Федеральное агентство по рыболовству
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Калининградский государственный технический университет»
(ФГБОУ ВО «КГТУ»)

УТВЕРЖДАЮ
Начальник УРОПСП

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
ФГБОУ ВО «КГТУ»
СВЕДЕНИЯ О СЕРТИФИКАТЕ ЭП
Сертификат: 01f058c007bae2c8149e7712c28ce8469
Владелец: Мельникова Виктория Александровна
Действителен: с 18.04.2022 по 18.04.2023

Фонд оценочных средств
(приложение к рабочей программе модуля)

«ПРОГРАММИРОВАНИЕ»

основной профессиональной образовательной программы бакалавриата
по направлению подготовки

09.03.03 ПРИКЛАДНАЯ ИНФОРМАТИКА
Профиль программы
«ПРИКЛАДНАЯ ИНФОРМАТИКА В ЭКОНОМИКЕ»

ИНСТИТУТ Цифровых технологий

РАЗРАБОТЧИК Кафедры систем управления и вычислительной техники

1 РЕЗУЛЬТАТЫ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Таблица 1 – Планируемые результаты обучения по дисциплине, соотнесенные с установленными индикаторами достижения компетенций

Код и наименование компетенции	Индикаторы достижения компетенции	Дисциплина	Результаты обучения (владения, умения и знания), соотнесенные с компетенциями/индикаторами достижения компетенции
ОПК-7: Способен разрабатывать алгоритмы и программы, пригодные для практического применения.	ОПК-7.2: Применяет языки программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ.	Алгоритмизация и программирование (раздел «Программирование»)	*Знать:* фундаментальные (базовые) понятия программирования компьютерной обработки данных; - структурную технологию разработки алгоритмов и программ для ЭВМ (проектирования, написания, тестирования и отладки многомодульных программ на процедурно-ориентированном языке); - основы документирования результатов программирования. *Уметь:* осуществлять постановку задач, проектировать программы их решения на ЭВМ, использовать систему прикладного программирования (применяемую в курсе), тестировать и осуществлять отладку программ, документировать результаты программирования. *Владеть:* навыками разработки программ на языке высокого уровня.

2 ПЕРЕЧЕНЬ ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПОЭТАПНОГО ФОРМИРОВАНИЯ РЕЗУЛЬТАТОВ ОСВОЕНИЯ ДИСЦИПЛИНЫ (ТЕКУЩИЙ КОНТРОЛЬ) И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

2.1 Для оценки результатов освоения дисциплины используются:

- оценочные средства текущего контроля успеваемости;

- оценочные средства для промежуточной аттестации по дисциплине.

2.2 К оценочным средствам текущего контроля успеваемости относятся:

- тестовые задания;

- задания и контрольные вопросы по лабораторным работам;

2.3 К оценочным средствам для промежуточной аттестации по дисциплине, проводимой в форме зачета и экзамена, относятся:

- задание по контрольной работе для студентов заочной формы обучения;

-задания по курсовой работе;

- экзаменационные вопросы.

- промежуточная аттестация в форме зачета проходит по результатам прохождения всех видов текущего контроля успеваемости.

3 ОЦЕНОЧНЫЕ СРЕДСТВА ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ

3.1 Тестовые задания представлены в Приложение № 1. Тестирование студенты проходят в конце второго семестра, после освоения все тем по дисциплине.

Для прохождения теста в электронной информационной образовательной среде отводиться одна попытка, время прохождения теста 40 минут. Тест считается успешно пройденным при 10 правильных ответах на вопросы теста (65%). Вопросы теста хранятся в банке вопросов, тест формируется путем случайного отбора вопросов, которые предварительно перемешиваются.

3.2 Задания на лабораторные работы и контрольные вопросы к ним:

Лабораторная работа №1. Линейные алгоритмы и программы.

Задание к лабораторной работе: согласно своему варианту составить для задачи линейный алгоритм, схему алгоритма, написать линейную программу на языке Python. Оформить ввод значений и вывод результатов. Выполнить задачу для разных наборов данных.

Контрольные вопросы для самопроверки:

1. Зачем нужен тип переменной?
2. Почему желательно выводить на экран подсказку перед вводом данных?
3. Какие простые типы данных вы знаете?
4. Что такое приоритет операций? Зачем он нужен?
5. В каком порядке выполняются операции, если они имеют одинаковый приоритет?
6. Зачем используются скобки?
7. Чем отличаются операции /, // и % ?

Лабораторная работа №2. Использования структур выбора.

Задание к лабораторной работе: модифицировать схему алгоритма и программу предыдущей работы №1 в соответствии с заданием по своему варианту. Организовать разветвление, представить результаты или сообщение о невыполнении условия.

Контрольные вопросы для самопроверки:

1. В чем сущность "структуры выбора"?
2. Как организован выбор условий в программе?
3. Какие средства языка используются при описании структур выбора?

Лабораторная работа №3. Использования структур повторения. Циклы.

Задание к лабораторной работе: организовать многократное (циклическое) выполнение предыдущей программы лабораторной работы №2 при изменении параметра (или одного из исходных параметров) в заданном диапазоне. Представить схему алгоритма.

Контрольные вопросы для самопроверки:

1. Что собой представляет алгоритмическая структура повторения? Какие основные операции она предусматривает?
2. Как в программе выделить область действия цикла?
3. Каким образом структура повторения "повторять пока" записывается в программах?
4. Что необходимо предусматривать для правильного исполнения структуры повторения?
5. В чем отличие операторов for и while?

Лабораторная работа №4. Списки. Операции, методы и функции списков.

Задание к лабораторной работе: согласно своему варианту написать программу на языке Python.

Тестирование программы осуществить несколько раз, используя различные варианты исходных данных для получения всех предусмотренных в программе возможных вариантов результатов решения задачи. Предусмотреть вывод сообщения при отрицательном результате решения. Исходные списки следует генерировать (см. таблицу Модуль random).

Контрольные вопросы для самопроверки:

1. Как можно определить список?
2. Какой метод определяет длину списка?
3. Элементами списка могут переменные разных типов?
4. Как определить наличие элементов в списке?
5. Какой метод предназначен для добавления элемента в список?

Лабораторная работа №5. Работа с матрицами.

Задание к лабораторной работе: согласно своему варианту написать программу на языке Python с выводом исходной матрицы. Предусмотреть вывод сообщения при отрицательном результате решения. Исходные матрицы генерировать, либо задавать в программе.

Контрольные вопросы для самопроверки:

1. Что такое матрица?
2. В чем сходство и отличие данных типа матрица и список?
3. Как организованы матрицы в Питоне?
4. Какие способы заполнения матрицы Вы знаете?
5. Какие типовые алгоритмы обработки матриц существуют?

Лабораторная работа №6. Строки. Операции, методы и функции строк.

Задание к лабораторной работе: согласно своему варианту написать программу на языке Python. Тестирование программы осуществить несколько раз, используя различные варианты исходных данных для получения всех предусмотренных в программе возможных вариантов результатов решения задачи (включая отрицательный). В предложении слова разделять одним пробелом, знаки препинания писать слитно с предшествующим словом.

Контрольные вопросы для самопроверки:

1. Как можно определить строку?
2. Какой метод возвращает длину строки?
3. Как разбить строку по пробелам?
4. С помощью какого метода можно сформировать из списка строку?
5. Как организовать доступ к элементам строки?

Лабораторная работа №7. Функции. Аргументы функции. Вызов функции. Локальные и глобальные переменные.

Задание к лабораторной работе: согласно своему варианту написать программу реализации функции на языке Python. Представьте результаты для диапазона (или списка) значений одного из параметров задания. Выводить исходные данные и результат.

Контрольные вопросы для самопроверки:

1. Как по тексту программы определить, какое значение возвращает функция?
2. Какие функции называются логическими? Зачем они нужны?
3. В чем отличие локальных параметров от глобальных?
4. Расположение функции в программе?

Лабораторная работа №8. Словари. Обработка исключений.

Задание к лабораторной работе: согласно своему варианту написать программу создания и обработки словаря на языке Python. Создать два списка, каждый из 8-9 элементов. Первый

список – ключи, уникальные элементы. Второй список – объекты. В качестве значений используйте элементы или списки.

Выполнить задания с выводом результат каждого пункта задачи:

- Преобразовать списки в словарь вида: {ключ1: объект1, ключ2: объект2…}
- Получить значение элемента по ключу
- Добавить элемент в словарь
- Удалить элемент словаря
- Отсортировать словарь по ключам
- Создать копию словаря и отсортировать значения
- Создать список из ключей

В программе использовать хотя бы одну обработку исключений – при работе со словарем, или при вводе данных для запроса, или при отрицательном результате решения.

Контрольные вопросы для самопроверки:

1. Что такое словарь?
2. В чем отличие словаря от списка?
3. Какие средства предназначены для создания словаря?
4. Какие элементы словаря должны быть уникальными?
5. Перечислите методы для работы со словарями
6. Основное назначение использования обработчика исключений.

Лабораторная работа №9. Основные методы для работы с файлами.

Задание к лабораторной работе: создать в Блокноте текстовый файл из 5 – 7 записей с данными по варианту.

Написать программу, которая:

- выводит на экран содержимое файла
- выполняет вычисления в соответствии с заданием по варианту
- создает новый файл в соответствии с заданием по варианту (предусмотреть вариант отсутствия данных по заданию)
- выводит на экран содержимое нового файла

В программе использовать хотя бы одну обработку исключений при работе с файлом.

Контрольные вопросы для самопроверки:

1. Что такое файл? Какие типы файлов используют в программировании?
2. Допустимы ли различные типы данных для элементов одной записи?
3. Какие основные действия с файлами? Для каких целей можно открыть файл?

4. Указать операторы: запись данных в файл и чтение данных из файла?
5. Что происходит с файлами, когда программа завершает свою работу?

Лабораторная работа №10. Графика в Python.

Задание к лабораторной работе: согласно своему варианту написать программу на языке Python, формирующую указанное изображение с использованием графических примитивов. Выбор программного обеспечения предоставляется студенту (модуль Graph или компонента Canvas). В рисунке использовать минимум шесть различных простейших фигур. Цветовую гамму и размер выбрать самостоятельно. Допускается собственный рисунок.

Написать программу для построения графика функции

Контрольные вопросы для самопроверки:

1. Какие средства языка предназначены для изображения простейших фигур?
2. Как задать цвет и толщину линий?
3. Для каких фигур можно выполнить заливку и как задать цвет заливки?
4. Как представлены координаты точек на холсте?

Лабораторная работа №11. Анимация в Python.

Задание к лабораторной работе: написать на языке Python программу анимации для графического объекта предыдущей работы, Перемещение управлять клавишами стрелками. Траекторию и завершение движения определите по варианту.

Контрольные вопросы для самопроверки:

1. Что понимают под объектом при создании анимации?
2. Какие средства языка предназначены для перемещения объектов?
3. Что такое "обработчики событий области рисования"?
4. Как задать шаг изменения координат при перемещении объекта?
5. Как завершить работу программы?

Лабораторная работа №12. Введение в объектно-ориентированное программирование. Создание классов. Атрибуты и методы.

Задание к лабораторной работе: написать на языке Python программу согласно своему варианту с созданием и использованием класса. Для вывода и решения использовать методы. Исходные данные определить вне класса. При решении предусмотреть отсутствие данных, удовлетворяющих условию.

Контрольные вопросы для самопроверки:

1. Что такое атрибуты и методы класса?
2. В чем отличие класса от объекта?
3. Какая разница между методом и функцией?

4. Какой параметр указывается при использовании в методе атрибута класса?

Лабораторная работа №13. Принципы объектно–ориентированного программирования. Наследование. Инкапсуляция.

Задание к лабораторной работе: напишите на языке Python программу по варианту своего задания из таблицы. Самостоятельно задайте параметры метода для определения класса – наследника. В основном классе используйте конструктор. При решении предусмотреть отсутствие данных, удовлетворяющих условию.

Контрольные вопросы для самопроверки:

1. Что такое атрибуты и методы класса?
2. В чем отличие класса от объекта?
3. Какая разница между методом и функцией?
4. Какой параметр указывается при использовании в методе атрибута класса?

Лабораторная работа №14. Создание графического интерфейса пользователя (GUI) средствами библиотеки tkinter.

Задание к лабораторной работе: ваша программа на языке Python должна:

- Создать главное окно.
- Создать виджеты по варианту и выполнить конфигурацию их свойств (опций). Дизайн формы придумайте самостоятельно
- Определить события, то есть то, на что будет реагировать программа.
- Определить обработчики событий, то есть то, как будет реагировать программа.
- Расположить виджеты в главном окне.
- Запустить цикл обработки событий.

(Последовательность не обязательно такая, но первый и последний пункты всегда остаются на своих местах).

Контрольные вопросы для самопроверки:

1. Что понимают под виджетами?
2. Что образует главный цикл обработки событий в tkinter?
3. Как организовано размещение виджетов?
4. Какие свойства общие для всех виджетов?
5. Назначение виджетов Button, Label, Entry, Text?
6. Назначение виджетов Listbox, Frame, Radiobutton, Scrollbar?

3.3 Оценка результатов выполнения задания по каждой лабораторной работе производится при представлении студентом отчета по лабораторной работе и на основании

ответов студента на вопросы по тематике лабораторной работы. Студент, самостоятельно выполнивший задание и продемонстрировавший знание использованных им средств и приемов программирования задачи получает по лабораторной работе оценку «зачтено».

4 ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ

4.1 Промежуточная аттестация по дисциплине проводится в форме зачета в первом семестре и курсовой работы и экзамена во втором семестре. Контрольную работу в первом семестре выполняют студенты заочной формы обучения.

Промежуточная аттестация в форме зачета проходит по результатам прохождения всех видов текущего контроля успеваемости в семестре.

К экзамену допускаются студенты, положительно аттестованные по результатам текущего контроля.

4.2 Задания по контрольной работе. Критерии и шкала оценивания по контрольной работе.

Контрольная работа выполняется студентами заочной формы обучения.

Задание для контрольной работы состоит из двух контрольных вопросов. Первая тема выбирается по предпоследней цифре студенческого билета. Вторая тема выбирается по последней цифре студенческого билета. Объем теоретических вопросов – 3-5 страниц. Ответ на вопрос должен содержать как теоретическое описание материала, так и примеры, иллюстрирующие описанный теоретический вопрос. При выполнении контрольной работы не следует бездумно копировать готовые рефераты и курсовые. Информация должна быть проанализирована и переработана, а не скопирована из источника.

Первый вопрос контрольной работы по дисциплине

1. Введение в Python. История и назначение.
2. Типы данных в Python. Преобразование типов
3. Арифметические операции в Python. Операции над целыми и вещественными числами
4. Ввод данных и преобразования при вводе для Python.
5. Вывод данных. Форматированный вывод.
6. Логический тип данных в Python. Операции отношения. Логические операции
7. Условный оператор в Python
8. Циклы с заданным числом повторений в Python.
9. Циклы по условию в Python.
10. Структуры данных. Строки в Python. Операции со строками

Второй вопрос контрольной работы по дисциплине

1. Структуры данных. Списки в Python. Операции со списками
2. Структуры данных. Списки в Python. Генераторы списков.
3. Структуры данных. Вложенные списки в Python (матрицы). Создание и обработка.
4. Структуры данных. Кортежи в Python. Операции с кортежами и методы кортежей.
5. Структуры данных. Словари в Python. Операции со словарями и методы словарей.
6. Структуры данных. Работа с файлами.
7. Функции. Определение функции. Передача параметров в функцию. Области видимости переменных в Python.
8. Понятие объектно-ориентированного программирования.
9. Основные принципы объектно-ориентированного программирования.
10. Описание классов в Python. Поля и методы класса. Создание класса.

Контрольная работа оценивается как «зачтено» или «не зачтено». Оценка «зачтено» выставляется при полном раскрытии теоретических вопросов, использовании актуальной учебной литературы по заданной тематике. При этом приведенные студентам примеры оригинальны и полностью соответствуют вопросам контрольной работы. При не полном выполнении перечисленных условий контрольная работа отправляется на переработку.

4.3 Задания на курсовую работу. Критерии и шкала оценивания по курсовой работе.

Задание на курсовую работу для студентов очной формы обучения.

Курсовая работа по дисциплине состоит из двух частей:

Первая часть.

Ответ на теоретический вопрос из списка вопросов. Номер вопроса совпадает с вариантом и определяется преподавателем. Объем теоретического вопроса – 3-5 листов. Ответ на вопрос должен содержать как теоретическое описание материала, так и 5-7 примеров задач, иллюстрирующих описанный теоретический вопрос, придуманных и написанных самостоятельно студентом. Каждая задача должна иметь формулировку и решение. В конце ответа на теоретический вопрос приводится источник или источники (списки литературы или/и адреса сайтов), оформленные по соответствующему ГОСТУ.

Список вопросов для первой части курсовой работы:

1. Арифметические операции в Python. Операции над целыми и вещественными числами.
2. Оператор присваивания в Python. Ввод данных.
3. Вывод данных. Форматы вывода.
4. Функции в Python. Глобальные и локальные переменные. Создание функции.

5. Логический тип данных в Python. Операции отношения. Логические операции.
6. Условный оператор в Python.
7. Циклы с заданным числом повторений и циклы по условию.
8. Списки в Python. Операции со списками и основные методы списков.
9. Списки списков в Python (матрицы). Создание и обработка.
10. Строки в Python. Операции со строками и основные методы строк.
11. Кортежи в Python. Операции с кортежами и методы кортежей.
12. Словари в Python. Операции со словарями и методы словарей.
13. Работа с файлами.
14. Модуль graph. Создание графических примитивов.
15. Модуль graph. Создание анимированных изображений.
16. Понятие ООП. Основные принципы ООП.
17. Описание классов. Атрибуты и методы класса. Экземпляры класса.
18. Реализация принципов ООП в Python.
19. Событийно–ориентированное программирование. Программы с графическим интерфейсом.
20. Модуль tkinter. Создание компонентов (виджетов): кнопка, метка (надпись), текст однострочный и многострочный. Свойства и методы этих виджетов.
21. Модуль tkinter. Создание компонентов (виджетов): рамка, флажки и радиокнопки, списки, шкала, окно верхнего уровня. Свойства и методы этих виджетов.
22. Модуль tkinter. Менеджеры геометрии.
23. Модуль tkinter. Метод bind. Типы переменных и события.
24. Исключения. Обработка исключений.

Вторая часть.

Написание оригинальной программы согласно варианту, выданному студенту преподавателем. Программа должна содержать графический интерфейс (GUI). При разработке формы GUI рекомендуется использовать следующие виджеты:

- кнопки начала и завершения работы;
- метку с указанием имени и фамилии студента, а также его группы;
- кнопки для поиска информации и расчетов;
- однострочное тестовое поле для ввода имени файла, который будет обрабатываться в программе,

- однострочное тестовое поле для ввода условия отбора записей из файла,
- метки для вывода результатов расчета,
- список или многострочное тестовое поле для вывода данных из файла, удовлетворяющих условию.

Дизайн формы студент определяет самостоятельно. Информация хранится в файле (10-15 записей), структура записи файла определена вариантом задания. Имя файла вводится в окно ввода, по кнопке «Открыть» файл открывается. В поле вводится условие поиска информации, результат – список или многострочное тестовое поле с записями по заданному условию (или сообщение об отсутствии данных). Кнопка «Расчет» инициирует расчеты и вывод результата. Кнопка «Закрыть» закрывает файл и форму.

Пример задания на курсовое проектирование:

Наименование задачи: Получение сведений о вузах.

Структура записей исходного файла: наименование вуза, наименование направления обучения, форма обучения, продолжительность обучения, стоимость курса.

Запрос: стоимость.

Результаты: Список направлений и вузов, стоимость обучения по которым не превышает указанной суммы, среднюю стоимость.

Задание на курсовую работу для студентов заочной формы обучения.

Курсовая работа предполагает разработку программы учетной задачи, при решении которой по указываемому пользователем запросу формируются (в результате обработки предварительно созданного файла) и представляются в виде таблицы определенные сведения. Тестовый файл, содержащий исходную информацию для решения задачи информации, должен содержать 10-15 записей, но значения элементов должны обеспечивать проверку правильности программы для различных вариантов решения задачи (получения упорядоченных сведений, получения таблицы с одной строкой и т. п.).

Варианты запросов при тестировании программы также должны быть ориентированы на различные варианты решения задачи, в том числе быть и ошибочными, и такими, для которых сведения в тестовом файле отсутствуют.

Пример задания на курсовое проектирование:

Наименование задачи: Получение сведения об абонентах

Структура записей исходного файла: наименование тарифного плана, количество абонентов на начало года, количество подключившихся абонентов, количество отключившихся абонентов.

Запрос: число абонентов.

Получить сведения: о тарифных планах, где количество подключившихся абонентов больше заданного.

Результаты курсовой работы представляются в виде пояснительной записки, в которой будут представлены результаты проектирования (в электронном и печатном вариантах), и программы для задачи по варианту, исполнение которой демонстрируется для разных наборов исходных данных.

Курсовая работа оценивается дифференцированно. Студенты, не защитившие работу, к сдаче экзамена не допускаются. Преподаватель может вернуть курсовую работу на доработку в случае неправильного ее оформления или невыполнения должным образом задания курсовой работы. Не зачтенная работа перерабатывается и сдается на повторную проверку.

Максимальный балл за курсовую работу выставляется в том случае, если студент полностью реализовал функциональные требования к программе, при вводе данных осуществляется проверка вводимых значений или вывод диагностических сообщений об ошибке. Тестирование не выявляет случаев сбоя программы при неверном вводе значений. Интерфейс программы удобен и понятен. Студент грамотно подготовил пояснительную записку, ответил на все вопросы преподавателя.

Средний балл за курсовую работу выставляется в том случае, если студент полностью реализовал функциональные требования к программе, но выполнил некоторые задания частично (степень выполнения не менее 70%), и защитил свою работу преподавателю, ответив не менее чем на 70% его вопросов.

Минимальный балл за курсовую работу выставляется в том случае, если студент полностью выполнил все задания своего варианта, но смог ответить на 50% вопросов преподавателя и продемонстрировал минимум умений по работе с изучаемым языком программирования. Либо, студент выполнил задание частично, и защитил свою работу преподавателю, ответив на более чем 50% вопросов.

Балл за курсовую работу не выставляется, если студент полностью или частично выполнил задания своего варианта, но не ответил на вопросы преподавателя и не продемонстрировал практических умений.

4.4 Вопросы к экзамену:

1. Элементарные конструкции языка программирования Python. Требования к идентификаторам программных единиц (переменных, констант, функций и т.д.). Структура программы. Комментарии.

2. Типы данных в Python. Динамическая типизация. Преобразование типов данных.

3. Арифметические операции в Python. Операции над целыми и вещественными числами.

4. Оператор присваивания в Python. Команды ввода-вывода данных.

5. Функции в Python. Глобальные и локальные переменные. Создание функции.

6. Логический тип данных в Python. Операции отношения. Логические операции. Условный оператор в Python.

7. Циклы с заданным числом повторений и по условию в Python.

8. Структуры данных. Списки в Python. Операции со списками и основные методы списков.

9. Структуры данных. Списки списков в Python (матрицы). Создание и обработка.

10. Структуры данных. Строки в Python. Операции со строками и основные методы строк.

11. Структуры данных. Кортежи в Python. Операции с кортежами и методы кортежей.

12. Структуры данных. Словари в Python. Операции со словарями и методы словарей.

13. Структуры данных. Работа с файлами.

14. Модуль graph. Создание графических примитивов.

15. Модуль graph. Создание анимированных изображений.

Понятие ООП. Основные принципы ООП.

15. Описание классов в Python. Члены класса: поля класса, поля экземпляра класса, конструктор, методы. Структура метода. Параметры метода. Параметр self.

16. Реализация принципов ООП в Python.

17. Событийно – ориентированное программирование. Программы с графическим интерфейсом.

18. Модуль tkinter. Создание компонентов (виджетов): кнопка, метка (надпись), текст однострочный и многострочный. Свойства и методы этих виджетов.

19. Модуль tkinter. Создание компонентов (виджетов): рамка, флажки и радиокнопки, списки. шкала, окно верхнего уровня. Свойства и методы этих виджетов.

20. Модуль tkinter. Менеджеры геометрии.

21. Модуль tkinter. Метод bind. Типы переменных и события.

4.5 При оценивании ответа используются показатели: правильность и полнота ответа на экзаменационные вопросы, а также правильность решения задания.

Если замечаний нет, студент получает отличную оценку.

Если ответ неполный, либо содержит неточности или небольшие ошибки, дальнейшая работа со студентом по итоговой аттестации ведется с учетом его активности в течение семестра (по результатам выполнения лабораторных работ), а также с учетом его посещаемости аудиторных занятий. При слабой активности и/или низкой посещаемости выставляется результирующая оценка – 3 или 4 в зависимости от качества ответа. Если

студент работал в течение семестра хорошо, проводится его дополнительный устный опрос, позволяющий, возможно, повысить ему оценку.

При низком качестве ответа на экзаменационный билет знания студента оцениваются неудовлетворительно, и ему предлагается прийти на пересдачу экзамена.

Экзаменационная оценка («отлично», «хорошо», «удовлетворительно» или «неудовлетворительно») выставляется в соответствии с критериями, указанными в табл. 2.

Таблица 2 – Система и критерии оценивания экзаменационного тестирования

Система оценок / Критерий	2	3	4	5
	0-40%	41-60%	61-80 %	81-100 %
	«неудовлетвори-тельно»	«удовлетвори-тельно»	«хорошо»	«отлично»
Системность и полнота знаний в отношении изучаемых объектов	Обладает частичными и разрозненными знаниями, которые не может научно- корректно связывать между собой (только некоторые из которых может связывать между собой)	Обладает минимальным набором знаний, необходимым для системного взгляда на изучаемый объект	Обладает набором знаний, достаточным для системного взгляда на изучаемый объект	Обладает полнотой знаний и системным взглядом на изучаемый объект

5 СВЕДЕНИЯ О ФОНДЕ ОЦЕНОЧНЫХ СРЕДСТВ И ЕГО СОГЛАСОВАНИИ

Фонд оценочных средств для аттестации по дисциплине «Программирование» представляет собой компонент основной профессиональной образовательной программы бакалавриата по направлению подготовки 09.03.03 Прикладная информатика, профиль «Прикладная информатика в экономике».

Фонд оценочных средств рассмотрен и одобрен на заседании кафедры систем управления и вычислительной техники 25.04.2022 г. (протокол № 5).

Заведующий кафедрой  В.А. Петрикин

Приложение № 1

ТЕСТОВЫЕ ВОПРОСЫ И ЗАДАНИЯ ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ»

Вариант 1.

№ вопроса	***Формулировка тестового задания***
1	Ключевое слово, которое используется при создании функции: 1. def 2. function 3. procedure 4. close
2	Метод, который удаляет элементы из списка: 1. remove 2. clear 3. append 4. extend
3	Инструкция управления циклом в языке Python: 1. if 2. else 3. break 4. go to
4	Верные выражения из нижеприведенных: 1. Элементы списка могут принимать значения одного типа 2. Длина списка может только увеличиваться 3. Список — это динамическая структура 4. Значения элементов списка постоянны
5	Цикл, в котором условие проверяется после окончания выполнения операторов цикла и цикл работает пока условие **неверно** – это цикл: 1. с параметром 2. с предусловием 3. с постусловием
6	x = 23 num = 0 if x > 10 else 11 print (num) Рез*у*льтатом выполнения фрагмента программы будет: 1. 0. 2. 23 3. 10 4. 11

7	count = 0
	for i in range(n):
	if a[i] %2 ==0:
	count +=1
	print (count)
	Фрагмент программы предназначен:

№ вопроса	*Формулировка тестового задания*
	1. Для подсчета четных элементов списка 2. Для подсчета нечетных элементов списка 3. Для подсчета всех элементов списка 4. Для нахождения суммы элементов списка
8	В результате выполнения фрагмента программы получим следующие числа: a = [i for i in range (10) if i % 2 == 0] 1. 0 2 4 6 8 2. 1 2 3 4 5 6 7 8 9 3. 0 1 2 3 4 5 6 7 8 4. 8 6 4 2 0
9	Неизменяемым типом данных из перечисленных является: 1. Список 2. Файл 3. Кортеж 4. Множество
10	Для замены в строке s всех символов * на символ пробел используется инструкция*:* 1. s=s.replace(‘ ‘, ’*’, 1) 2. s=s.replace(‘*‘, ’ ’,1) 3. s=s.replace(‘ *‘, ’ ’) 4. s=s.replace(‘ ‘, ’*’)
11	Метод класса, который вызывается для создания объекта этого класса: 1. Мастер 2. Конструктор 3. Специалист *4.* Инициатор
12	Дан фрагмент программы: *penColor ('blue")* *brushColor("yellow")* *rectangle(10,20,60,50)* будет нарисован объект: 1. Прямоугольник синего цвета внутри желтый 2. Прямоугольник желтого цвета синий внутри 3. Треугольник желтого цвета *4.* Треугольник синего цвета
13	Объекты в программе для обмена данными друг с другом используют: 1. Интерфейс 2. Диалог 3. Естественный язык 4. Глобальные переменные
14	Подход к программированию, при котором программа представляет собой множество взаимодействующих объектов, каждый из которых является

№ вопроса	*Формулировка тестового задания*
	экземпляром определенного класса, а классы образуют иерархию наследований: 1. Процедурное программирование 2. Функциональное программирование 3. Объектно-ориентированное программирование 4. Структурное программирование
15	Класс, который не предназначен для создания объектов (экземпляров), а предназначен только для создания наследников: 1. Одиночный 2. Абстрактный 3. Пустой 4. Базовый
16	После выполнения команды будет создан следующий тип данных: A={} 1. Словарь 2. Множество 3. Кортеж 4. Список
17	words = ['hello', 'daddy', 'hello', 'mum'] mn=set(words) В множестве mn будет следующее количество элементов: 1. 4 2. 3 3. 2 4. 5
18	Метод a.intersection(b) выполняет следующую операцию над множествами : 1. Объединение множеств 2. Пересечение множеств 3. Сложение множества 4. Вычитание множеств
19	Неизменяемым типом данных из перечисленных является: 1. set 2. list 3. frozenset 4. dict
20	Любые (не синтаксические) ошибки, которые могут возникнуть при выполнении, программы называются: 1. баг 2. исключение 3. опечатка 4. глюк
21	Исключение ValueError возникает в случае:

№ вопроса	*Формулировка тестового задания*
	1. когда встроенная операция или функция получают аргумент, тип которого правильный, но неправильное значение; 2. когда второй аргумент операции деления равен нулю; 3. когда операция или функция применяется к объекту несоответствующего типа. 4. Когда значение переменной превышает допустимое.
22	Ассоциативный массив – это следующий тип данных: 1. Список 2. Множество 3. Файл 4. Словарь
23	Для доступа к данным в словаре используется: 1. Индекс 2. Ключ 3. Порядковый номер 4. Значение
24	Ключ ‘a’ при открытии файла указывает на открытие файла со следующей целью: 1. Для записи 2. Для чтения 3. Для добавления информации 4. Для удаления
25	Выберите верное утверждение из следующих: 1. По окончании работы программы все файлы закрываются автоматически 2. По окончании работы программы все файлы автоматически удаляются 3. Файл остается открытым поле завершения программы, если не была использована инструкция close()
26	В какой тип данных читает информацию из текстового файла метод read(): 1. В список 2. В строку 3. В множество 4. В другой файл
27	Способность классов наследников по-разному реализовать метод базового класса называется: 1. Полиморфизм 2. Абстракция 3. Инкапсуляция 4. Наследование
28	Функция, которая запускает режим рисования и открывает графическое окно: 1. go() 2. run() 3. draw()

№ вопроса	*Формулировка тестового задания*
29	На холсте для рисования точка с координатами 0,0 находится: 1. в верхнем левом углу 2. в верхнем правом углу 3. в нижнем правом углу 4. в нижнем левом углу
30	Какой цвет линии определяет команда penColor (0,0,0): 1. Красный 2. Синий 3. Белый 4. Черный

Вариант 2.

№ вопроса	*Формулировка тестового задания*
1	Возведение всех элементов списка a в куб выполняет команда: 1. a = a + [x ** 3 for x in a] 2. a = [x ** 3 for x in a] 3. a = [x * 3 for x in a] 4. a = [for x ** 3 in a]
2	Список B состоящий из элементов списка A, отсортированных по возрастанию получается после выполнения следующей команды: 1. B = sorted (A, reverse = True) 2. A = sorted (B) 3. B = sorted (A) 4. B = sorted (A, reverse = False)
3	При поиске максимума среди целых чисел в диапазоне от -100 до 100 начальное значение переменной max может быть равно: 1. 0 2. -101 3. 100 4. -99
4	n=3 a = [1,3,4] * n Список, полученный таким образом, будет содержать следующее количество элементов: 1. 3 2. 6 3. 9 4. 12
5	Верным утверждением из перечисленных является: 1. Элементы символьной строки можно изменять, обращаясь к ним по их индексам 2. Квадратной называется матрица, в которой количество строк не равно количеству столбцов

№ вопроса	*Формулировка тестового задания*
	3. Элемент матрицы имеет два индекса 4. Кортеж — это изменяемый тип данных
6	Элементы списка после выполнения фрагмента программы: *a=[5,4,3,2,1]* *n=len(a)* *for i in range(n):* *a[i] +=i* 1. [5,5,5,5,5] 2. [0,1,2,3,4] 3. [4,3,2,1,0]
7	Для перехода на новую строку в текстовых файлах используется: 1. /r 2. /n 3. /t 4. /p
8	Метод a.remove (15): 1. Удаляет из списка первое значение равное 15 2. Удаляет из списка все значения равные 15 3. Перемещает в начало списка все элементы равные 15 4. Перемещает в конец списка все элементы равные 15
9	Значение i после исполнения фрагмента программы будет равно: i, n = 0, 625 while n > 0: n = n // 5 i = i +1 1. 625 2. 100 3. 0 4. 5
10	В результате выполнения фрагмента программы будет напечатано: *a = [4, 5, 7, 8, 10, 15]* *print (len (a))* 1. 4 2. 5 3. 6 4. 7
11	Значение переменной после выполнения фрагмента программы будет равно: a = (True +3)/(False +1) 1. True 2. False 3. 2 4. 1

№ вопроса	*Формулировка тестового задания*
12	При попытке открыть для записи несуществующий файл: 1. Указанный файл будет создан 2. Программа завершится с сообщением об ошибке 3. Возникнет исключение типа FileNotFoundError 4. Откроется другой файл программы
13	Выделение существенных характеристик объектов, отличающих его от других объектов, в ООП называется: 1. Абстракция 2. Декомпозиция 3. Классификация 4. Инициализация
14	Скрытие внутреннего устройства объектов или объединение в одном объекте данных и методов работы с ними: 1. Инкапсуляция 2. Полиморфизм 3. Абстракция 4. Наследование
15	Два символа, с которых начинаются имена закрытых полей и методов в Python 1. Знак + 2. Подчеркивание 3. Знак = 4. Минус
16	Метод a**.difference**(b) выполняет следующую операцию над множествами: 1. Объединение множеств 2. Пересечение множеств 3. Сложение множества 4. Вычитание множеств
17	Исключение ZeroDivisionError возникает в случае: 1. когда встроенная операция или функция получают аргумент, тип которого правильный, но неправильное значение; 2. когда второй аргумент операции деления равен нулю; 3. когда операция или функция применяется к объекту несоответствующего типа. 4. Когда значение переменной превышает допустимое.
18	Команда d=dict() создаст следующий тип данных: 1. Список 2. Строку 3. Словарь 4. Множество
19	Перебор элементов словаря выполняется следующим циклом: 1. for key in d: 2. for i in range(d): 3. for i in range(len(d)):

№ вопроса	*Формулировка тестового задания*
20	Метод values() выполняет следующее действие: 1. Возвращает пары ключ-значение словаря 2. Возвращает ключи словаря 3. Возвращает значения словаря 4. Возвращает все индексы словаря
21	Ключ ‘r’ при открытии файла указывает на открытие файла со следующей целью: 1. Для записи 2. Для чтения 3. Для добавления информации 4. Для удаления
22	Метод readline() читает информацию из текстового файла в следующий тип данных: 1. В список 2. В строку 3. В множество 4. В другой файл
23	Нумерация индексов в списках начинается с: 1. 1 2. 0 3. Буквы a 4. 100
24	penColor("red") brushColor("green") circle(50, 30, 20) Данный фрагмент программы нарисует следующую фигуру: 1. Желтый прямоугольник с красным контуром 2. Зеленую окружность с красным контуром 3. Желтый прямоугольник с зеленым контуром 4. Желтый треугольник с зеленым контуром
25	Метод s.upper() выполняет следующее действие: 1. Все символы строки переходят в верхний регистр 2. Все символы строки переходят в нижний регистр 3. Складывает строки 4. Удаляет лишние пробелы из строки
26	Метод, с помощью которого можно сформировать из списка строку называется: 1. replace() 2. str() 3. join() 4. split()
27	Метод списков append(x) выполняет следующее действие: 1. Добавляет элемент x в конец списка

№ вопроса	*Формулировка тестового задания*
	2. Добавляет элемент x в начало списка 3. Удаляет элемент x из списка 4. Удваивает значение элемента x
28	Виджетами или компонентами называются: 1. Любые изображения на экране 2. Элементы классов 3. Графические элементы интерфейса 4. Переменные, используемые в программе
29	Менеджер геометрии при создании графического интерфейса отвечает за: 1. Размеры главного окна графического интерфейса 2. Способ расположения виджетов на экране 3. Вызов библиотеки для работы с графическими изображения 4. Определение размеров виджетов относительно размеров экрана
30	lab = Label (root, text=«ПРИВЕТ! \n Из двух строк.", font="Arial 18") Эта команда предназначена для: 1. Создания метки (надписи) 2. Создания кнопки 3. Создания поля для ввода текста 4. Создания радиокнопки

Вариант 3.

№ вопроса	*Формулировка тестового задания*
1	Инструкция для открытия в переменной f файла input.txt для чтения имеет следующий вид: 1. f=open (‘input.txt’) 2. f=open (‘input.txt’,’w’) 3. f=open(‘input.txt’,’a’) 4. f=open (‘r’, ‘input.txt’)
2	Переменные логического типа могут принимать следующие значения: 1. -10 2. 10 3. True 4. ‘False’
3	Функция, которая определяет количество символов в строке: 1. count() 2. len() 3. split() 4. upper()
4	Верные утверждения: Кортеж — это список, который ... 1. нельзя изменить 2. использует столько же памяти как список

№ вопроса	*Формулировка тестового задания*
	3. использует больше памяти 4. заключен в квадратные скобки
5	Значение строки s1 после выполнения фрагмента программы: *s = "0123456789"* *s1 = s[3:8]* 1. "34567" 2. "23456" 3. "345678" 4. "56789"
6	Результате выполнения фрагмента программы: a= [5,7,4,9] print (a[-2]) 1. 5 2. 12 3. 7 4. 4
7	Количество повторений тела цикла: while False: <тело цикла> 1. Бесконечно 2. 15 раз 3. 1 раз 4. Ни одного раза
8	Результате выполнения фрагмента программы: A = [1, 2, 3, 4, 5] print (3 in A) 1. 1 2. 0 3. True 4. False
9	Возможность классов-наследников по-разному реализовывать метод базового класса называется: 1. Инкапсуляция 2. Полиморфизм 3. Абстракция 4. Наследование
10	Функция для запуска рисования и открытия графического окна: 1. run() 2. pack() 3. draw() 4. start()

№ вопроса	***Формулировка тестового задания***
11	Блок данных определенной структуры, который используется для обмена информацией между объектами: 1. Письмо 2. Пакет 3. Сообщение 4. Послание
12	Назначение конструкции: ***try*** ***опасные команды*** ***except*** ***обработка ошибки*** 1. для обработки исключений 2. для подключения графического интерфейса 3. для подключения графики 4. для определения типа ошибки
13	Команда для определения цвета заливки замкнутого контура: 1. penColor(цвет) 2. brushColor (цвет) 3. Color() 4. setColor()
14	Пустое множество: 1. a=set() 2. a=dict() 3. a=tuple() 4. a={}
15	Типы файлов в Python: 1. Текстовые и двоичные 2. Двоичные и исполняемые 3. Типизированные и исполняемые 4. Звуковые и графические
16	Метод a**.union**(b) выполняет следующую операцию над множествами: 1. Объединение множеств 2. Пересечение множеств 3. Сложение множества 4. Вычитание множеств
17	Метод **a.discard**(elem) выполняет следующее действие: 1. Удаляет все элементы множества 2. Проверяет наличие элемента в множестве 3. Удаляет элемент, если он находится в множестве 4. Добавляет элемент в множество
18	Текстовые файлы относятся к файлам: 1. Прямого доступа 2. Последовательного доступа

№ вопроса	*Формулировка тестового задания*
	3. Индексированного доступа 4. Доступа по выбору
19	Ключ ‘w’при открытии файла указывает на открытие файла со следующей целью: 1. Для записи 2. Для чтения 3. Для добавления информации 4. Для удаления
20	В какой тип данных читает информацию из текстового файла метод readlines(): 1. В список 2. В строку 3. В множество 4. В другой файл
21	Результатом работы метода s.count(x) будет значение следующего типа данных: 1. Строка 2. Целое число 3. Вещественное число 4. Список
22	Метод, который переводит первый символ строки в верхний регистр, а все остальные в нижний называется: 1. title() 2. capitalize() 3. upper() 4. low()
23	Метода для записи данных в файл называется: 1. read() 2. close() 3. write() 4. readlines()
24	Метод, предназначенный для удаления элемента в списке, называется: 1. pop() 2. clear() 3. reverse() 4. count()
25	Метод для работы со словарями d.keys() возвращает: 1. Все значения словаря 2. Все ключи словаря 3. Все пары ключ-значение 4. Все индексы словаря
26	Метод для работы с множествами a.symmetric_difference(b) возвращает: 1. Элементы, которые есть с обоих множествах

№ вопроса	*Формулировка тестового задания*
	2. Элементы, которые есть как в множестве, а так и в множестве b 3. Элементы, встречающихся в одном множестве, но не встречающиеся в обоих. 4. Элементы множества a, не принадлежащие множеству b.
27	Строчка кода root=Tk() в программе с графическим интерфейсом предназначена для: 1. Создания экземпляра класса Tk 2. Инициации работы графического интерфейса 3. Вызова библиотеки tkinter 4. Создания виджетов на экране
28	Операция конкатенации – это: 1. Дублирование строки 2. Слияние строк 3. Удаление строки 4. Удаление пробелов из строки
29	s = "abc" print (s.isdigit()) В результате выполнения следующего фрагмента программы будет выведено значение: 1. 3 2. True 3. False 4. 0
30	Функция x=int(s, n) предназначена для: 1. Для перевода строки s в целое десятичное число x из системы счисления с основанием n 2. Для перевода строки s в целое число x в системе счисления с основанием n 3. Для перевода числа s в строку x из n символов 4. Для перевода числа x в строку s из n символов