Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КАЛИНИНГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

В. В. Подтопельный

СКРИПТОВЫЕ ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Учебно-методическое пособие по выполнению
лабораторных работ по дисциплине для студентов специальности
10.05.03 «Информационная безопасность автоматизированных систем»
специализация «Безопасность открытых информационных систем»

Калининград
Издательство ФГБОУ ВО «КГТУ»
2022

УДК 004.4 (075)

Рецензент
доцент кафедры информационной безопасности института информационных технологий ФГБОУ ВО «Калининградский государственный технический университет»
А. Г. Жестовский

Подтопельный, В. В.

Скриптовые языки программирования: учебно-методическое пособие по выполнению лабораторных работ по дисциплине для студентов специальности 10.05.03 «Информационная безопасность автоматизированных систем». – Калининград: Изд-во ФГБОУ ВО «КГТУ», 2022. – 21 с.

Учебно-методическое пособие включает в себя рассмотрение практических вопросов в области защиты информации по дисциплине «Скриптовые языки программирования». В учебно-методическом пособии приведен перечень лабораторных работ для изучения и закрепления материала дисциплины. Представлены методические указания по изучению дисциплины. Пособие подготовлено в соответствии с требованиями утвержденной рабочей программы модуля.

Пособие предназначено для студентов специальности 10.05.03 «Информационная безопасность автоматизированных систем» и смежных специальностей.

Табл. 1, список лит. – 5 наименований

Учебно-методическое пособие рассмотрено и одобрено в качестве электронного методического материала кафедрой информационной безопасности 19 мая 2022 г., протокол № 7

Учебно-методическое пособие по выполнению лабораторных работ по дисциплине рекомендовано к использованию в учебном процессе в качестве локального электронного методического материала методической комиссией института цифровых технологий ФГБОУ ВО «Калининградский государственный технический университет» 28 июня 2022 г., протокол № 4

УДК 004.4 (075)

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Данное учебно-методическое пособие предназначено для студентов специальности 10.05.03 «Информационная безопасность автоматизированных систем», специализация: «Безопасность открытых информационных систем», изучающих дисциплину «**Скриптовые языки программирования**».

Цель изучения лабораторного курса дисциплины обучить студентов использовать языки скриптого типа.

Лабораторный практикум содержит 9 лабораторные работ.

Лабораторные работы проводятся в компьютерном классе.

В результате выполнения лабораторных работ ожидается, что студенты сформируют навыки применения:

– построение алгоритмов;

– принципов программирования;

– освоениея ситнаксиса скриптовых языков;

– способов защиты трафика от изучения, разрушающих программных действий и изменений.

Программное обеспечение:

1. Microsoft Desktop Education. Операционные системы: Microsoft Windows Desktop operating systems, офисные приложения (Microsoft Office), по соглашению V9002148 Open Value Subscription (срок действия: три года)

2. Программное обеспечение, распространяемое по лицензии GNU General Public License (лицензия на свободное программное обеспечение, созданная в рамках проекта GNU, по которой автор передаёт программное обеспечение в общественную собственность):

1. JetBrains PyCharm Community Edition;
2. IDLE.

Типовое ПО на всех ПК:

1. Microsoft Desktop Education (операционные системы Microsoft Windows Desktop operating system, офисные приложения Microsoft Office, по соглашению V9002148 Open Value Subscription). Дата заключения контракта 05.07.2018. Номер контракта 0335100016118000073-0484577-02.

2. Антивирусное программное обеспечение Kaspersky Total Space Security Russian Edition, лицензия 17EO-171225-104659-470-270, срок использования с 2017-12-26 до 2020-03-13.

Таблица. Шкала оценок уровня

Оценка			
неудовлетворительный	пороговый	углубленный	продвинутый
«2» (неудовлетворительно)	«3» (удовлетворительно)	«4» (хорошо)	«5» (отлично)
Работа выполнена в полном объеме. Отчет не оформлен и представлен. При защите отчетных материалов правильные ответы даны менее чем на 50% включительно. Материал излагается непоследовательно, сбивчиво, не представляет определенной системы знаний по работе	Работа выполнена в полном объеме. Отчет оформлен и представлен. При защите отчетных материалов правильные ответы на 51–64 % вопросов. Допускаются нарушения в последовательности изложения. Демонстрируются поверхностные знания вопроса. Имеются затруднения с выводами. Допускаются нарушения норм литературной речи	Работа выполнена в полном объеме. Отчет оформлен и представлен. При защите отчетных материалов правильные ответы на 65–94 % вопросов. Ответы на поставленные вопросы излагаются систематизировано и последовательно. Материал излагается уверенно. Демонстрируется умение анализировать материал, однако не все выводы носят аргументированный и доказательный характер	Работа выполнена в полном объеме. Отчет оформлен и представлен. При защите отчетных материалов правильные ответы даны на 95–100 % вопросов. Ответы на поставленные в билете вопросы излагаются логично, последовательно и не требуют дополнительных пояснений. Делаются обоснованные выводы. Демонстрируются глубокие знания предмета

Лабораторная работа № 1. Простейшая программа и основы ввода-вывода информации в Python

Общие сведения

Цель: установить интегрированную среду разработки Python и научиться писать, редактировать, сохранять и запускать программы на Python.

Материалы, оборудование, программное обеспечение: ПК, ОС, JetBrains PyCharm Community Edition, IDLE текстовый редактор.

1. Теоретическое введение

Язык программирования Python – современный мощный высокоуровневый язык программирования, созданный голландским программистом Гвидо Ван Россумом и представленный в 1991 году. Своему названию язык обязан популярному комедийному телесериалу «Воздушный цирк Монти Пайтона», который транслировался по каналу BBC, что неизбежно добавляет юмора в примеры кода на языке Python.

Python испытал на себе влияние таких языков программирования, как Java, C, C++ (как отмечает сам автор – он использовал наиболее непротиворечивые конструкции из C, чтобы не вызвать неприязнь у C-программистов к новому языку. В то же время Python оказал влияние на язык программирования Ruby.

2. Задание к лабораторной работе

1. Установить интегрированную среду разработки *Python* (рекомендуется скачать установщик с официального сайта *www.python.org*).
2. Ознакомиться с базовой документацией по языку программирования *Python.*
3. Протестировать среду разработки *IDLE* в двух режимах: в интерактивном и сценарном посредством разработки приложения, образец которого приведен в разделе «Пример написания простейшей программы в среде разработке *IDLE*».
4. Изучить основные горячие клавиши для последующей быстрой разработки программ в сценарном режиме.

Требования к отчету и защите

Защита предполагает опрос по материалу, изложенному в отчете.

Отчет должен быть оформлен: требуется наличие титульного листа, указание цели лабораторной работы, последовательное изложение разработанного материала, подписанные скриншоты хода работы, если использовалась компьютерная техника.

Проверяется знание теоретического материала с учетом знаний ответов на контрольные вопросы, приведенные в пункте «Контрольные вопросы».

Задаются вопросы по пунктам, приведенным в пункте «Методические указания и порядок выполнения работы».

Критерии к оценке защиты отчета студентом приведены в пункте «Общие сведения».

Контрольные вопросы для самопроверки:

1. Что такое алгоритм?
2. Что такое язык программирования?
3. Что такое компьютерная программа?
4. Почему можно рассматривать *Python* как интерпретируемый, интерактивный и объектно-ориентированный язык программирования?
5. В чем особенности *Python*?

Лабораторная работа № 2. Переменные, простые типы данных и операции над ними

Общие сведения

Цель: научиться использовать стандартные функции ввода-вывода Python для написания интерактивных программ, организующих диалог между пользователем и компьютером.

Материалы, оборудование, программное обеспечение: ПК, ОС, JetBrains PyCharm Community Edition, IDLE текстовый редактор.

1. Теоретическое введение

Для достижения большей гибкости в отображении текста на экране в строки можно вставлять специальные escape-последовательности, которые не отображаются на экране и всегда начинаются с символа правого слеша ('\').

Если требуется вывести строку точно в таком же виде, как она описана в коде, без всевозможных преобразований символов *escape*-последовательностей, достаточно перед написанием самой строки поставить префикс в виде символа *r* (*raw* – запись/строка). Данный тип строки не преобразует слеши.

Функция для вывода информации на дисплей print(). Стандартная функция print() – это базовая функция для организации вывода одного или нескольких значений по умолчанию на консоль (для этого используется модуль sys.stdout) или в любой другой указанный поток.

Этапы разработки программы в среде PyCharm:

1) создание нового проекта «Create New Project»;
2) сохранение проекта – в поле «Location» (месторасположение) выбрать место на диске, где будет сохранен проект и его имя;
   – в поле «Interpreter» указать место расположения Python,
   – нажать кнопку «Create»;
3) создание файла-скрипта – в появившемся основном окне среды разработки в левой панели «Project» на проекте Lab2Project щелкнуть правой кнопкой мыши и из контекстного меню и выбрать

New→Python File;

4) название файла-скрипта – в появившемся диалоговом окне необходимо ввести имя файла-скрипта и нажать кнопку ОК;

5) написание текста программы – в основном окне файла-скрипта вводится исходный текст программы.

2. Задание к лабораторной работе

1. Установить специализированную среду разработки JetBrains PyCharm Community Edition.
2. В установленной среде разработки создать программу, выводящую на экран адрес расположения заданий для лабораторных работ на сервере и адрес любого электронного ресурса, содержащего программное обеспечение для разработки на языке программирования Python.
3. Разработать программу «Game Over», которая выводит соответствующую запись на экран монитора с внушительным видом. Художественное оформление выводимой информации ограничено только фантазией разработчика.

Требования к отчету и защите

Защита предполагает опрос по материалу, изложенному в отчете.

Отчет должен быть оформлен: требуется наличие титульного листа, указание цели лабораторной работы, последовательное изложение разработанного материала, подписанные скриншоты хода работы, если использовалась компьютерная техника.

Проверяется знание теоретического материала с учетом знаний ответов на контрольные вопросы, приведенные в пункте «Контрольные вопросы».

Задаются вопросы по пунктам, приведенным в пункте «Методические указания и порядок выполнения работы».

Критерии к оценке защиты отчета студентом приведены в пункте «Общие сведения».

Контрольные вопросы для самопроверки:

1. Какими способами отображаются строки в Python?
2. Зачем в Python используют тройные кавычки (апострофы)?
3. Зачем нужны escape-последовательности?
4. Зачем нужны raw-строки и как их записать в коде?
5. Что такое таблица кодировок?
6. Чем отличается ASCII-кодировка от Unicode-кодировки?
7. Какой стандарт (тип кодировки) используется для кодирования символьной информации в Python 3?
8. Опишите синтаксис функции print().
9. Опишите синтаксис функции input().

Лабораторная работа № 3. Условные конструкции Python

Общие сведения

Цель: освоить базовый синтаксис языка Python, простые типы данных, приобрести навыки создания интерактивных программ с использованием линейных алгоритмов.

Материалы, оборудование, программное обеспечение: ПК, ОС, JetBrains PyCharm Community Edition, IDLE текстовый редактор.

1. Теоретическое введение

Информация, сохраненная в памяти компьютера, может быть разных типов данных. К стандартным типам данных, используемым в Python, относятся: число (Number); строка (String); список (List); кортеж (Tuple); словарь (Dictionary); множество (Set).

Числовой тип данных в Python предназначен для хранения числовых значений и представляет собой неизменяемый тип данных, то есть изменение значения числового типа приведет к созданию нового объекта в памяти (и удалению старого).

Для хранения и обработки текстовой информации используется строковый тип данных. В Python строки могут задаваться следующими способами:

1) строка в одинарных кавычках (апострофах);
2) строка в двойных кавычках;
3) строка в тройных кавычках;

2. Задание к лабораторной работе

1 Разработать интерактивную программу, демонстрирующую работу с простыми типами данны

2 Разработать интерактивную программу «What is My Age in Seconds» («Каков мой возраст в секундах»), которая на входе принимает дату рождения пользователя, рассчитывает количество прожитых пользователем секунд и выводит результат на экран монитора.

Требования к отчету и защите

Защита предполагает опрос по материалу, изложенному в отчете.

Отчет должен быть оформлен: требуется наличие титульного листа, указание цели лабораторной работы, последовательное изложение разработанного материала, подписанные скриншоты хода работы, если использовалась компьютерная техника.

Проверяется знание теоретического материала с учетом знаний ответов на контрольные вопросы, приведенные в пункте «Контрольные вопросы».

Задаются вопросы по пунктам, приведенным в пункте «Методические указания и порядок выполнения работы».

Критерии к оценке защиты отчета студентом приведены в пункте «Общие сведения».

Контрольные вопросы для самопроверки:

1. Опишите архитектуру и основные элементы компьютера. Какой элемент является центральным при построении любой вычислительной системы?
2. Какие типы памяти доступны при разработке программ?
3. Что такое машинный код?
4. В чем отличия языка высокого уровня от языка низкого уровня?
5. Какой язык понимает и обрабатывает центральный процессор (Central Process Unit, CPU)?
6. Что такое транслятор и что он делает?
7. Что общего между компилятором и интерпретатором и чем они отличаются?

Лабораторная работа № 4. Циклические конструкции. Итерационные алгоритмы. Цикл while Циклические конструкции. Цикл for

Общие сведения

Цель: изучить синтаксис циклической конструкции while языка Python для программирования итерационных алгоритмов, продемонстрировать возможности конструкции while, for на примере разработки интерактивных приложений.

Материалы, оборудование, программное обеспечение: ПК, ОС, JetBrains PyCharm Community Edition, IDLE текстовый редактор.

1. Теоретическое введение

Логическими (boolean expression) называются выражения, которые могут принимать одно из двух значений – истина или ложь. Операнды логических операторов должны быть логическими выражениями. В Python любое ненулевое число интерпретируется им как «истинное».

Не существует ограничения на число инструкций, которые могут встречаться в теле if, но хотя бы одна инструкция там должна быть. Иногда полезно иметь тело if без инструкций (обычно оставляют место для кода, который еще не написан).

Программы, генерирующие одни и те же выходные значения для одинаковых входных значений, называются детерминированными. Для многих приложений, например игр, характерна непредсказуемость в поведении. Для создания недетерминированных программ можно использовать алгоритмы генерации псевдослучайных чисел.

2. Задание к лабораторной работе:

1. Разработать интерактивную программу «Quadric Equation» («Квадратное уравнение») для решения квадратных уравнений вида: ax2 + bx + c = 0. Программа должна запрашивать соответствующие параметры a, b и c, проверять параметры и выдавать результат.

2. Запрограммировать последовательность чисел Фибоначчи (пользователь вводит порядковый номер элемента последовательности Фибоначчи, а программа выводит на экран значение).

Контрольные вопросы для самопроверки:

1. Для чего используются циклы? Что такое итерация?
2. Какие разновидности циклов существуют?
3. Описать Python-синтаксис цикла с предусловием while.
4. Чем является выражение после ключевого слова while – инициализацией, условием или обновлением?
5. Какова роль оператора break в теле цикла?
6. Какова роль оператора continue в теле цикла?
7. Какова роль оператора pass в теле цикла?

Лабораторная работа № 5. Функции. Рекурсивные алгоритмы Модули и пакеты

Общие сведения

Цель: познакомиться с наиболее востребованными стандартными функциями Python, изучить синтаксис объявления и использования пользовательских функций; познакомиться с рекурсивными алгоритмами; научиться проектировать и разбивать большую программу на мелкие фрагменты (функции); закрепить знания на примере разработки интерактивных приложений.

Материалы, оборудование, программное обеспечение: ПК, ОС, JetBrains PyCharm Community Edition, IDLE текстовый редактор.

1. Теоретическое введение

В Python предоставляется возможность создавать свои собственные (пользовательские) функции, а также работать со встроенными стандартными функциями (функции ввода/вывода данных, функции преобразования типов, математические функции модуля math, функции генерации псевдослучайных функций модуля random).

Правила наименования функций такие же, как для переменных, например, нельзя использовать зарезервированные слова в качестве имен функций. Имена функций и переменных не должны совпадать. Первая строка определения функции называется заголовком (header), оставшаяся часть – телом (body) функции. Заголовок заканчивается двоеточием, тело функции имеет отступ. Тело функции может содержать любое количество инструкций. Инструкции внутри функции не получат управления, пока функция не будет вызвана. Для возврата результата функции, используется инструкция return. Вызвать функцию (function call) можно, обратившись к ней по имени.

2. Задание к лабораторной работе:

1. Рекурсивно описать функцию f(x, n), вычисляющую xn/n! при любом действительном x и любом неотрицательном целом n.

2. Рекурсивно описать функцию pow(x, n), вычисляющую xn для любого действительного $x \neq 0$ и любого целого n.

3. Реализовать функцию, которая вычисляет N-й элемент ряда Фибоначчи с использованием рекурсивного алгоритма. На базе данной функции разработать программу, которая должна предлагать пользователю следующие возможности:

1) вывод конкретного элемента последовательности;

2) вывод всех элементов до указанного пользователем элемента;

3) вывод части последовательности, значение последнего элемента которой не превосходит введенного пользователем значения.

Требования к отчету и защите

Защита предполагает опрос по материалу, изложенному в отчете.

Отчет должен быть оформлен: требуется наличие титульного листа, указание цели лабораторной работы, последовательное изложение разработанного материала, подписанные скриншоты хода работы, если использовалась компьютерная техника.

Проверяется знание теоретического материала с учетом знаний ответов на контрольные вопросы, приведенные в пункте «Контрольные вопросы».

Задаются вопросы по пунктам, приведенным в пункте «Методические указания и порядок выполнения работы».

Критерии к оценке защиты отчета студентом приведены в пункте «Общие сведения».

Контрольные вопросы для самопроверки:

1. Что такое структурированное программирование?

2. Что такое функция? Как описывается функция в Python?

3. Зачем нужны функции?

4. Для чего используется оператор return в функциях? Как возвратить из функции несколько значений?

5. Чем формальные параметры отличаются от фактических?

6. Что такое позиционные параметры?

7. Что такое параметры по умолчанию?

Лабораторная работа № 6. Строки и обработка текстовой информации, Кортежи, списки, словари и множества

Общие сведения

Цель: приобрести навыки работы с Python строками и закрепить их на примере разработки интерактивных приложений.

Материалы, оборудование, программное обеспечение: ПК, ОС, JetBrains PyCharm Community Edition, IDLE текстовый редактор.

1. Теоретическое введение

Строка – это неизменяемая (immutable) последовательность символов. Доступ к одному символу можно получить с помощью оператора квадратных скобок.

Выражение в скобках называется индексом (index). Индекс – это смещение от начала строки; смещение для первого символа – нуль. Таким образом, нумерация в строках начинается с нуля. В качестве индекса можно использовать любое целочисленное выражение, включая переменные или операторы. Можно использовать отрицательные индексы, которые считаются с конца. fruit[-1] выдаст последний символ, fruit[-2] – второй с конца строки и так далее.

Множество алгоритмов включают посимвольную обработку строк, которая, как правило, начинается с начала строки. Такая обработка называется обходом (traversal). Обход может осуществляться с помощью различных циклов.

Срезом (slice) называется часть строки. Оператор [n:m] возвращает часть строки от n-го символа до m-го, включая первый, но исключая последний. Если опустить первый индекс, то срез будет начинаться с начала строки.

Метод вызывается аналогично функции, принимает аргументы и возвращает значение. Однако метод и функция различаются синтаксисом: метод вызывается путем добавления имени метода к имени переменной с использованием точки в качестве разделителя.

Список внутри другого списка является вложенным (nested). Список без элементов, называется пустым. Пустой список создается с помощью пустых квадратных скобок [ ].

Функция list() преобразует строку в список символов. Поскольку list() – это имя встроенной функции, следует избегать его использование в качестве имени переменной. Функция list() разбивает строку на отдельные буквы.

Словарь (dictionary) похож на список, но имеет более широкие возможности. В списке индекс имеет целочисленное значение, в словаре может быть любого типа. dict() создает словарь без записей.

2. Задание к лабораторной работе:

Написать программу «Анаграммы» («Anagrams»). Суть игры заключается в следующем: формируется группа слов в виде кортежа (tuple), компьютер случайным образом выбирает одно из слов, и случайным образом переставляет в нем буквы, а затем представляет пользователю (игроку). Цель игрока – угадать выбранное и «перемешанное» компьютером слово.

Разработать интерактивную программу, которая будет моделировать игру «Виселица» («Hangman»). Компьютер загадывает слово и выводит на консоль количество подчеркиваний, равное числу букв в загаданном слове. Пользователь начинает вводить буквы, чтобы отгадать слово. Если буква есть в слове, компьютер вписывает ее на свое место в слово, а если нет, компьютер рисует в

консоли один элемент импровизированной виселицы (к примеру: стойка, перекладина, веревка, голова, туловище, две руки, две ноги). Дополнительно выводятся буквы, которые уже вводились.

Если игрок не успел угадать слово раньше, чем компьютер нарисовал полностью виселицу, то игрок считается проигравшим. Если игрок успевает угадать слово, то он выигрывает.

Требования к отчету и защите

Защита предполагает опрос по материалу, изложенному в отчете.

Отчет должен быть оформлен: требуется наличие титульного листа, указание цели лабораторной работы, последовательное изложение разработанного материала, подписанные скриншоты хода работы, если использовалась компьютерная техника.

Проверяется знание теоретического материала с учетом знаний ответов на контрольные вопросы, приведенные в пункте «Контрольные вопросы».

Задаются вопросы по пунктам, приведенным в пункте «Методические указания и порядок выполнения работы».

Критерии к оценке защиты отчета студентом приведены в пункте «Общие сведения».

Контрольные вопросы для самопроверки:

1. Чем характеризуется строковый тип данных в Python?
2. Какие есть способы объявления строк в Python?
3. Что такое неизменяемость строк?
4. Зачем нужна индексация строк, и как ее использовать?
5. Зачем и как используются срезы строк?
6. Какие основные операторы используются для работы со строками?
7. Какие основные встроенные функции класса str используются для работы со строками?
8. Приведите примеры объявления каждого из высокоуровневых типов данных.
9. Объясните понятие «распаковка последовательности».
10. Какое главное отличие списков от кортежей? Когда лучше использовать кортежи, а когда – списки?
11. Что такое распределенные ссылки?
12. Как получить несколько ссылочных переменных на один и тот же список, а как получить полную копию списка?

Лабораторная работа № 7. Введение в основы Java script.

Общие сведения

Цель: получения навыков работы с скриптами Java.

Материалы, оборудование, программное обеспечение: ПК, ОС текстовый редактор.

1. Теоретическое введение

Самый простой способ внедрения JavaScript в HTML-документ – использование тега <script>. Теги <script> часто помещают в элемент <head>, и ранее этот способ считался чуть ли не обязательным. Однако в наши дни теги <script> используются как в элементе <head>, так и в теле веб-страниц.

Таким образом, на одной веб-странице могут располагаться сразу несколько сценариев. В какой последовательности браузер будет выполнять эти сценарии? Как правило, выполнение сценариев браузерами происходит по мере их загрузки. Браузер читает HTML-документ сверху вниз и, когда он встречает тег <script>, рассматривает текст программы как сценарий и выполняет его. Остальной контент страницы не загружается и не отображается, пока не будет выполнен весь код в элементе <script>.

Чтобы ваша первая программа (или сценарий) JavaScript запустилась, ее нужно внедрить в HTML-документ.

Сценарии внедряются в HTML-документ различными стандартными способами:

– поместить код непосредственно в атрибут события HTML-элемента;

– поместить код между открывающим и закрывающим тегами <script>;

– поместить все ваши скрипты во внешний файл (с расширением .js), а затем связать его с документом HTML.

JavaScript можно добавить в HTML-документ с помощью элемента <script> двумя способами:

Определить встроенный сценарий, который располагается непосредственно между парой тегов <script> и </script>.

Подключить внешний файл с JavaScript-кодом через <script src="путь"></script>.

Если JavaScript-код используется в нескольких страницах, то его лучше подключать в качестве внешнего сценария. Это существенно облегчает сопровождение и редактирование кода, а также ускорит загрузку и обработку веб-страниц – внешний сценарий загружается браузером всего один раз (в дальнейшем он будет извлекаться из кэша браузера).

Атрибут defer сигнализирует браузеру, что загрузку сценария можно начать немедленно, но его выполнение следует отложить до тех пор, пока весь HTML-документ будет загружен.

В тех случаях, когда файл скрипта содержит функции, взаимодействующие с загружаемым HTML-документом или существует зависимость от другого файла на странице необходимо, чтобы HTML-документ был полностью загружен, прежде чем скрипт будет выполнен. Как правило, такая ссылка на javaScript-сценарий помещается в низ страницы перед закрывающим тегом <body>, чтобы убедиться, что для его работы весь документ был разобран. Однако, в ситуации, когда по каким-либо причинам JS-файл должен быть размещён в другом месте документа – атрибут defer может быть полезен.

Атрибут defer сохраняет относительную последовательность выполнения скриптов, а async – нет.

Скрипт с атрибутом async выполняется асинхронно с обработкой страницы, когда скрипт будет загружен – он выполнится, даже если HTML-документ ещё не полностью готов.

Для JS-файлов, которые не зависят от других файлов, атрибут async будет наиболее полезен. Поскольку нам не важно, когда скрипт будет исполнен, асинхронная загрузка – наиболее подходящий вариант.

2. Задание к лабораторной работе

1. Разместите в теле HTML-страницы сценарий, выводящий всплывающее окно с надписью: "Привет, javascript!"
2. Разместите в теле HTML-страницы сценарий, выводящий всплывающее окно с надписью: "Привет, javascript!"

Требования к отчету и защите

Защита предполагает опрос по материалу, изложенному в отчете.

Отчет должен быть оформлен: требуется наличие титульного листа, указание цели лабораторной работы, последовательное изложение разработанного материала, подписанные скриншоты хода работы, если использовалась компьютерная техника.

Проверяется знание теоретического материала с учетом знаний ответов на контрольные вопросы, приведенные в пункте «Контрольные вопросы».

Задаются вопросы по пунктам, приведенным в пункте «Методические указания и порядок выполнения работы».

Критерии к оценке защиты отчета студентом приведены в пункте «Общие сведения».

Контрольные вопросы для самопроверки:

1. JavaScript можно добавить в HTML-документ с помощью элемента <script> двумя способами. Какими?
2. В какой последовательности браузер будет выполнять эти сценарии?
3. Сценарии внедряются в HTML-документ различными стандартными способами. Какими?

Лабораторная работа № 8. Введение в основы Bash

Общие сведения

Цель: получения навыков работы со скриптами Bash.

Материалы, оборудование, программное обеспечение: ПК, ОС.

1. Теоретическое введение

Bash (акроним от «Bourne-again SHell») это стандартный интерпретатор команд на большинстве линукс систем. В его обязанности входит обработка и исполнение команд с помощью которых пользователь управляет компьютером. После того как вы завершили работу, можно завершить процесс командного интерпретатора.

После нажатия клавиш Ctrl–D, команд exit или logout процесс командного интерпретатора будет завершен и наэкране снова появится приглашение ввести имя пользователя и пароль.

Запустить сценарий можно командой sh scriptname [5] или bash scriptname. (Не рекомендуется запуск сценария командой sh <scriptname>, поскольку это запрещает использование устройства стандартного ввода stdin в скрипте). Более удобный вариант – сделать файл скрипта исполняемым, командой chmod.

Это:

chmod 555 scriptname (выдача прав на чтение/исполнение любому пользователю в системе) [6]

или

chmod +rx scriptname (выдача прав на чтение/исполнение любому пользователю в системе)

chmod u+rx scriptname (выдача прав на чтение/исполнение только "владельцу" скрипта)

После того, как вы сделаете файл сценария исполняемым, вы можете запустить его примерно такой командой ./scriptname. Если, при этом, текст сценария начинается с корректной сигнатуры ("sha-bang"), то для его исполнения будет вызван соответствующий интерпретатор.

И наконец, завершив отладку сценария, вы можете поместить его в каталог /usr/local/bin (естественно, что для этого вы должны обладать правами root), чтобы сделать его доступным для себя и других пользователей системы. После этого сценарий можно вызвать, просто напечатав название файла в командной строке и нажав клавишу [ENTER].

У языка Bash нет системы типов. В нём все скалярные переменные хранятся в памяти как строки. Но в Bash есть составные типы – массивы. Они представляют собой комбинации строк.

Тип переменной (скалярная или составная) выбирается при её определении. Для этого надо указать метаинформацию, которая в Bash называется атрибутами. Кроме типа атрибуты определяют константность и область видимости переменной.

Чтобы указать атрибуты переменной, используйте встроенную команду declare. Если вызвать её без параметров, она выведет имена и значения всех

объявленных в данный момент переменных: локальных и окружения. Эту же информацию выводит команда set.

У команды declare есть опция -p. Она добавляет в вывод атрибуты переменных.

2. Задание к лабораторной работе:

Напишите сценарий, который выводит дату, время, список зарегистрировавшихся пользователей, и uptime системы и сохраняет эту информацию в системном журнале.

Требования к отчету и защите

Защита предполагает опрос по материалу, изложенному в отчете.

Отчет должен быть оформлен: требуется наличие титульного листа, указание цели лабораторной работы, последовательное изложение разработанного материала, подписанные скриншоты хода работы, если использовалась компьютерная техника.

Проверяется знание теоретического материала с учетом знаний ответов на контрольные вопросы, приведенные в пункте «Контрольные вопросы».

Задаются вопросы по пунктам, приведенным в пункте «Методические указания и порядок выполнения работы».

Критерии к оценке защиты отчета студентом приведены в пункте «Общие сведения».

Контрольные вопросы для самопроверки:

1. Как запустить сценарий в bash?
2. У языка Bash есть или нет системы типов?
3. Какой командой сделать файл скрипта исполняемым?
4. Чтобы указать атрибуты переменной, какую нужно использовать команду?

ЗАКЛЮЧЕНИЕ

Скриптовые языки быстро становятся языками общей реализации для многих областей, блистая там, где время разработчика более важно, чем время исполнения (и даже там, где важно время исполнения; например, благодаря встроенным операциям высокого уровня быстродействие программ, написанных на Python, такое же, или даже быстрее, чем программ, написанных на Java). Скриптовые языки обладают более сложным инструментарием и поддерживают более прогрессивные техники программирования. То, что в язык встроены все основные инструменты программирования, избавляет от необходимости создавать их самостоятельно и означает, что для решения конкретной проблемы нужно писать меньше кода, что увеличивает производительность разработчика. Скриптовые языки позволяют быстро выполнять доработку кода без раздражающей потери времени на ожидание окончания компиляции.

Время исполнения все еще является главной проблемой. Конечно, есть области, где скорость слишком важна, чтобы можно было программировать непосредственно на скриптовом языке. Эта проблема обычно решается тем, что код тщательно выбранной части приложения (скажем, 10–30 %) пишется на языке низкого уровня (таком, как C или C++); например, в Python есть развитые механизмы для того, чтобы вставить такой код (как и в большинстве других динамических языков). Общей проблемой всех скриптовых языков является отсутствие хорошей интегрированной среды разработки (IDE). Конечно, какие-то интегрированные среды разработки существуют, однако в них недостает мощности, как у Visual Studio. Ключевым нетехническим, однако важным недостатком является отсутствие маркетингового бюджета.

ЛИТЕРАТУРА

Основная учебная литература

1. Шкаберина, Г. Ш. Программирование. Основы языка Python: учеб. пособие / Г. Ш. Шкаберина, Н. Л. Резова. – Красноярск: СибГУ им. академика М. Ф. Решетнёва, 2018. – 92 с. – URL: https://e.lanbook.com/book/147450. – Б. ц. – Текст: электронный.

2. Копырин, А. С. Программирование на Python: учеб. пособие для студентов специальности 09.03.03 «Прикладная информатика (в экономике)» / А. С. Копырин, Т. Л. Салова. – Сочи: СГУ, 2018. – 48 с. – URL: https://e.lanbook.com/book/147665. – Б. ц. – Текст: электронный.

3. Воронина, В. В. Теория и практика машинного обучения: учеб. пособие / В. В. Воронина. – Ульяновск: УлГТУ, 2017. – 290 с. – URL: https://e.lanbook.com/book/165053. – ISBN 978-5-9795-1712-4: Б. ц. – Текст: электронный.

Дополнительная учебная литература

1. Дэвид М. Бизли. Python. Подробный справочник, 4-е издание. – Перевод с английского. – Санкт-Петербург: Символ-Плюс, 2010. – 864 с. – ISBN 978-5-93286-157-8.

2. Марк Саммерфилд. Программирование на Python 3. Подробное руководство. – Перевод с английского. – Санкт-Петербург: СимволПлюс, 2009. – 608 с. – ISBN 978-5-93286161-5.

Локальное электронное издание

Подтопельный Владислав Владимирович

СКРИПТОВЫЕ ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Редактор С. Кондрашова

Уч.-изд. л. 1,7. Печ. л. 1,4.

Федеральное государственное
бюджетное образовательное учреждение высшего образования
«Калининградский государственный технический университет»,
236022, Калининград, Советский проспект, 1