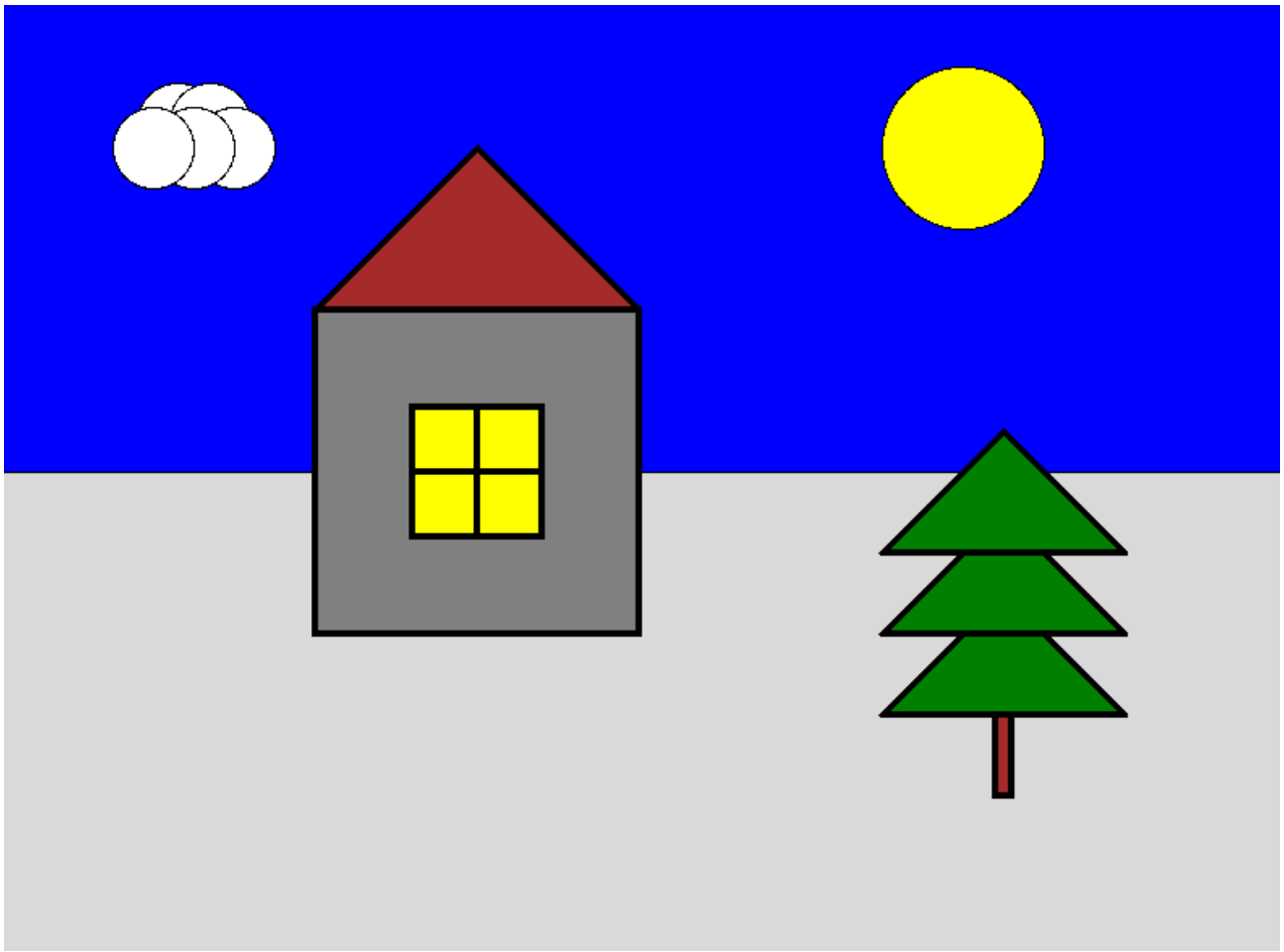
Пример программы на Python, которая отображает текст в графическом окне.

```
fromgraphicsimport *
win = GraphWin("Окно для графики", 400, 400)
```

```
obj = Polygon(Point(10, 10), Point(300, 50), Point(200, 300), Point(150, 150), Point(70,
70))
obj.setOutline("blue")
obj.setWidth(5)
obj.setFill("cyan")
obj.draw(win)
win.getMouse()
obj.undraw()
msg = Text(Point(200, 200), "Фигураудалиласьсэкрана")
msg.setSize(12)
msg.setTextColor("black")
msg.setStyle("bold italic")
msg.draw(win)
win.getMouse()
win.close()
```

Задание №1. Пейзаж

Используя полученные знания, нарисуйте любую статическую сцену, которая содержит не менее 5 различных объектов, состоящих из пяти и более примитивов. Проявите свою творческую натуру. Примером сцены может являться следующая картинка:



Задание №2 Измените вашу сцену так, чтобы объекты были нарисованы на пейзаже в других местах. Добавьте ещё два таких же облака, но так, чтобы все три облака выглядели естественно, не выстроившись в линейку.

Функции с параметрами

А теперь представьте, что в предыдущемзадании вас попросили сделать не две копии, а сто?

Наивным решением будет написать сто почти одинаковых функций с измененными цифрами, но если мы вдруг внезапно захотим во всех этих объектах убрать какой-либо примитив — нам придется залезть в каждую такую функцию и изменить соответствующие строчки. Такой подход нежизнеспособен.

Рациональным выходом из подобной ситуации будет являться использование функций с параметрами. В физике положение объекта мы задавали с помощью координат, почему бы такой подход не распространить и на графические объекты?

Продемонстрируем, как этот код можно оптимизировать.

```
defdraw_eye(x,y,size):
eye=gr.Circle(gr.Point(x,y),size)
pupil=gr.Circle(gr.Point(x,y),size/2)
```

```
eye.setFill('red')
pupil.setFill('black')

eye.draw(window)
pupil.draw(window)

defdraw_angry_lecturer():
draw_face()
draw_eye(150,180,20)
draw_eye(250,180,14)
draw_eyebrows()
draw_mouth()
```

Как видите теперь, если мы хотим изменить конструкцию обоих глаз одновременно, нам достаточно изменить код в одном месте, и это изменение распространиться на все объекты, которые отрисовываются данной функцией.

Выводы и предложения о проделанной работе

Содержаниеотчета:

1. Наименование практического занятия;
2. Цель занятия;
3. Отчет о выполнении на каждый этап раздела «Содержание и порядок выполнения задания»;
4. Список используемых источников;
5. Выводы и предложения;
6. Дата и подпись курсанта и преподавателя.

Вопросы для самопроверки

1. Что делает графический объект GraphWin?
2. Какая процедура используется для рисования прямоугольника?
3. Какая процедура используется для рисования эллипса?
4. Какая команда используется для отображения текста?

Практическое занятие №35 Итоговая работа по первому семестру

Цель занятия:

1. Проверить знания и умения обучающихся на первый семестр.

Исходные данные: конспекты по пройденным темам

Содержание и порядок выполнения:

1. Выполнить задания по вариантам.

Значение логического выражения

Задание 1 Напишите наименьшее целое число *x*, для которого истинно высказывание:

НЕ (*X* < 6) **И** (*X* нечётное).

Решение. Запишем выражение в виде

(*X* >= 6) **И** (*X* нечётное).

Значит, наименьшее число, для которого высказывание будет истинным — 7.

Ответ: 7.

Задание 2 Напишите наименьшее целое число *x*, для которого истинно высказывание:

НЕ (*X* <= 7) **И** (*X* < 20).

Решение. Запишем выражение в виде

(*X* > 7) **И** (*X* < 20).

Значит, наименьшее число, для которого высказывание будет истинным — 8.

Ответ: 8.

Задание 3 Напишите наибольшее целое число *x*, для которого истинно высказывание:

НЕ (*X* <= 14) **И** (*X* <= 18).

Решение. Запишем выражение в виде

(*X* > 14) **И** (*X* <= 18).

Значит, наибольшее число, для которого высказывание будет истинным — 18.

Ответ: 18.

Задание 4 Напишите наибольшее целое число *x*, для которого истинно высказывание:

НЕ (*X* <= 6) **И НЕ** (*X* >= 11).

Решение. Запишем выражение в виде

(*X* > 6) **И** (*X* < 11).

Значит, наибольшее число, для которого высказывание будет истинным — 10.

Ответ: 10.

Задание 5 Напишите наибольшее целое число *x*, для которого истинно высказывание:

НЕ (*X* чётное) **И НЕ** (*X* >= 7).

Решение. Запишем выражение в виде

(*X* нечётное) **И** (*X* < 7).

Значит, наибольшее число, для которого высказывание будет истинным — 5.

Ответ: 5.

Задание 6 Напишите наибольшее целое число *x*, для которого истинно высказывание:

НЕ (*X* нечётное) **И НЕ** (*X* >= 10).

Решение. Запишем выражение в виде

(*X* чётное) **И** (*X* < 10).

Значит, наибольшее число, для которого высказывание будет истинным — 8.

Ответ: 8.

Задание 7 Напишите наибольшее целое число *x*, для которого истинно высказывание:

(*X* >= 6) **И НЕ** (*X* > 12).

Решение. Логическое «И» ложно тогда, когда ложно одно из высказываний. Запишем выражение в виде

(*X* >= 6) **И** (*X* <= 12).

Значит, наибольшее число, для которого высказывание будет истинным — 12.

Ответ: 12.

Задание 8 Напишите наименьшее целое число *x*, для которого истинно высказывание:

(*X* > 2) **И НЕ** (*X* > 13).

Решение. Запишем выражение в виде

(*X* > 2) **И** (*X* <= 13).

Значит, наименьшее число, для которого высказывание будет истинным — 3.

Ответ: 3.

Задание 9 Напишите наименьшее целое число *x*, для которого истинно высказывание:

НЕ (*X* < 5) **И НЕ** (*X* > 9).

Решение. Запишем выражение в виде

(*X* >= 5) **И** (*X* <= 9).

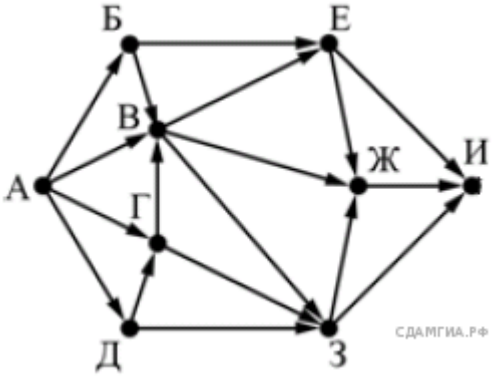
Значит, наименьшее число, для которого высказывание будет истинным — 5.

Ответ: 5.

Анализирование информации, представленной в виде схем (графы)

Задание 1 На рисунке — схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, З, И. По каждой дороге можно двигаться только в одном направлении, указанном

стрелкой. Сколько существует различных путей из города А в город И, проходящих через город В?



Решение. Количество путей до города Х = количество путей добраться в любой из тех городов, из которых есть дорога в Х.

При этом, если путь не должен проходить через какой-то город, нужно просто не учитывать этот город при подсчёте сумм. А если город, наоборот, обязательно должен лежать на пути, тогда для городов, в которые из нужного города идут дороги, в суммах нужно брать только этот город.

С помощью этого наблюдения посчитаем последовательно количество путей до каждого из городов:

А = 1

Б = А = 1

Д = А = 1

Г = А + Д = 1 + 1 = 2

В = А + Б + Г = 4

Е = В = 4 (Б не учитываем, т. к. там не проходим через В)

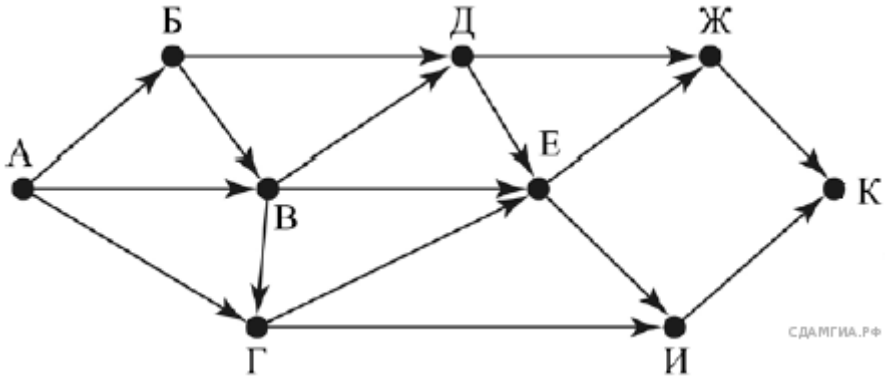
З = В = 4 (Д и Г не учитываем по тому же принципу)

Ж = В + Е + З = 4 + 4 + 4 = 12

И = Е + Ж + З = 4 + 12 + 4 = 20.

Ответ: 20.

Задание 2 На рисунке — схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, И, К. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из города А в город К, проходящих через город Г?



Решение. Количество путей до города Х = количество путей добраться в любой из тех городов, из которых есть дорога в Х.

При этом, если путь не должен проходить через какой-то город, нужно просто не учитывать этот город при подсчёте сумм. А если город, наоборот, обязательно должен лежать на пути, тогда для городов, в которые из нужного города идут дороги, в суммах нужно брать только этот город.

С помощью этого наблюдения посчитаем последовательно количество путей до каждого из городов:

А = 1.

Б = А = 1.

В = А + Б = 2.

Г = А + В = 3.

Е=Г = 3 (В и Д не учитываем, поскольку путь должен проходить через город Г).

Ж = Е = 3.

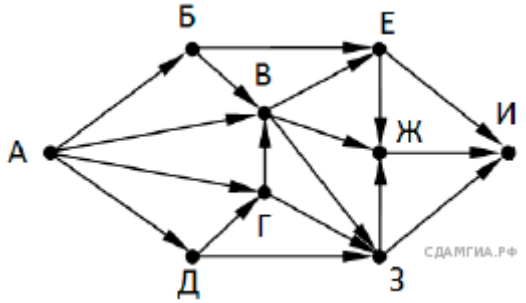
И = Е + Г = 6.

К = И + Ж = 9.

Примечание. Необходимо найти количество различных путей из города А в город К, проходящих через город **Г**.

Ответ: 9.

Задание 3 На рисунке — схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, З, И. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из города А в город И, проходящих через город Г?



Решение. Количество путей до города Х = количество путей добраться в любой из тех городов, из которых есть дорога в Х.

При этом, если путь не должен проходить через какой-то город, нужно просто не учитывать этот город при подсчёте сумм. А если город, наоборот, обязательно должен лежать на пути, тогда для городов, в которые из нужного города идут дороги, в суммах нужно брать только этот город.

С помощью этого наблюдения посчитаем последовательно количество путей до каждого из городов:

А = 1.

Д = А = 1.

Г = А + Д = 2.

В = Г = 2. (А и Б не учитываем, поскольку путь должен проходить через Г)

Е = В = 2. (Б не учитываем, поскольку путь должен проходить через Г)

З = В + Г = 4. (Д не учитываем, поскольку путь должен проходить через Г)

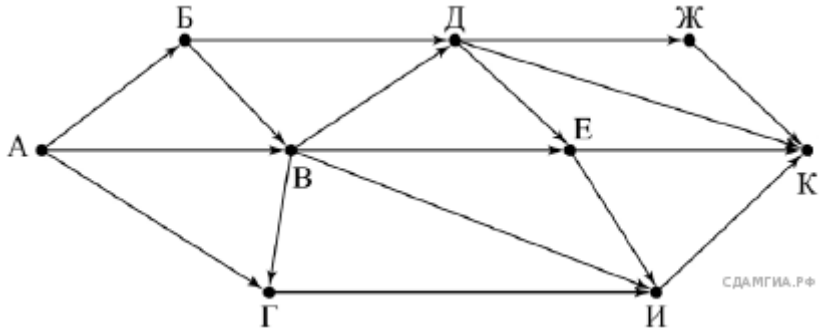
Ж = В + Е + З = 8.

И = Е + Ж + З = 14.

Примечание. Необходимо найти количество различных путей из города А в город И, проходящих через город **Г**.

Ответ: 14.

Задание 4 На рисунке — схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, И, К. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из пункта А в пункт К, не проходящих через пункт В?



Решение. Количество путей до города Х = количество путей добраться в любой из тех городов, из которых есть дорога в Х.

При этом, если путь не должен проходить через какой-то город, нужно просто не учитывать этот город при подсчёте сумм. А если город, наоборот, обязательно должен лежать на пути, тогда для городов, в которые из нужного города идут дороги, в суммах нужно брать только этот город.

С помощью этого наблюдения посчитаем последовательно количество путей до каждого из городов:

А = 1.

Б = А = 1.

Г = А = 1.

Д = Б = 1.

Е= Д = 1 (В не учитываем, поскольку путь не должен проходить через город В).

Ж = Д = 1.

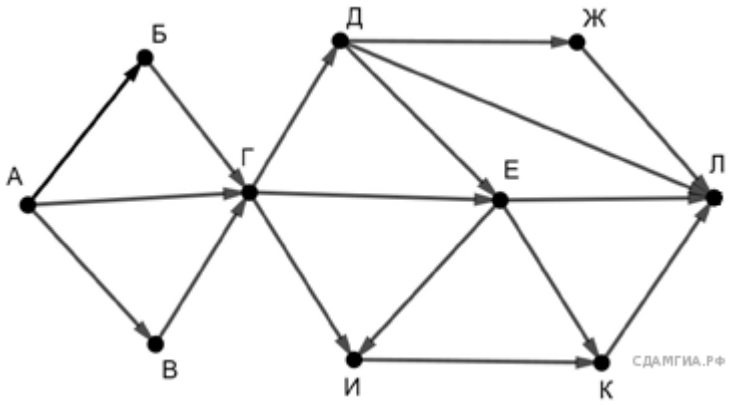
И = Г + Е = 2.

К = Ж + Е + И + Д = 5.

Примечание. Необходимо найти количество различных путей из города А в город К, не проходящих через город **В**.

Ответ: 5.

Задание 5 На рисунке — схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, И, К. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из пункта А в пункт Л, не проходящих через пункт Е?



Решение. Количество путей до города Х = количество путей добраться в любой из тех городов, из которых есть дорога в Х.

При этом, если путь не должен проходить через какой-то город, нужно просто не учитывать этот город при подсчёте сумм. А если город, наоборот, обязательно должен лежать на пути, тогда для городов, в которые из нужного города идут дороги, в суммах нужно брать только этот город.

С помощью этого наблюдения посчитаем последовательно количество путей до каждого из городов:

А = 1.

Б = А = 1.

В = А = 1.

Г = А + Б + В = 3.

Д = Г = 3.

И=Г = 3 (Е не учитываем, поскольку путь не должен проходить через город Е).

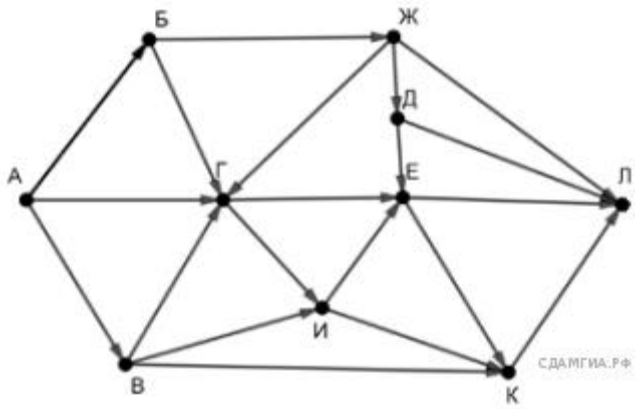
Ж = Д = 3.

К=И = 3 (Е не учитываем, поскольку путь не должен проходить через город Е).

Л = Д + Ж + К = 3 + 3 + 3 = 9 (Е не учитываем, поскольку путь не должен проходить через город Е).

Ответ: 9.

Задание 6 На рисунке — схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, И, К, Л. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из пункта А в пункт Л, проходящих через пункт Е?



Решение. Количество путей до города Х = количество путей добраться в любой из тех городов, из которых есть дорога в Х.

При этом, если путь не должен проходить через какой-то город, нужно просто не учитывать этот город при подсчёте сумм. А если город, наоборот, обязательно должен лежать на пути, тогда для городов, в которые из нужного города идут дороги, в суммах нужно брать только этот город.

С помощью этого наблюдения посчитаем последовательно количество путей до каждого из городов:

А = 1.

Б = А = В = 1.

Ж = Б = 1

Г = А + Б + В + Ж = 4

Д = Ж = 1

И = Г + В = 5.

Е = Д + Г + И = 10.

К = Е = 10. (И и В не учитываем, поскольку путь должен проходить через город Е).

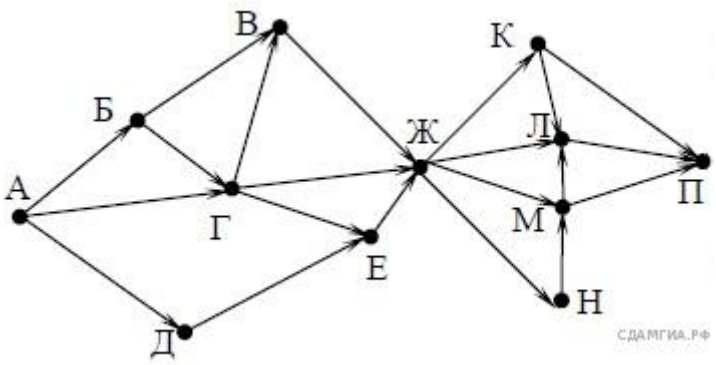
Л = К + Е = 20. (Д и Ж не учитываем, поскольку путь должен проходить через город Е).

Примечание. Необходимо найти количество различных путей из города А в город Л, проходящих через город **Е**.

Ответ: 20.

Задание 7 На рисунке — схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, К, Л, М, Н, П. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой.

Сколько существует различных путей из города А в город П, проходящих через город Н?



Решение. Количество путей до города П равно сумме путей в каждый из тех городов, из которых есть дорога в П. С помощью этого наблюдения посчитаем последовательно количество путей до каждого из городов, кроме тех, которые не проходят через город Н:

А = 1

Б = А = 1

Д = А = 1

Г = А + Б = 1 + 1 = 2

В = Б + Г = 1 + 2 = 3

Е = Г + Д = 2 + 1 = 3

Ж = В + Г + Е = 3 + 2 + 3 = 8

Н = Ж = 8

М = Н = 8 **(так как ищем пути проходящие через город Н)**

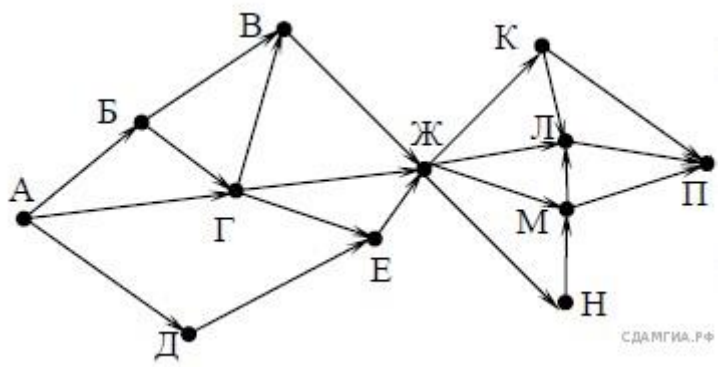
Л = М = 8

П = Л + М = 16.

Ответ: 16.

Задание 8 На рисунке — схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, К, Л, М, Н, П. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой.

Сколько существует различных путей из города А в город П, проходящих через город В?



Решение. Количество путей до города Х = количество путей добраться в любой из тех городов, из которых есть дорога в Х. При этом, если путь не должен проходить через какой-то город, нужно просто не учитывать этот город при подсчёте сумм. А если город, наоборот, обязательно должен лежать на пути, тогда для городов, в которые из нужного города идут дороги, в суммах нужно брать только этот город.

С помощью этого наблюдения посчитаем последовательно количество путей до каждого из городов:

А = 1.

Б = А = 1.

Г = А + Б = 2.

Д = А = 1.

В = Б + Г = 3.

Е = Г + Д = 3.

Ж=В=3. (Е и Г не учитываем, поскольку в этих вершинах не проходим через В).

К = Ж = 3.

Н = Ж = 3.

М = Ж + Н = 6.

Л = К + Ж + М = 12.

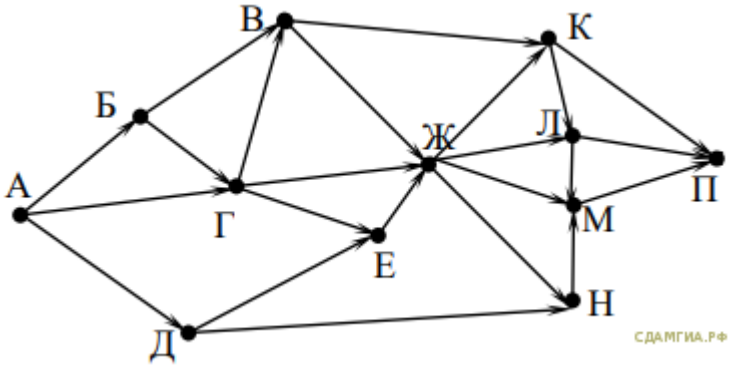
П = К + Л + М = 21.

Примечание. Необходимо найти количество различных путей из города А в город П, проходящих через город В.

Ответ: 21.

Задание 9 На рисунке — схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, К, Л, М, Н, П. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой.

Сколько существует различных путей из города А в город П, проходящих через город Л?



Решение. Для нахождения количества путей из города А в город П, проходящих через город Л, надо найти количество путей из города А в город Л, количество путей из города Л в город П, и перемножить получившиеся числа.

Найдем количество путей из А в Л:

А = 1.

Б = А = 1.

Г = А + Б = 2.

Д = А = 1.

В = Б + Г = 3.

Е = Г + Д = 3.

Ж = В + Г + Е = 8.

К = Ж + В = 11.

Л = Ж + К= 19.

Найдем количество путей из Л в П (при этом Л является исходным пунктом):

Л = 1.

М = Л = 1.

П = Л + М = 2.

Тогда количество путей из А в П, проходящих через Л, равно 19 · 2 = 38.

Ответ: 38

Сравнение чисел в различных системах счисления (системы счисления)

Задание 1Среди приведённых ниже трёх чисел, записанных в различных системах счисления, найдите максимальное и запишите его в ответе в десятичной

системе счисления. В ответе запишите только число, основание системы счисления указывать не нужно.

38_{16}, 75_8, 110100_2.

Решение. Переведём все числа в десятичную систему счисления:

1. $38_{16} = 56_{10}$;

2. $75_8 = 61_{10}$;

3. $110100_2 = 52_{10}$.

Таким образом, наибольшим среди этих трёх чисел является число 61.

Ответ: 61.

Задание 2 Среди приведённых ниже трёх чисел, записанных в различных системах счисления, найдите максимальное и запишите его в ответе в десятичной системе счисления. В ответе запишите только число, основание системы счисления указывать не нужно.

50_{16}, 106_8, 1001010_2.

Решение. Переведём все числа в десятичную систему счисления:

1. $50_{16} = 80_{10}$;

2. $106_8 = 70_{10}$;

3. $1001010_2 = 74_{10}$.

Таким образом, наибольшим среди этих трёх чисел является число 80.

Ответ: 80

Задание 3 Среди приведённых ниже трёх чисел, записанных в различных системах счисления, найдите минимальное и запишите его в ответе в десятичной системе счисления. В ответе запишите только число, основание системы счисления указывать не нужно.

32_{16}, 60_8, 110110_2.

Решение.

Переведём все числа в десятичную систему счисления:

1. $32_{16} = 50_{10}$;

2. $60_8 = 48_{10}$;

3. $110110_2 = 54_{10}$.

Таким образом, наименьшим среди этих трёх чисел является число 48.

Ответ: 48.

Задание 4 Среди приведённых ниже трёх чисел, записанных в различных системах счисления, найдите минимальное и запишите его в ответе в десятичной

системе счисления. В ответе запишите только число, основание системы счисления указывать не нужно.

14_{16}, 17_8, 10011_2.

Решение. Переведём все числа в десятичную систему счисления:

1. $14_{16} = 20_{10}$;

2. $17_8 = 15_{10}$;

3. $10011_2 = 19_{10}$.

Таким образом, наименьшим среди этих трёх чисел является число 15.

Ответ: 15.

Задание 5 Среди приведённых ниже трёх чисел, записанных в различных системах счисления, найдите минимальное и запишите его в ответе в десятичной системе счисления. В ответе запишите только число, основание системы счисления указывать не нужно.

60_{16}, 134_8, 1100001_2.

Решение. Переведём все числа в десятичную систему счисления:

1. $60_{16} = 96_{10}$;

2. $134_8 = 92_{10}$;

3. $1100001_2 = 97_{10}$.

Таким образом, наименьшим среди этих трёх чисел является число 92.

Ответ: 92.

Задание 6 Среди приведённых ниже трёх чисел, записанных в различных системах счисления, найдите максимальное и запишите его в ответе в десятичной системе счисления. В ответе запишите только число, основание системы счисления указывать не нужно.

59_{16}, 126_8, 1011100_2.

Решение.

Переведём все числа в десятичную систему счисления:

1. $59_{16} = 89_{10}$;

2. $126_8 = 86_{10}$;

3. $1011100_2 = 92_{10}$.

Таким образом, наибольшим среди этих трёх чисел является число 92.

Ответ: 92.

Задание 7 Среди приведённых ниже трёх чисел, записанных в различных системах счисления, найдите максимальное и запишите его в ответе в десятичной

системе счисления. В ответе запишите только число, основание системы счисления указывать не нужно.

26_{16}, 26_8, 11101_2.

Решение. Переведём все числа в десятичную систему счисления:

1. $26_{16} = 38_{10}$;

2. $26_8 = 22_{10}$;

3. $11101_2 = 29_{10}$.

Таким образом, наибольшим среди этих трёх чисел является число 38.

Ответ: 38.

Задание 8 Среди приведённых ниже трёх чисел, записанных в различных системах счисления, найдите максимальное и запишите его в ответе в десятичной системе счисления. В ответе запишите только число, основание системы счисления указывать не нужно.

28_{16}, 47_8, 101010_2.

Решение. Переведём все числа в десятичную систему счисления:

1. $28_{16} = 40_{10}$;

2. $47_8 = 39_{10}$;

3. $101010_2 = 42_{10}$.

Таким образом, наибольшим среди этих трёх чисел является число 42.

Ответ: 42.

Задание 9 Среди приведённых ниже трёх чисел, записанных в различных системах счисления, найдите максимальное и запишите его в ответе в десятичной системе счисления. В ответе запишите только число, основание системы счисления указывать не нужно.

49_{16}, 102_8, 1000111_2.

Решение. Переведём все числа в десятичную систему счисления:

1. $49_{16} = 73_{10}$;

2. $102_8 = 66_{10}$;

3. $1000111_2 = 71_{10}$.

Таким образом, наибольшим среди этих трёх чисел является число 73.

Ответ: 73.

Простейший циклический алгоритм, записанный на алгоритмическом языке

Задание 1 Запишите значение переменной *s*, полученное в результате работы следующей программы. Текст программы приведён на пяти языках программирования.

Python

```
s = 0
for kinrange(6,10):
  s = s + 12
print (s)
```

Алгоритмический язык

```
алг
нач
  цел s, k
  s := 0
  нцдля k от6до9
    s := s + 12
  кц
  вывод s
кон
```

Решение. Цикл «for k := 6 to 9 do» выполняется четыре раза. Каждый раз переменная s увеличивается на 12. Поскольку изначально s = 0, после выполнения программы получим: s = 4 · 12 = 48.

Задание 2 Запишите значение переменной *s*, полученное в результате работы следующей программы. Текст программы приведён на пяти языках программирования.

Python

```
s = 0
for kinrange(7,13):
  s = s + 11
print (s)
```

Алгоритмический язык

```
алг
нач
  цел s, k
  s := 0
  нцдля k от7до12
    s := s + 11
  кц
  вывод s
кон
```

Решение. Цикл «for k := 7 to 12 do» выполняется шесть раз. Каждый раз переменная s увеличивается на 11. Поскольку изначально s = 0, после выполнения программы получим: s = 6 · 11 = 66.

Задание 3. Запишите значение переменной *s*, полученное в результате работы следующей программы. Текст программы приведён на пяти языках программирования.

Python

```
s = 0
for kinrange(6,13):
  s = s + 10
print (s)
```

Алгоритмический язык

```
алг
нач
  цел s, k
  s := 0
  нцдля k от6до12
    s := s + 10
  кц
  вывод s
кон
```

Решение. Цикл «fork := 6 to 12 do» выполняется семь раз. Каждый раз переменная s увеличивается на 10. Поскольку изначально s = 0, после выполнения программы получим: s = 7 · 10 = 70.

Задание 4. Запишите значение переменной s, полученное в результате работы следующей программы. Текст программы приведён на пяти языках программирования.

Python

```
s = 0
for kinrange(12,16):
  s = s + 13
print (s)
```

Алгоритмический язык

```
алг
нач
  цел s, k
  s := 0
  нцдля k от12до15
    s := s + 13
  кц
  вывод s
кон
```

Решение. Цикл «for k := 12 to 15 do» выполняется четыре раза. Каждый раз переменная s увеличивается на 13. Поскольку изначально s = 0, после выполнения программы получим: s = 4 · 13 = 52.

Задание 5. Запишите значение переменной *s*, полученное в результате работы следующей программы. Текст программы приведён на пяти языках программирования.

Python

```
s = 0
for kinrange(13,17):
   s = s + 14
print (s)
```

Алгоритмический язык

```
алг
нач
  цел s, k
  s := 0
  нцдля k от13до16
    s := s + 14
  кц
  вывод s
кон
```

Решение. Цикл «for k := 13 to 16 do» выполняется четыре раза. Каждый раз переменная s увеличивается на 14. Поскольку изначально s = 0, после выполнения программы получим: s = 4 · 14 = 56.

Задание 6. Запишите значение переменной *s*, полученное в результате работы следующей программы. Текст программы приведён на пяти языках программирования.

Python

```
s = 8
for kinrange(3,9):
   s = s + 8
print (s)
```

Алгоритмический язык

```
алг
нач
  цел s, k
  s := 8
  нцдля k от3до8
    s := s + 8
  кц
  вывод s
кон
```

Решение. Цикл «for k := 3 to 8 do» выполняется шесть раз. Каждый раз переменная s увеличивается на 8. Поскольку изначально s = 8, после выполнения программы получим: s = 8 + 6 · 8 = 56.

Задание 7 Запишите значение переменной *s*, полученное в результате работы следующей программы. Текст программы приведён на пяти языках программирования.

Python

```
s = 50
for kinrange(9):
```

```
  s = s − 4
print (s)
```

Алгоритмический язык

```
алг
нач
  цел s, k
  s := 50
  нцдля k от0до8
    s := s − 4
  кц
  вывод s
кон
```

Решение. Цикл «for k := 0 to 8 do» выполняется девять раз. Каждый раз переменная s уменьшается на 4. Поскольку изначально s = 50, после выполнения программы получим: s = 50 − 9 · 4 = 14.

Задание 8. Определите, что будет напечатано в результате работы следующей программы. Текст программы приведён на пяти языках программирования.

Python

```
s = 50
for kinrange(1,10):
  s = s − 3
print (s)
```

Алгоритмический язык

```
алг
нач
  цел s, k
  s := 50
  нцдля k от1до9
    s := s − 3
  кц
  вывод s
кон
```

Решение. Цикл «for k := 1 to 9 do» выполняется девять раз. Каждый раз переменная s уменьшается на 3. Поскольку изначально s = 50, после выполнения программы получим: s = 50 − 9 · 3 = 23.

Задание 9. Определите, что будет напечатано в результате работы следующей программы. Текст программы приведён на пяти языках программирования.

Python

```
s = 100
for kinrange(1,10):
  s = s − 5
print (s)
```

Алгоритмический язык

```
алг
нач
  цел s, k
  s := 100
  нцдля k от1до9
    s := s − 5
  кц
  вывод s
кон
```

Решение. Цикл «for k := 1 to 9 do» выполняется девять раз. Каждый раз переменная s уменьшается на 5. Поскольку изначально s = 100, после выполнения программы получим: s = 100 − 9 · 5 = 55.

Скорость передачи информации

Задание 1 Файл размером 64 Кбайт передаётся через некоторое соединение со скоростью 1024 бит в секунду. Определите размер файла (в Кбайт), который можно передать за то же время через другое соединение со скоростью 256 бит в секунду. В ответе укажите одно число — размер файла в Кбайт. Единицы измерения писать не нужно.

Решение. Определим время передачи:

64 Кбайт/1024 бит в секунду = (64 · 1024 · 8 бит)/(1024 бит в секунду) = 64 · 8 секунд.

Вычислим размер файла:

64 · 8 секунд · 256 бит в секунду = 64 · 256 байт = 16 Кбайт.

Ответ: 16.

Задание 2 Файл размером 100 Кбайт передаётся через некоторое соединение со скоростью 1536 бит в секунду. Определите размер файла (в Кбайт), который можно передать за то же время через другое соединение со скоростью 768 бит в секунду. В ответе укажите одно число — размер файла в Кбайт. Единицы измерения писать не нужно.

Решение. Определим время передачи:

100 Кбайт/1536 бит в секунду = 100 · 1024 · 8 бит/3 · 512 бит в секунду = 100 · 16/3 секунд.

Вычислим размер файла:

100 · 16/3 секунд · 768 бит в секунду = 100 · 512 · 8 бит = 50 Кбайт.

Задание 3 Файл размером 160 Кбайт передаётся через некоторое соединение со скоростью 2048 бит в секунду. Определите размер файла (в Кбайт), который можно передать за то же время через другое соединение со скоростью 768 бит в

секунду. В ответе укажите одно число — размер файла в Кбайт. Единицы измерения писать не нужно.

Решение. Определим время передачи:

160 Кбайт/2048 бит в секунду = (160 · 1024 · 8 бит)/(2 · 1024 бит в секунду) = 160 · 4 секунд.

Вычислим размер файла:

160 · 4 секунд · 768 бит в секунду = 60 Кбайт.

Задание 4 Файл размером 60 Кбайт передаётся через некоторое соединение со скоростью 3072 бит в секунду. Определите размер файла (в Кбайт), который можно передать за то же время через другое соединение со скоростью 256 бит в секунду. В ответе укажите одно число — размер файла в Кбайт. Единицы измерения писать не нужно.

Решение. Определим время передачи:

60 Кбайт/3072 бит в секунду = (60 · 1024 · 8 бит)/(3 · 1024 бит в секунду) = 20 · 8 секунд.

Вычислим размер файла:

20 · 8 секунд · 256 бит в секунду = 5120 · 8 бит = 5 Кбайт.

Задание 5 Файл размером 80 Кбайт передаётся через некоторое соединение со скоростью 1536 бит в секунду. Определите размер файла (в Кбайт), который можно передать за то же время через другое соединение со скоростью 768 бит в секунду. В ответе укажите одно число — размер файла в Кбайт. Единицы измерения писать не нужно.

Решение. Размер переданного файла = время передачи · скорость передачи. Заметим, что скорость передачи во втором случае в 1536/768 = 2 раза меньше скорости в первом случае. Поскольку время передачи файлов одно и то же, размер файла, который можно передать во втором случае, тоже в 2 раза меньше. Он будет равен 80/2 = 40 Кбайт.

Задание 6 Файл размером 80 Кбайт передаётся через некоторое соединение со скоростью 2048 бит в секунду. Определите размер файла (в Кбайт), который можно передать за то же время через другое соединение со скоростью 768 бит в секунду. В ответе укажите одно число — размер файла в Кбайт. Единицы измерения писать не нужно.

Решение. Определим время передачи:

80 Кбайт/2048 бит в секунду = (80 · 1024 · 8 бит)/(2048 бит в секунду) = 40 · 8 секунд.

Вычислим размер файла:

40 · 8 секунд · 768 бит в секунду = 30720 · 8 бит = 30 Кбайт.

Задание 7 Файл размером 120 Кбайт передаётся через некоторое соединение со скоростью 3072 бит в секунду. Определите размер файла (в Кбайт), который можно передать за то же время через другое соединение со скоростью 1024 бит в секунду. В ответе укажите одно число — размер файла в Кбайт. Единицы измерения писать не нужно.

Решение. Размер переданного файла = время передачи · скорость передачи. Заметим, что скорость передачи во втором случае в 3072/1024 = 3 раза меньше скорости в первом случае. Поскольку время передачи файлов одно и то же, размер файла, который можно передать во втором случае, тоже в 3 раза меньше. Он будет равен 120/3 = 40 Кбайт.

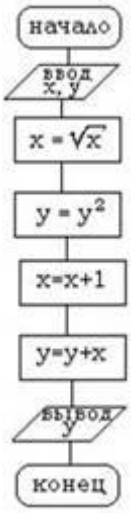
Задание 8 Файл размером 4000 Кбайт передаётся через некоторое соединение в течение 1 минуты. Определите размер файла (в Кбайт), который можно передать через это соединение за 45 секунд. В ответе укажите одно число — размер файла в Кбайт. Единицы измерения писать не нужно.

Решение Вычислим скорость передачи данных по каналу: 4000 Кбайт/60 сек = 200/3 Кбайт/сек. Следовательно, размер файла, который можно передать за 45 секунд равен 200/3 Кбайт/сек · 45 сек = 3000 Кбайт.

Ответ: 3000.

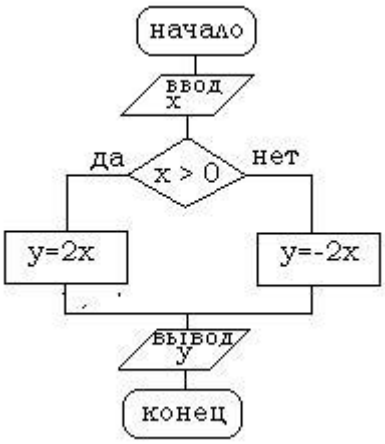
Блок-схемы

1. Дана блок-схема алгоритма. Определить результат выполнения алгоритма при определённых значениях исходных данных

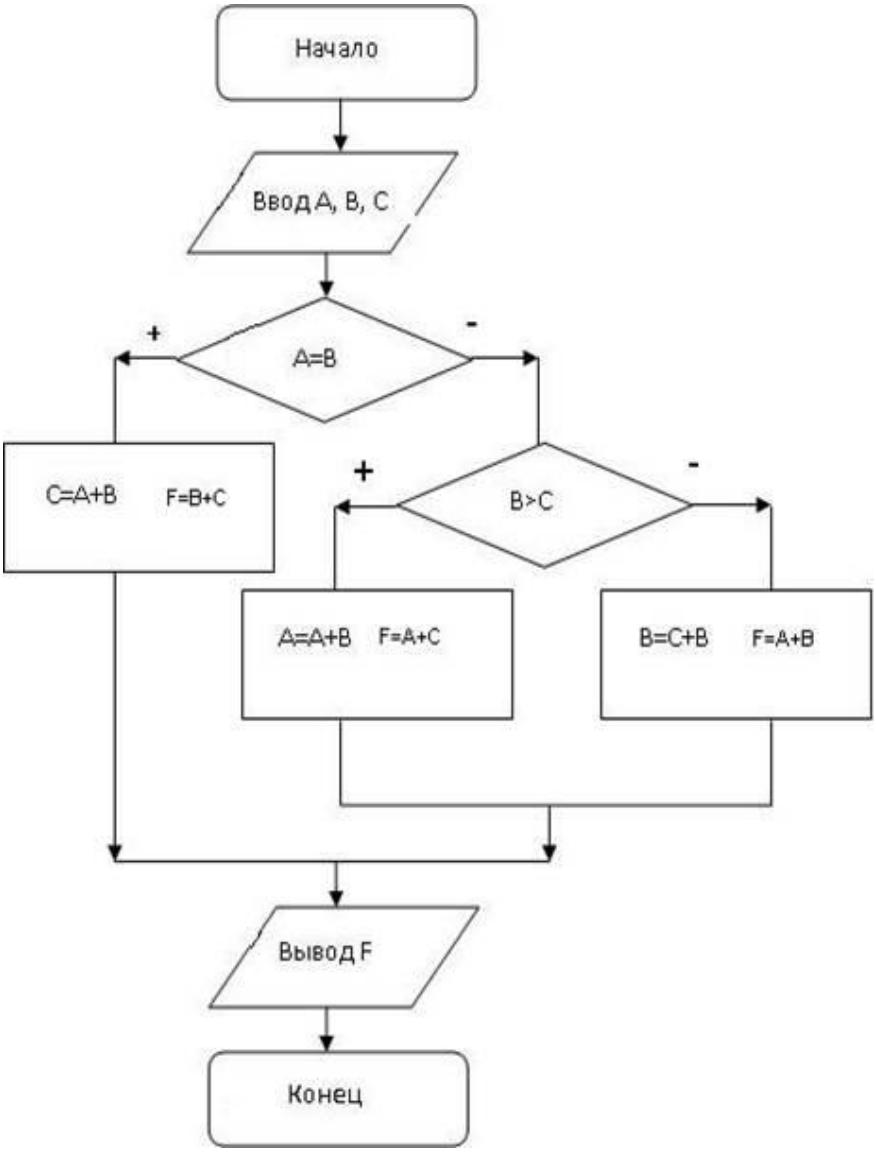


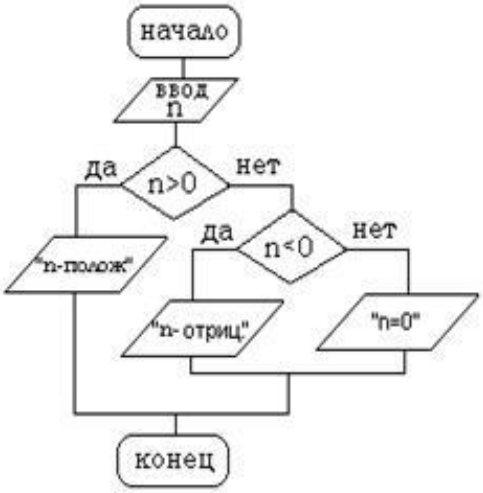
2 Дана блок-схема алгоритма

Определить результат выполнения алгоритма при определённых значениях исходных данных



3. Дана блок-схема алгоритма





Определить результат выполнения алгоритма при определённых значениях исходных данных

Например, при n=15 или n=0 или n=-7

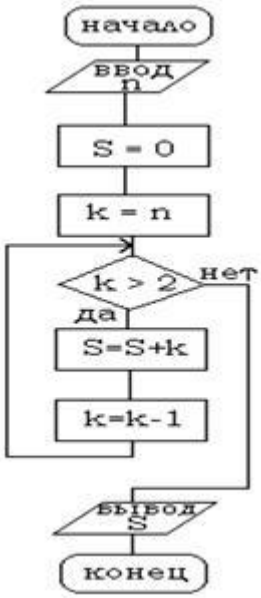
4. Дана блок-схема алгоритма

Определить результат выполнения алгоритма при определённых значениях исходных данных

Например, при A=7; B=8; C=9 или A=6; B=6; C=-10 или A=6; B=10; C=-10

5.Дана блок-схема алгоритма

Определить результат выполнения алгоритма при определённых значениях исходных данных



Например, при n=4 или n=1