УДК 002

ББК

В

Рецензенты:

Травкин Евгений Иванович — кандидат педагогических наук, доцент кафедры информационной, техносферной безопасности и правовой защиты информации Курской академии государственной и муниципальной службы

Тарасюк Владимир Борисович — кандидат технических наук, доцент кафедры компьютерных технологий и информатизации образования Курского государственного университета

Васильев Д.А.

В Основы программирования на языке PYTHON. Васильев Д.А.: учеб.-метод. пособие. – Курск. , 2015. – 94 с.

Рассматриваемые в предлагаемом пособии лабораторные работы предусматривают изучение основ языка программирования Python. В пособии предложены к рассмотрению основные конструкции языка Python, рассмотрены основные типы данных, используемые в языке программирования Python, типовые алгоритмы их обработки. В пособии рассмотрены возможность программирования на языке Python.

Пособие предназначено для организации учебного процесса бакалавров по направлению подготовки «Информационная безопасность» при изучении курсов «Технологии и методы программирования», «Программирование» и других учебных курсов связанных с изучением основ программирования.

ББК

Содержание

Введение

Python — мощный и простой для изучения язык программирования. В нём предоставлены проработанные высокоуровневые структуры данных и простой, но эффективный подход к объектно-ориентированному программированию. Сочетание изящного синтаксиса и динамической типизации, совмещённых с интерпретируемой сущностью, делает Python наиболее подходящим языком для написания сценариев и ускоренной разработки приложений в различных сферах и на большинстве платформ.

Интерпретатор Python и разрастающаяся стандартная библиотека находятся в свободном доступе в виде исходных кодов и бинарных файлов для всех основных платформ на официальном сайте Python http://www.python.org и могут распространяться без ограничений по лицензии GPL. Кроме этого на сайте содержатся дистрибутивы и ссылки на многочисленные модули третьих сторон для языка Python, различные программы и инструменты, а также дополнительная документация.

Интерпретатор Python может быть легко расширен с помощью новых функций и типов данных, написанных на C/C++ (или других языков, к которым можно получить доступ из C). Также Python можно применять как язык расширений для настраиваемых приложений.

Лабораторные работы данного пособия предназначены для выполнения в среде Eclipse и расширении PyDev среды Eclipse, однако могут быть выполнены в IDLE Python.

В ходе выполнения каждой работы студент должен:

1) ознакомиться с соответствующим лекционным и теоретическим материалом, изучить учебные примеры;

2) выполнить общее и индивидуальное задания;

3) продемонстрировать преподавателю работу на составленных примерах;

4) составить отчет согласно указанным требованиям;

5) защитить работу по предлагаемому отчету.

Защита работы предполагает проведение собеседования по вопросам самопроверки, а также умение пояснять программный код и полученные результаты выполнения примеров.

Требования к отчету по итогам выполнения работ:

Структура отчета:

1. Титульный лист.
2. Цель работы.
3. Условия задач в соответствии со своим вариантом.
4. Блок-схема алгоритма решения задачи.
5. Программный код приложения.
6. Тесты для проверки правильности работы программы.
7. Результаты работы приложения (копии вывода окон с результатами).

Лабораторная работа №1

Знакомство с языком программирования Python. Настройка среды разработки Eclipse для Python.

Цель: Познакомиться со средой разработки для языка программирования Python. Освоить принципы работы с языком из-под командной строки Windows.

Краткие теоретические сведения

Установка Python в Windows XP и более поздних версиях Windows. Необходимо зайти на официальный сайт Python из любого браузера по адресу: http://www.python.org/. Выбираем раздел *Download* и активируем вкладку *Windows*.

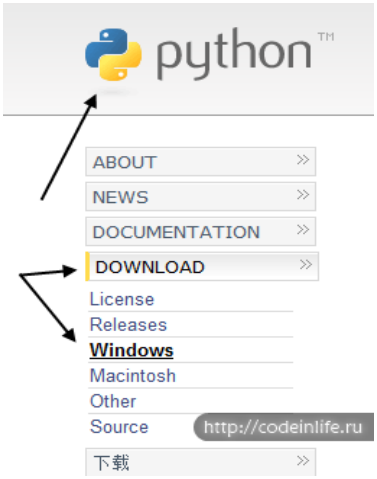


Рис.1.1 Сайт http://www.python.org/

Далее на открывшейся странице под заголовком Python для Windows необходимо найти ссылку на релизы.

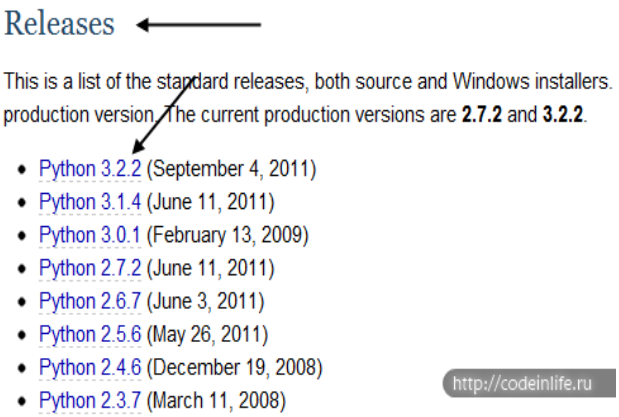


Рис. 1.2 Выбор необходимого релиза

Запускаем и следуем инструкциям.

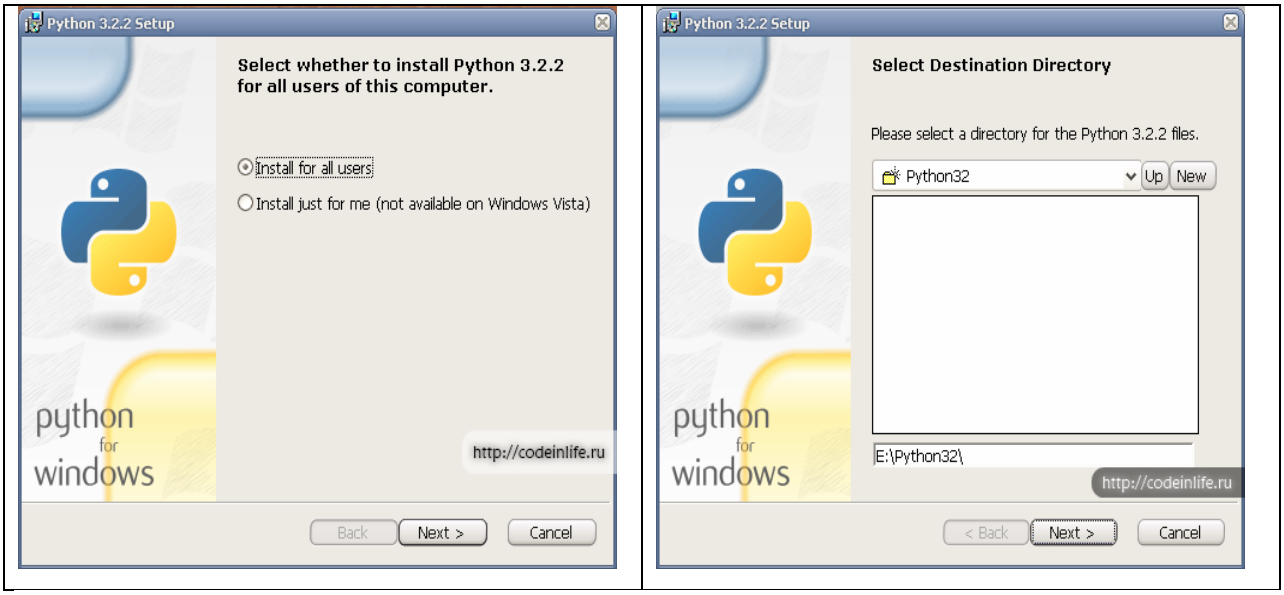


Рис. 1.3. Установка Python

Изначально после установки реализованы возможности работы в двух режимах — в командной строке (очень удобно для быстрой проверки небольших скриптов и отдельных функций) и через IDLE (Shell оболочка для Python).

Для корректной работ необходимо так же настроить переменные среды. Для этого активировать свойства системы меню «Мой компьютер» и выбрать команду «Переменные среды».

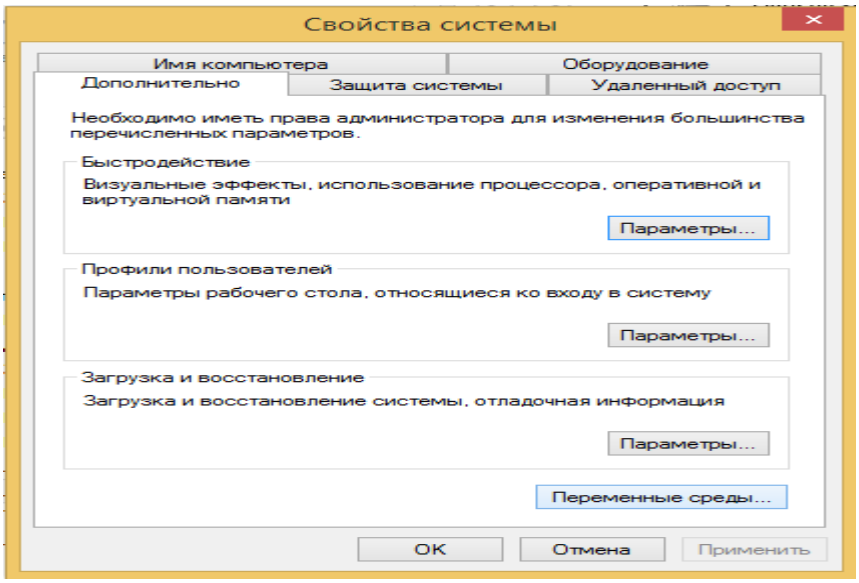
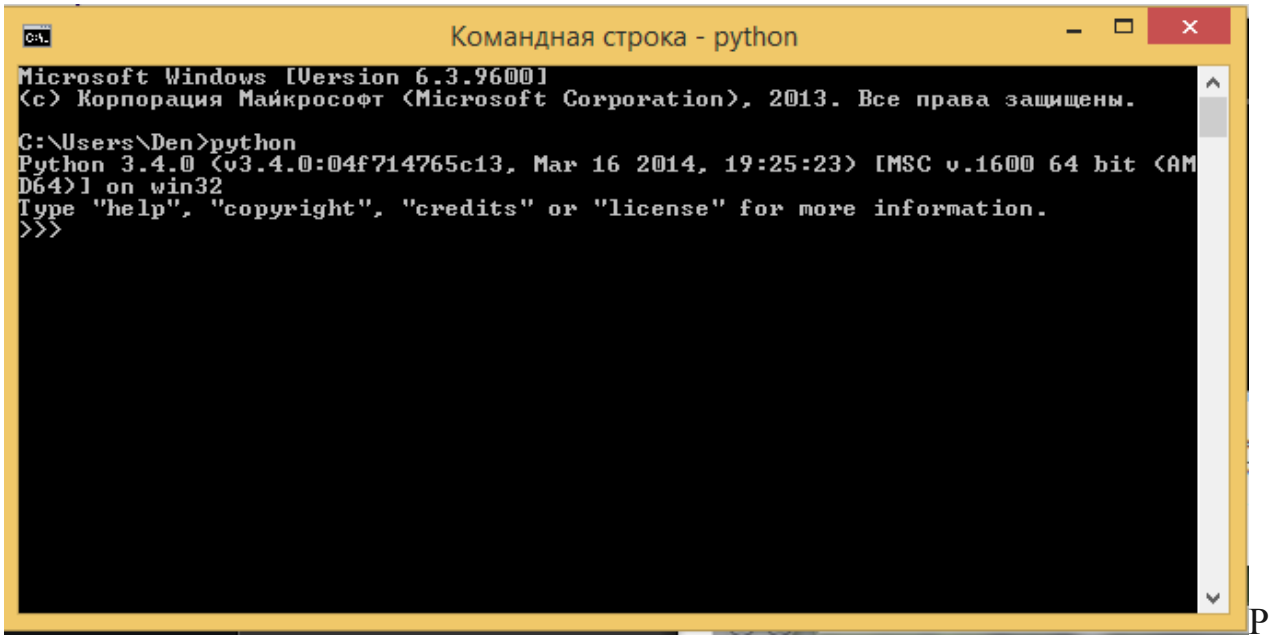


Рис. 1.4. Изменение параметра Path раздела переменные среды.

После этого необходимо выбрать команду Path, и активировав ее указать путь до установленного exe файла Python.

Проверка запуска Python из командной строки Windows. Для этого активируем пункт меню “Выполнить”, вызовем команду “CMD”, а в ней вызовем Python.



Р

Рис. 1.5. Пример работы в командной строке.

Выполните в вызванном языке несколько простейших команд:

>>>2+2

4

>>> #Это комментарий

...2+2

4

>>> 2+2 # комментарий с кодом в строке совместно

4

Установка Eclipse.

Eclipse (от англ. *затмение*) — свободная интегрированная среда разработки модульных кроссплатформенных приложений. Развивается и поддерживается Eclipse Foundation.

Наиболее известные приложения на основе Eclipse Platform — различные «Eclipse IDE» для разработки ПО на множестве языков (например, наиболее популярный «Java IDE», поддерживавшийся изначально, не полагается на какие-либо закрытые расширения, использует стандартный открытый API для доступа к Eclipse Platform).

Скачать необходимую версию для вашей платформы рекомендуется с официального сайта http://www.eclipse.org/downloads/.

По завершении скачивания, распакуйте папку Eclipse в любой раздел на компьютере. Для корректной работы Eclipse необходим JDK - Java Development

Kit, без него не запуск невозможен. JDK — это Java Development Kit, скачать его возможно с сайта Oracle по адресу: http://www.oracle.com/. При запуске Eclipse спросит куда ему сохранять все проекты, укажите ему ваш Workspace (рабочую директорию проектов).

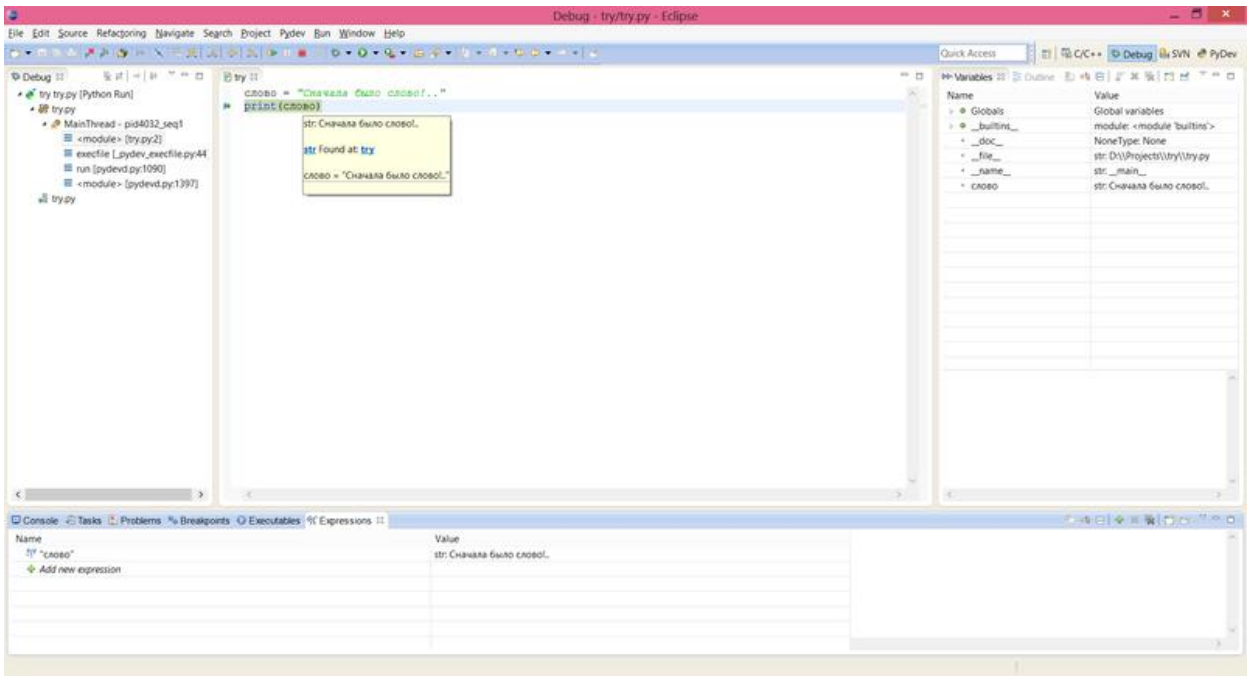


Рис. 1.6 Установка Eclipse

Необходимо убедиться, что ваше рабочее место по умолчанию настроено на кодировку UTF-8. В меню **Window => Preferences** выбираем **General => Workspace** в самом низу страницы необходимо указать параметр "**file encoding**", если он не выставлен в UTF-8 по умолчанию, прописать это строковое значение в поле **Other**, как показано на рис.1.7:

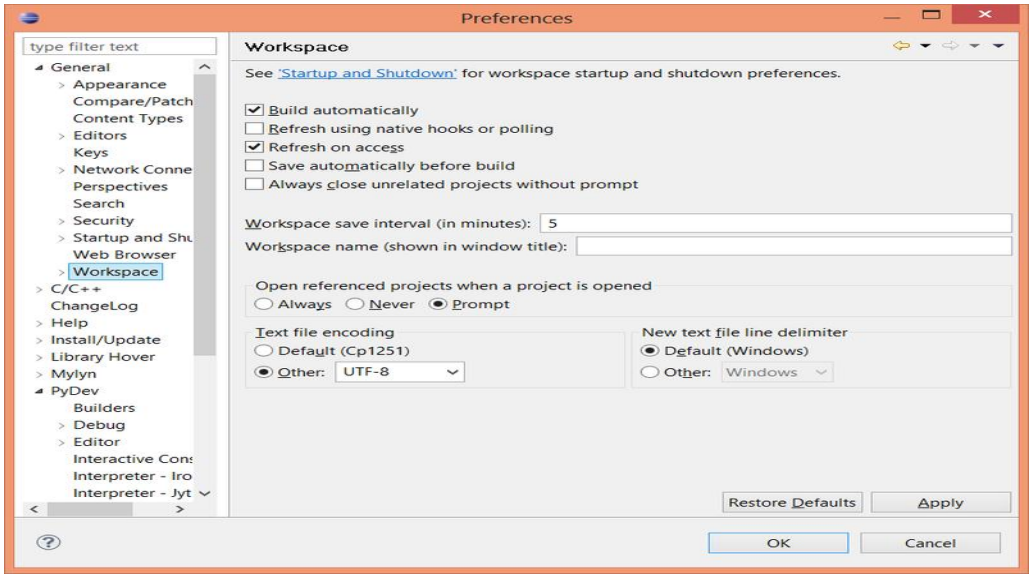


Рис. 1.7 Настройка кодировки UTF-8

Для корректной отладки программ написанных в языке Python нам необходимо использовать плагин PyDev. Активируем команду меню "**Help => Eclipse Marketplace**" набираем в строке поиска **PyDev** и нажимаем в найденном плагине кнопку [**Install**]

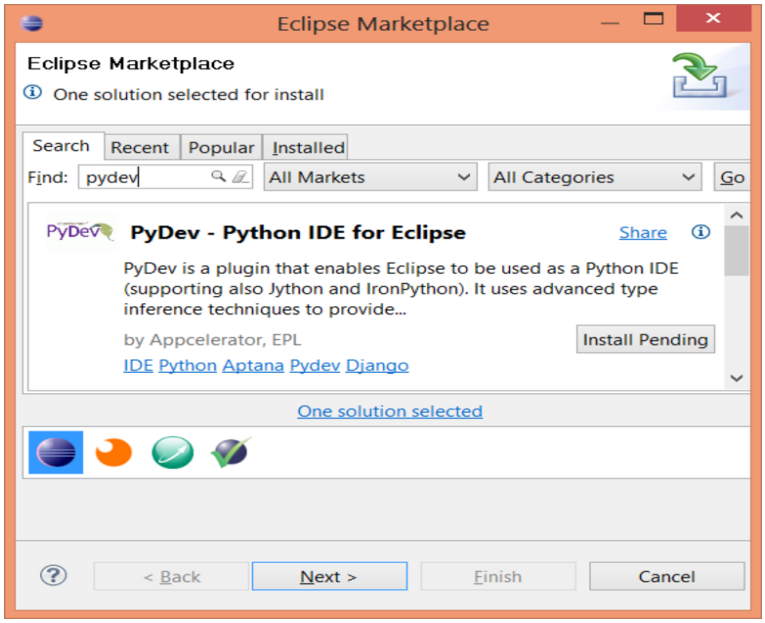


Рис. 1.8 Установка плагина PyDev под Eclipse

После чего в открывшемся окне Open Perspective выбираем окружение для разработки Python — свежеустановленный PyDev. Вы должны сразу после выбора переключиться на перспективу PyDev, в меню и в основном окне должны произойти некоторые изменения, которые Вы, возможно, не сразу заметите, но они переключают создание новых проектов и файлов и процесс отладки в режим разработки на Python.

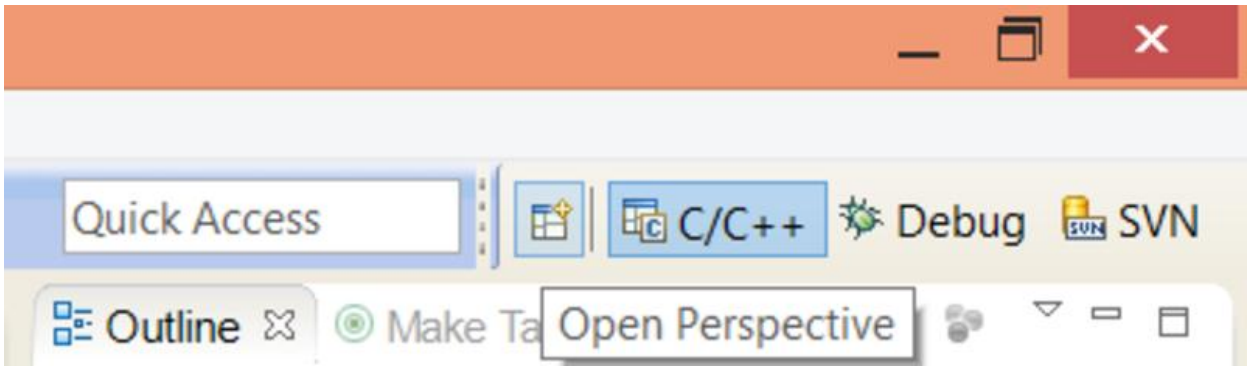


Рис. 1.9 Настройка плагина PyDev

Первый проект

Активируем команду меню File => New => PyDev Project, заполняем имя, выбираем версию Python 3.x и теперь самое главное: необходимо указать

расположение интерпретатор Python, для этого необходимо активировать команду «Please configure an interpreter...»

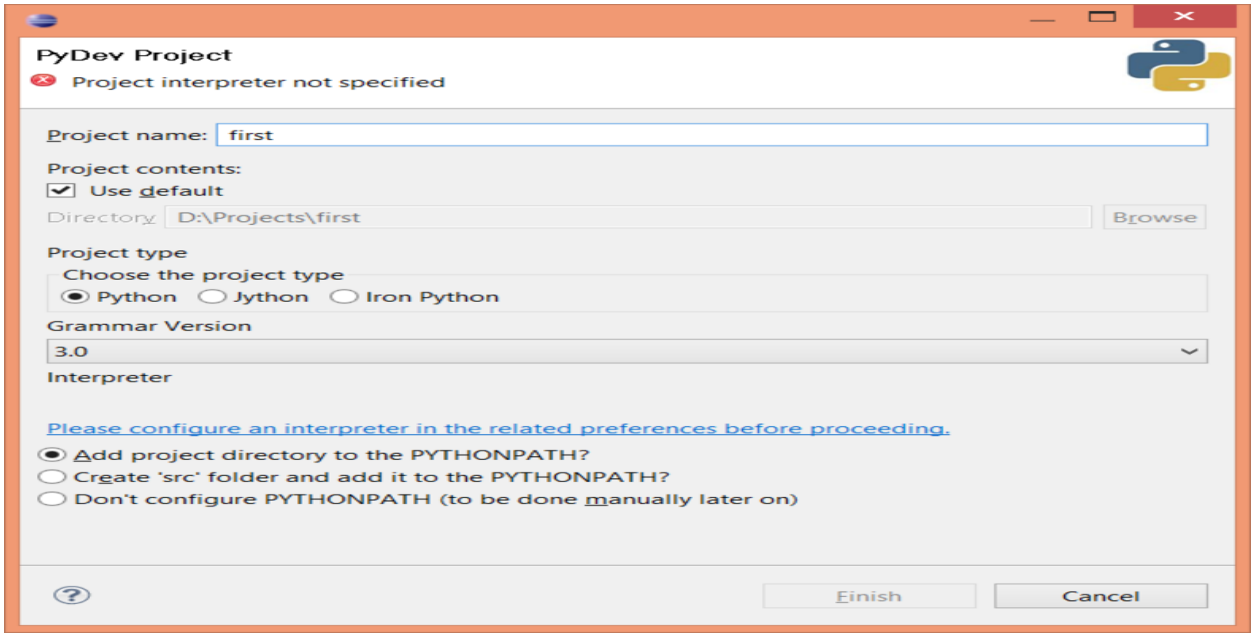


Рис.1.10 Настройка интерпретатора для проекта

Для написания первого скрипта активируем команду меню «File => New => PyDev module», либо просто правой кнопкой активируем проект «New => PyDev module», задаём нашему новому скрипту имя и нажимаем Finish. Файл скрипта автоматически создастся в кодировке, указанной в настройках Workspace, которые мы задали в самом начале. В Python 3.x весь код по стандарту в кодировке UTF-8. Для проверки поддержки юникода пишем такой скрипт:

```
#coding: utf-8;
слово = "Сначала было слово!.."
print(слово)
```

Для отладки проекта и его запуска активируем значок на зелёного жука (DeBug) или просто нажимаем F11 на клавиатуре.

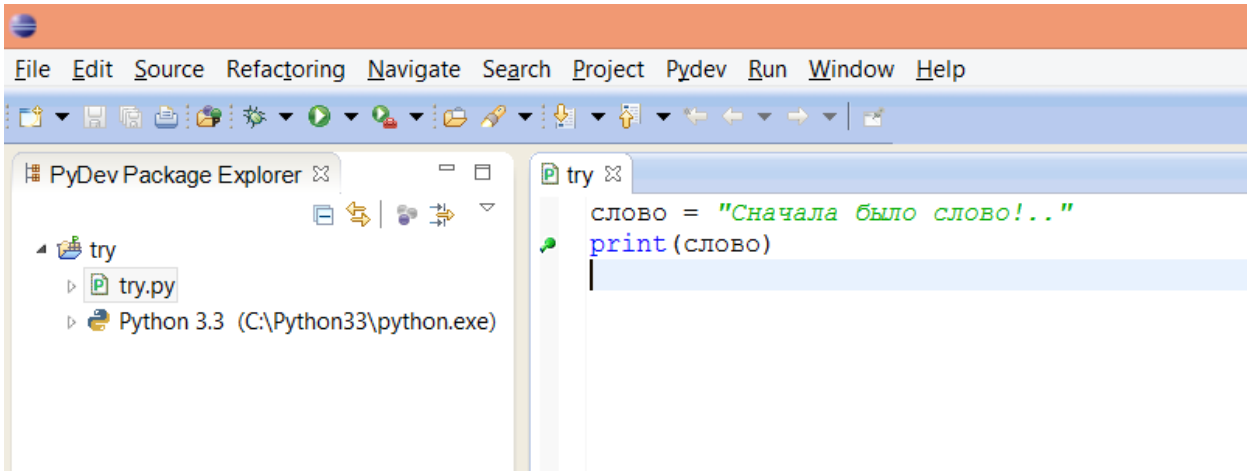


Рис.1. 12. Пример первой программы

Контрольные вопросы

1. Что такое среда разработки Eclipse
2. Каким образом можно писать программы на языке Pythonв режиме командной строки
3. Что такое IDLE Python.

Лабораторная работа №2

Синтаксис и структура программы, ввод-вывод данных, вычисления

Цель работы: изучить структуру программы, функции ввода-вывода, арифметические операции и функции, правили написания выражений и функции преобразования типов.

Краткие теоретические сведения

Структура программы

Программа на Python представляет собой последовательность команд для ввода данных, вычислений и других операций с данными, и вывода результатов. Простые команды (операторы) принято записывать по одной строке на оператор.

В составных операторах большую роль играют пробелы в начале строки (отступы). Программа создаётся в виде текстового файла в любом текстовом редакторе. Использование интегрированных сред разработки (IDE) обеспечивает подсветку синтаксиса и выделение особенностей структуры программы, а также упрощает поиск ошибок в написании команд. Файл с программой должен иметь «расширение» . py (например, my_program.py).

Первую строку программы необходимо оформить как комментарий, в котором указывается кодировочная таблица («кодировка») данных файла, в противном случае Python не сможет правильно интерпретировать любые символы, встретившиеся в тексте программы и выходящие за пределы основной латиницы (в том числе кириллицу). Указанная в этом комментарии кодировка должна соответствовать действительной кодировке файла.

Инструкции – это то, что сообщается интерпретатору, то, какие действия должна выполнять ваша программа. Если программа «выполняет какие-то действия», то инструкции – это способ указать, какие именно действия должна выполнять программа. Python – это процедурный язык программирования, основанный на использовании инструкций;

Комбинируя инструкции, Вы задаете процедуру, которую выполняет интерпретатор в соответствии с целями программы.

Другой способ понять роль инструкций состоит в том, чтобы рассмотреть иерархии понятий. Выделяют следующую ступень иерархии:

1. Программы делятся на модули.

2. Модули содержат инструкции.

3. Инструкции состоят из выражений.

4. Выражения создают и обрабатывают объекты.

Синтаксис языка

Синтаксис языка Python по сути построен на инструкциях и выражениях. Выражения обрабатывают объекты и встраиваются в инструкции. Инструкции представляют собой более крупные *логические* блоки программы – они напрямую используют выражения для обработки объектов, которые мы рассматривали в предыдущих главах. Кроме того, инструкции – это место, где создаются объекты (например, в инструкциях присваивания), а в некоторых инструкциях создаются совершенно новые виды объектов (функции, классы и так далее).

Инструкции всегда присутствуют в модулях, которые сами управляются инструкциями.

Каждая инструкция в программе завершается символом перевода строки (не ставится никаких символов типа ; как во многих других языках).

Длинные инструкции допускается располагать в нескольких строках, используя символ продолжения строки (\), как показано ниже:

```
a = math.cos(3 * (x - n)) + \
math.sin(3 * (y - n))
```

Символ продолжения строки не используется внутри строк в тройных кавычках, в определениях списков, кортежей или словарей. В общем случае любая часть программы, заключенная в круглые (...), квадратные [...], фигурные {...} скобки или в тройные кавычки, может занимать несколько строк без применения символа продолжения строки, потому что в данных случаях явно обозначены начало и конец блока определения.

Отступы используются для отделения различных блоков программного кода, таких как тело функции, условного оператора, цикла или определения класса. Величина отступа для первой инструкции в блоке может выбираться произвольно, но отступы всех остальных инструкций в блоке должны быть равны отступу в первой инструкции.

Типы данных Python

В Python имеется множество встроенных типов данных. Вот наиболее важные из них:

1. **Логический**, может принимать одно из двух значений — True (истина) или False (ложь).

2. **Числа**, могут быть целыми (1 и 2), с плавающей точкой (1.1 и 1.2), дробными (1/2 и 2/3), и комплексными.

3. **Строки** — последовательности символов Юникода, например, HTML-документ.

4. **Байты** и **массивы байтов**, например, файл изображения в формате JPEG.

5. **Списки** — упорядоченные последовательности значений.

6. **Кортежи** — упорядоченные неизменяемые последовательности значений.

7. **Множества** — неупорядоченные наборы значений.

8. **Словари** — неупорядоченные наборы пар вида ключ-значение.

Целые числа

В языке Python имеется два целочисленных типа, int и bool. И целые числа, и логические значения являются неизменяемыми объектами, но благодаря присутствию в языке Python комбинированных операторов присваивания эта особенность практически незаметна. В логических выражениях число 0 и значение False представляют False, а любое другое целое число и значение True представляют True. В числовых выражениях значение True представляет 1, а False - 0. Это означает, что можно записывать весьма странные выражения, например, выражение i+=True увеличит значение i на единицу. Естественно, более правильным будет записывать подобные выражения как i += 1.

Двоичные числа записываются с префиксом 0Ь, восьмеричные - в префиксом Оо и шестнадцатеричные - с префиксом Ох. В префиксах допускается использовать символы верхнего регистра.

Таблица 2.1. Математические операторы в Python

Символ	Значение
+	Сложение/тождество
-	Вычитание/отрицание
*	Умножение
/	Деление

%	Деление по модулю
* *	Возведение в степень
divmod (x, y)	Функция, возвращающая значения x / y и x % y
Abs(x)	Возвращает абсолютное значение x
Round (x,n)	Возвращает значение типа int, соответствующее значению x типа float, округленному до ближайшего целого числа (или значение типа float, округленное до /7-го знака после запятой, если задан аргумент n)
//	Делит x на y, при этом усекает дробную часть, поэтому результатом всегда является значение типа int, смотрите также функцию round()
Pow(x,y)	Возводит x в степень y; то же самое, что и оператор **
Pow(x,y,z)	Более быстрая альтернатива выражению (x ** y) % z

Таблица 2.2. Функции преобразования целых чисел

bin(i)	Возвращает двоичное представление целого числа i в виде строки, например, bin(1980) == '0b11110111100'
hex(i)	Возвращает шестнадцатеричное представление целого числа i в виде строки, например, hex(1980) == '0x7bc'
int(x)	Преобразует объект x в целое число; в случае ошибки во время преобразования возбуждает исключение ValueError, а если тип объекта x не поддерживает преобразование в целое число, возбуждает исключение TypeErrror. Если x является числом с плавающей точкой, оно преобразуется в целое число путем усечения дробной части.
int(s, base)	Преобразует строку s в целое число, в случае ошибки возбуждает исключение ValueError. Если задан необязательный аргумент base, он должен быть целым числом в диапазоне от 2 до 36 включительно.
oct(i)	Возвращает восьмеричное представление целого числа i в виде строки, например, oct(1980) == '0o3674"

Тип чисел с плавающей точкой

Язык Python предоставляет три типа значений с плавающей точкой: встроенные типы **float** и **complex** и тип **decimal**. **Decimal** в стандартной библиотеке. Все три типа данных относятся к категории **неизменяемых**. Тип float представляет числа с плавающей точкой двойной точности, диапазон

значений которых зависит от компилятора языка C (или C# или Java), применявшегося для компиляции интерпретатора Python. Числа этого типа имеют ограниченную точность и не могут надежно сравниваться на равенство значений. Числа типа float записываются с десятичной точкой или в экспоненциальной форме записи, например,

0.0, 4., 5.7, -2.5, -2e9, 8.9e-4.

В машинном представлении числа с плавающей точкой хранятся как двоичные числа. Это означает, что одни дробные значения могут быть представлены точно (такие как 0.5), а другие - только приблизительно (такие как 0.1 и 0.2). Кроме того, для представления используется фиксированное число битов, поэтому существует ограничение на количество цифр в представлении таких чисел.

Пример, полученный в IDLE:

```
>>> 0.0, 5.4, -2.5, 8.9e-4
(0.0, 5.4000000000000004, -2.5, 0.00088999999999999995)
```

Проблема потери точности - это не проблема, свойственная только языку Python; все языки программирования обнаруживают проблему с точным представлением чисел с плавающей точкой.

Если вам действительно необходимо обеспечить высокую точность, можно использовать числа типа **decimal. decimal**. Эти числа обеспечивают уровень точности, который Вы укажете (по умолчанию 28 знаков после запятой), и могут точно представлять периодические числа, такие как 0.11, но скорость работы с такими числами существенно ниже, чем с обычными числами типа float.

Вследствие высокой точности числа типа **decimal.Decimal** прекрасно подходят для производства финансовых вычислений. Смешанная арифметика поддерживается таким образом, что результатом выражения с участием чисел типов int и float является число типа float, а с участием типов float и complex результатом является число типа complex. Поскольку числа типа decimal. Decimal имеют фиксированную точность, они могут участвовать в выражениях только с другими числами decimal. Decimal и с числами типа int; результатом

таких выражений является число decimal.Decimal. В случае попытки выполнить операцию над несовместимыми типами возбуждается исключение TypeEггог.

Преобразование типов

Возможно использовать функцию type() для проверки типа любого значения или переменной, возможно также явно преобразовать значение типа int в тип float, вызвав функцию float(), возможно также преобразовать значение типа float в значение типа int, с помощью функции int(). Функция int() отбрасывает дробную часть числа, а не округляет его. Функция int() «округляет» отрицательные числа в сторону увеличения (подробнее преобразование типов см. в приложении 1).

Примеры

*>>>**a + 1, a - 1** # Сложение (3 + 1), вычитание (3 - 1)*

(4, 2)

*>>>**b * 3, b / 2** # Умножение (4 * 3), деление (4 / 2)*

(12, 2.0)

*>>>**a % 2, b ** 2** # Деление по модулю (остаток), возведение в степень*

(1, 16)

*>>>**2 + 4.0, 2.0 ** b** # Смешивание типов, выполняется преобразование*

(6.0, 16.0)

Знак # позволяет комментировать программу в Python

Деление с округлением вниз.

X // Y - этот оператор впервые появился в Python 2.2 и доступен в обеих версиях Python, 2.6 и 3.0. Он всегда отсекает дробную часть, округляя результат до ближайшего наименьшего целого, независимо от типов операндов.

Округление вниз – это совсем не то же самое, что усечение дробной части, – это обстоятельство приобретает значение при работе с отрицательными числами, например:

*>>>**5 / 2, 5 / -2***

(2.5, -2.5)

*>>>**5 // 2, 5 // -2** # Округление вниз: результат 2.5 округляется до 2,*

(2, -3) # а -2.5 округляется до -3

*>>>**5 / 2.0, 5 / -2.0***

(2.5, -2.5)

*>>>**5 // 2.0, 5 // -2.0** # То же относится и к вещественному делению,*

(2.0, -3.0) # только результат имеет вещественный тип

Извлечение квадратного корня

$$t_1 = \frac{v_0 - \sqrt{v_0^2 - 2gy_c}}{g}, t_2 = \frac{v_0 + \sqrt{v_0^2 - 2gy_c}}{g}$$

Стандартный набор математических операций не содержит в себе операцию извлечения квадратного корня. В Python функция, извлекающая квадратный корень и множество других функции доступны в модуле, который называется math. Чтобы их использовать мы должны импортировать этот модуль в программу, написав перед тем местом, где мы будем использовать функции, команду import math. После этого, чтобы взять квадратный корень от переменной *a* напишите math.sqrt(a).

```
# импортируем модуль для работы с математическими функциями
# для быстрых вычислений следует использовать модуль cmath
# в отличии от math - cmath является не скриптом, а исполняемым файлом
Import math
v0 =5
g=9.81
yc=0.2
# используем функцию извлечения
корня квадратного - math.sqrt
t1 =(v0 - math.sqrt(v0**2 - 2*g*yc))/g
t2 =(v0 + math.sqrt(v0**2 - 2*g*yc))/g
print(t1, ‘ and ’,t2, ‘ ’)
```

Результат:

0.0417064 and 0.977662.

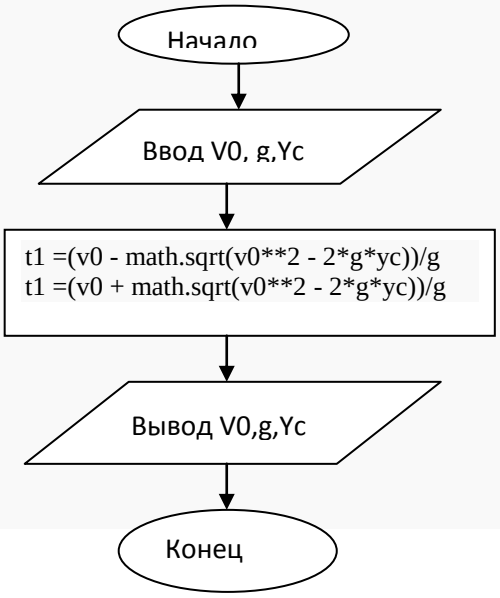


Рис. 2.1. блок схема программы нахождения квадратных корней

Команды import и from...import

Импортировать функции из каких-либо модулей можно двумя способами. Первый мы только что использовали:

```
Impor t  math
```

Чтобы вызвать функцию нужно записать имя_модуля.имя_функции(), например для присваивания переменной 'x' квадратного корня из переменной 'y' записывают:

```
x =math.sqrt(y)
```

Альтернативный способ, позволяющий использовать функцию без префикса имени модуля. Этот альтернативный способ имеет вид "from module import function". Например:

```
From  math  import   sqrt
```

Если в вычислениях вам требуется несколько функций, Вы можете перечислить их через запятую:

```
From   math   import   sqrt, exp, log,  sin
```

Иногда даже пишут:

```
From  math   import *
```

и тогда из модуля импортируются все функции и константы. Это удобный способ, если вам требуется импортировать много функций (что встречается, кстати, не так часто). Но Вы можете столкнуться с ошибками, поскольку Вы импортируете множество имен, за которыми Вы не следите, и имена Ваших функций и переменных могут совпасть, а это приведет к некорректным результатам. Поэтому рекомендуется импортировать только те функции и константы, которые Вам действительно нужны. Обычно инструкция import находится в начальной части программы, и, по мере написания программы, Вы добавляете к этой инструкции новые модули и функции, которые Вам требуются.

Еще одним удобным вариантом является импорт с новым именем, когда одновременно с добавлением модуля или функции мы их именуем произвольным образом:

```
Import  math  as  m
# m теперь имя math в программе
v = m.sin(m.pi)
from  math  import  log  as  ln
v = ln(5)
```

```
from math import sin as s, cos as c, log as ln
v = s(x)*c(x) + ln(x)
```

Как мы помним, в Python всё объекты и переменные могут ссылаться на объекты модулей и функций также как на объекты строк и чисел, например:

```
m =math
ln = m.log
s = m.sin
c = m.cos
```

Пример: гиперболический синус:

$$\sinh(x) = \frac{1}{2}(e^x - e^{-x})$$

Для того чтобы вызвать гиперболический синус, мы можем использовать три способа:

1) самый простой - вызвать встроенную функцию math.sinh( ),

2) использовать два раза функцию math.exp( ),

3) использовать возведение в степень числа e.

```
From math import sinh, exp, e, pi
x =2*pi
r1 = sinh(x)
r2 =0.5*(exp(x) - exp(-x))
r3 =0.5*(e**x - e**(-x))
print (r1,r2,r3)
```

На выходе имеем идентичные результаты:

267.744894041 267.744894041 267.744894041

Расчёт для r1 по времени самый короткий, и именно его стоит использовать в программах.

Команды dir( ) и help( )

Для того, чтобы посмотреть все, что Вы можете извлечь из какого-либо модуля, достаточно после того как Вы его импортировали в интерактивном режиме, написать dir(имя_модуля). Например, это может выглядеть так:

```
>>>import math
>>>dir(math)
['__doc__','__name__','__package__','acos','acosh','asin','asinh','atan','atan2','
atanh','ceil','copysign','cos','cosh','degrees','e','exp','fabs','factorial','floor','fmod',
'frexp','fsum','hypot','isinf','isnan','ldexp','log','log10','log1p','modf','pi','pow',
'radians','sin','sinh','sqrt','tan','tanh','trunc']
```

Имена с двумя подчеркиваниями относятся к системным, и о них мы поговорим несколькими уроками позже. Все остальные — это функции и константы, все, что мы непосредственно используем. Для того, чтобы узнать о назначении каждой функции, достаточно написать help (имя_модуля.имя_функции). Аналогично можно просмотреть и всю информацию о модуле, например help(math).

Ниже приведен список основных функций модуля math.

Некоторые из перечисленных функций (int, round, abs) являются стандартными и не требуют подключения модуля math для использования.

Таблица 2.3. Функции модуля Math

Функция	**Описание**
Округление	
int(x)	Округляет число в сторону нуля. Это стандартная функция, для ее использования не нужно подключать модуль math.
round(x)	Округляет число до ближайшего целого. Если дробная часть числа равна 0.5, то число округляется до ближайшего четного числа.
round(x, n)	Округляет число x до n знаков после точки. Это стандартная функция, для ее использования не нужно подключать модуль math.
floor(x)	Округляет число вниз (“пол”), при этом floor(1.5) == 1, floor(-1.5) == -2
ceil(x)	Округляет число вверх (“потолок”), при этом ceil(1.5) == 2, ceil(-1.5) == -1
trunc(x)	Округление в сторону нуля (так же, как функция int).
abs(x)	Модуль (абсолютная величина). Это - стандартная функция.
fabs(x)	Модуль (абсолютная величина). Эта функция всегда возвращает значение типа float.
Корни, степени, логарифмы	
sqrt(x)	Квадратный корень. Использование: sqrt(x)
pow(a, b)	Возведение в степень, возвращает a^b. Использование: pow(a,b)
exp(x)	Экспонента, возвращает e^x. Использование: exp(x)
log(x)	Натуральный логарифм. При вызове в виде log(x, b) возвращает логарифм по

	основанию b.
log10(x)	Десятичный логарифм
e	Основание натуральных логарифмов $e \approx 2{,}71828\ldots$
Тригонометрия	
sin(x)	Синус угла, задаваемого в радианах
cos(x)	Косинус угла, задаваемого в радианах
tan(x)	Тангенс угла, задаваемого в радианах
asin(x)	Арксинус, возвращает значение в радианах
acos(x)	Арккосинус, возвращает значение в радианах
atan(x)	Арктангенс, возвращает значение в радианах
atan2(y, x)	Полярный угол (в радианах) точки с координатами (x, y).
hypot(a, b)	Длина гипотенузы прямоугольного треугольника с катетами a и b.
degrees(x)	Преобразует угол, заданный в радианах, в градусы.
radians(x)	Преобразует угол, заданный в градусах, в радианы.
pi	Константа π

Ввод данных с клавиатуры

Ввод данных с клавиатуры в программу (начиная с версии Python 3.0) осуществляется с помощью функции **input()**. Когда данная функция выполняется, то поток выполнения программы останавливается в ожидании данных, которые пользователь должен ввести с помощью клавиатуры. После ввода данных и нажатия Enter, функция **input()** завершает свое выполнение и возвращает результат, который представляет собой строку символов, введенных пользователем.

```
>>> input()
1234
'1234'
>>> input()
Hello World!
'Hello World!'
>>>
```

Когда выполняющаяся программа предлагает пользователю что-либо ввести, то пользователь может не понять, что от него требуется. В этом случае необходимо вывести пояснение. С этой целью функция **input()** может

принимать необязательный аргумент-приглашение строкового типа; при выполнении функции сообщение будет появляться на экране и информировать человека о запрашиваемых данных.

```
>>> input("Введите номер карты: ")
Введите номер карты: 98765
'98765'
>>> input('Input your name: ')
Input  your name:   Sasha
'Sasha'
>>>
```

Из примеров видно, что данные возвращаются в виде переменной строкового типа данных, даже если было введено числовое значение. В более ранних версиях Python использовались две встроенные функции, позволяющие получать данные с клавиатуры: **raw_input()**, возвращающая в программу строку и **input()**, возвращающая число. Начиная с версии Python 3.0, если требуется получить число, то результат выполнения функции **input()** изменяют с помощью функций **int()** или **float()**.

Строки

Строки используются для записи текстовой информации, а также произвольных последовательностей байтов. Это наш первый пример последовательностей, или упорядоченных коллекций других объектов, в языке Python. Последовательности поддерживают порядок размещения элементов, которые они содержат, слева направо: элементы сохраняются и извлекаются исходя из их позиций в последовательностях. Строго говоря, строки являются последовательностями односимвольных строк. Другими типами последовательностей являются списки и кортежи (будут описаны ниже).

Вывод в Python

Вывод информации осуществляется с помощью функции print() (в версии до 3.0 - оператор print.

Пример. Вывод объекта.

```
Python 2.6
>>>x = 1
>>> print x
1
```

Python 3.1.1

>>> x = 1

>>> print(x)

1

Для вывода сообщений необходимо использовать кавычки

x = 1

>>> print("x=",x)

x=1

Могут быть использованы двойные и одинарные кавычки, за исключением случаев форматирования больших разделов текста.

Варианты заданий

Задача 1

Вычислить значения a,b,c в зависимости от значений x,y для двух случаев:

a) объявив a,b,c вещественными

b) объявив a,b,c целыми

$$a = cos(x^2 + \frac{1}{y}) + |x + y - 5|$$

1) $b = e^{(x-1)} + y\sqrt{x+10}$

$$c = tg^3(x^2 + y^3) - ln(x + y)^2$$

$$a = (|x^2 + y - 10| + \sqrt{x^2 - 25 + y})x$$

2) $b = tg(x^2 + 10)^2 + sin(x + y)$

$$c = \frac{x^2 - 25}{y^2 + 10y} + cos^2(x + y)$$

$$a = \frac{y^3 - 9}{(x+1)y} + cos^2(x + y)$$

3) $b = tg(|x| + 10) + sinx$

$$c = ln(x - 16) + |x + y|5$$

$$a = \frac{\sqrt{x-1} - \sqrt{y}}{1 + \frac{x^2}{2} + \frac{y^2}{4}}$$

4) $b = x(tgz + e^{-(x+3)})$

$$c = \frac{3 + e^{y-1}}{1 + x^2 - 10}$$

5)
$$a = (1+y)\frac{x+y^2}{e^{-x-2}} + \frac{1}{y}$$
$$b = 1 + |y-x| + \frac{(y-x)^2}{2} + \frac{|y-x|^3}{x(y-10)}$$
$$c = (x+10+y)^2 \frac{\sin x + \sin^2(x+5)}{|x-y-5|}$$

6)
$$a = (\frac{\sqrt{x}}{y+x} + cos^2(x+25))$$
$$b = (x^2+y^2)\,(ln(|x|+y) + y^2 + 10)$$
$$c = (cos(x + \frac{x}{y} + 25) + sin^2 x$$

---

7)
$$a = ln(x^2 + |y+10|^2)\, sin x$$
$$b = (|25+x|+y)^2 \frac{x^2}{y+x}$$
$$c = (cos^2 x + sin(x+25+y)^2)\frac{x}{25-y}$$

8)
$$a = \frac{x^2+y^2}{10} * \frac{(x+y)^2}{(x-y)10}$$
$$b = \frac{(x+25y)\sin x^2}{10} + \sqrt{(10-x^2)(y+5)}$$
$$c = \sin(x+10)^2 + e^{x/(x-5)}$$

---

9)
$$a = \ln(x^2 + |y|^2)\sqrt{\sin x}$$
$$b = (x-y)\frac{y^5}{|y+x|}$$
$$c = (tg\ x + \cos(x-y)^2)\frac{x}{|5-y|}$$

10)
$$a = \sqrt{x} + sin^2(x+5) + y^3$$
$$b = (\frac{x^2+5}{xy} + |x|)\,(x+y)^2$$
$$c = sin x^2 + 5(x^3 + ln y)$$

---

11)
$$a = \sqrt{x-y} + \sin\ x^2$$
$$b = (\frac{x^2}{x-y} + |x|) - tg\ \ y$$
$$c = \sin x^2 \,/\, x^3$$

12)
$$a = \frac{1}{xy} tg \frac{xy}{1+xy}$$
$$b = (x^2+y^2) - e^{-(x+y)}$$
$$c = \frac{\ln(x+e^y)}{\sqrt{x^2+y^2}}$$

13)
$$a = e^{x^2-y^2}\sin 2xy$$
$$b = \frac{\ln(1-x^2-y^2)}{\sin x \sin y}$$
$$c = tg\frac{x^2}{y} \cdot \frac{1}{\sqrt{x^2}+y^2}$$

14)
$$a = \frac{\cos x}{\sqrt{x^3+e^x}}$$
$$b = \sin e^x \cdot \ln(\sqrt{x^2+y^2} - 3)$$
$$c = 25\ln^2 x - 20\ln(x-3) - \frac{1}{x}$$

Задача 2

1. Идет k-я секунда суток. Определить, сколько полных часов и полных минут прошло к этому моменту (например, если k равно 13 257, то количество часов равно 3, минут – 40, секунд – 57).
2. Присвоить целой переменной Y первую цифру из дробной части положительного вещественного числа X (пример: если X=32.597 то Y=5).
3. Вычислить S – сумму цифр трехзначного целого числа K (пример: если K=259, то S=16).
4. Используя одну дополнительную переменную, поменять местами значения переменных X, Y и Z так, чтобы в X оказалось значение переменной Y, в Y – значение Z, а в Z – прежнее значение X.
5. Определить угол (в градусах) между положением часовой стрелки в начале суток (12 часов 00 минут 00 секунд) и ее положением в h часов, m минут и s секунд ($0 \le h \le 11$, $0 \le m \le 59$, $0 \le s \le 59$).
6. Написать программу, которая вычисляет периметр и площадь треугольника, если известны координаты его углов.
7. Написать программу, которая преобразует введенное с клавиатуры дробное число в денежный формат. Например, число 12,5 должно быть преобразовано к виду 12 руб. 50 коп.
8. Написать программу вычисления величины дохода по вкладу. Процентная ставка (в процентах годовых), время хранения (в днях) и величина вклада задаются во время работы программы.
9. Вычислить дробную часть среднего геометрического трех заданных положительных чисел.
10. Получить число, полученное выписыванием в обратном порядке цифр заданного трехзначного числа.
11. Поменять местами целую и дробную части числа. Например, если переменная x=13,75, то должна стать 75,13. Примечание: у числа до и после запятой должно быть две цифры.
12. По длинам двух сторон некоторого треугольника и углу (в градусах) между ними найти длину третьей стороны и площадь этого треугольника.
13. Найти произведение цифр заданного четырехзначного числа.
14. Даны два действительных числа. Найти среднее арифметическое этих чисел и среднее геометрическое их модулей.

Контрольные вопросы

1. Опишите структуру программы в Python
2. Опишите синтаксис языка Python.
3. Каким образом в Python оформляются комментарии?
4. Что понимается под типом данных в языках программирования?
5. Правила описания переменных в Python.

6. Перечислите целые типы данных в Python.
7. Оператор ввода в Python.
8. Оператор (функция) вывода в Python.
9. Какие арифметические операторы использует Python?
10. Какие операции использует модуль mathв Python?
11. Назовите основные операции преобразования типов данных в Python.
12. Каким образом можно округлить вещественное число (число с плавающей точкой) в Python?
13. Каким образом можно изменять точность математических операций в Python?
14. Каким образом можно вызвать математические операции из модуля math в Python?
15. Операторы dir() и help() в Python.

Лабораторная работа №3

Алгоритмические конструкции ветвления

Цель работы: изучить логические функции и работу операторов ветвления, научиться правильно их использовать при написании программ.

Краткие теоретические сведения

Достаточно часто для реализации поставленной задачи использования линейных программ может быть недостаточным, в этом случае используется условный оператор.

Простая форма записи

If условие:

Оператор

Если условие выполняется, то есть принимает значение true, то выполняется команда (оператор), следующая после двоеточия.

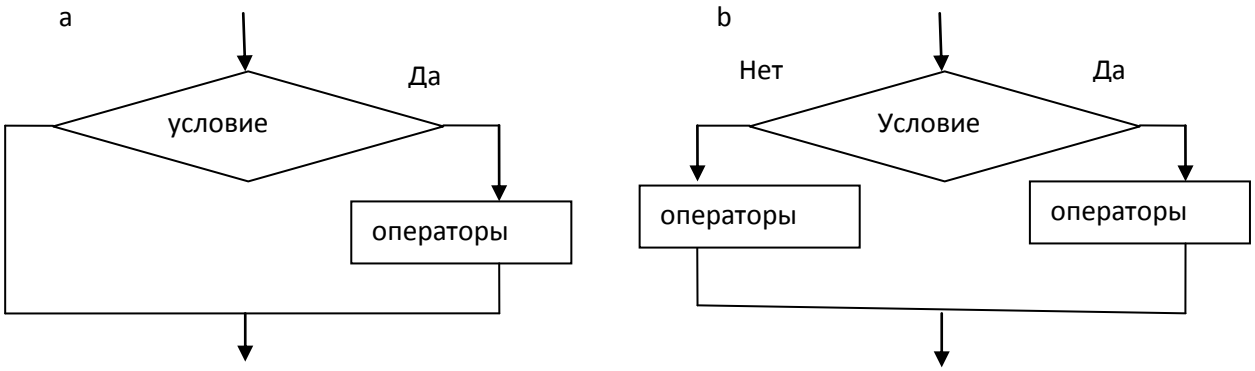


Рис. 3.1. Изображение условного оператора на блок-схемах

а) в сокращенной форме; б) в полной форме.

В том случае если необходимо чтобы был выполнен не один оператор по ветке true,то его необходимо писать на следующей строке с таким же отступом, как первый оператор:

If условие:

Оператор 1

Оператор 2

Оператор n

Если условие принимает значение false, то управление передается оператору, следующему за конструкцией if.

Расширенная форма условного оператора выглядит следующим образом:

```
If условие:
    Оператор 1
    Оператор 2
    Оператор n
else:
    Оператор 1
    Оператор 2
    Оператор n
```

Если условие выполняется, то есть принимает значение true,то выполняется оператор (группа операторов), следующая после условия и двоеточия, если же значение false то выполняется оператор (группа операторов), стоящая после else.

Оператор ветвления имеет в данном случае две части, операторы каждой из которых записываются с отступом вправо относительно оператора ветвления.

Еще более общий случай **оператора выбора** можно записать с помощью следующего синтаксиса (пример вычисления знака числа):

```
if a < 0:
   s = -1
elif a == 0:
   s = 0
else:
   s = 1
```

Стоит заметить, что elif - это сокращенный else if. Без сокращения пришлось бы применять **вложенный** оператор ветвления:

```
if a < 0:
  s = -1
else:
  if a == 0:
     s = 0
```

```
else:
    s = 1
```

В отличие от оператора print, оператор if-else является *составным оператором*.

Сравнение

Таблица 3.1. Логические операции в Python

x is y, x is not y	Проверка идентичности объектов
x == y, x != y	Операторы проверки на равенство (рано, не равно)
x < y, x <= y, x > y, x >= y	Операторы сравнения
x in y, x not in y	Проверка на вхождение (для итерируемых объектов и множеств)
not x	Логическое отрицание
x and y	Логический оператор И (значение y вычисляется, только если значение x истинно)
x or y	Логическая операция ИЛИ (значение y вычисляется, только если значение x ложно)
x if y else z	Трехместный оператор выбора (значение x вычисляется, только если значение y истинно)
x | y	Битовая операция ИЛИ, объединение множеств
x ^ y	Битовая операция «исключающее ИЛИ» (XOR),
x & y	Битовая операция И, пересечение множеств
x << y, x >> y	Сдвиг значения x влево или вправо на y битов
~x	Битовая операция НЕ (инверсия)

Обычные операторы сравнения действуют именно так, как и можно было бы ожидать, – они сравнивают значения операндов и возвращают логический результат (который, как правило, проверяется объемлющей инструкцией):

```
>>>1 < 2 # Меньше чем
True
>>>2.0 >= 1 # Больше или равно: число 1 преобразуется 1.0
True
>>>2.0 == 2.0 # Проверка на равенство значений
True
>>>2.0 != 2.0 # Проверка на неравенство значений
```

False

Обратите внимание, смешивание разнотипных операндов допускается, только если оба они принадлежат к числовым типам. Во второй инструкции, в примере выше, интерпретатор выполняет сравнивание с позиции более сложного типа, преобразуя целое число в вещественное.

*>>>**X < Y < Z** # Составная операция сравнения: принадлежность диапазону*

True

*>>>**X < Y and Y < Z***

True

То же самое относится и к выражениям с ложным результатом; кроме того, допускается составлять цепочки произвольной длины:

*>>>**X < Y > Z***

False

*>>>**X < Y and Y > Z***

False

*>>>**1 < 2 < 3.0 < 4***

True

*>>>**1 > 2 > 3.0 > 4***

False

Пример

Задача 1. Написать программу, выполняющую следующие действия: если x меньше нуля обнулите его, меньше 10 - не меняйте, меньше 100 умножьте на два в противоположном случае разделите на 3.

```
#coding: utf-8;
x = int(input( 'Введите число'))
if x <0:
   x = 0
print ("Обнулим")
elif x <10:
print ("не меняем")
```

```
elif x <100:
    x=x*2
print ("умножим на два")
else:
    x=x/3
print('разделим на 3')
print ('вот так работает условие, результат = ', x )...
```

Варианты заданий

Задача 1

1) $y=\begin{cases}\pi x^2-5/x^2, & \text{если } X<3\\ ax^3+b\sqrt{x}, & \text{если } X=3\\ \ln(x+\sqrt{x+a}), & \text{если } X>3\end{cases}$

2) $y=\begin{cases}1/x^3, & \text{если } X<\pi/2;\\ bx-\sin^2 x, & \text{если } \pi/2\le X\le\pi\\ 1/(ae^x), & \text{если } X>\pi\end{cases}$

3) $y=\begin{cases}\pi x^2-7/(x+1)^2, & \text{если } X<1{,}4\\ a(x-2)^3/7\sqrt{bx}, & \text{если } X=1{,}4\\ x+7\sqrt{|x+a|}, & \text{если } X>1{,}4\end{cases}$

4) $y=\begin{cases}5b\cos^2 x, & \text{если } X<1\\ 1{,}8/(ax), & \text{если } X=1\\ (x-1)^2+6, & \text{если } 1<X<2\\ 3tgx, & \text{если } X>2\end{cases}$

5) $y=\begin{cases}x\sqrt{x-a}, & \text{если } X>a\\ x/\sin ax, & \text{если } X=a\\ e^{-ax}\cos ax, & \text{если } X<a\end{cases}$

6) $y=\begin{cases}(b+x^2)/\sqrt{x+a}, & \text{если } X<0{,}5\\ (b+1)\sqrt{x+a}, & \text{если } X=0.5\\ \cos x+b\sin^2 x, & \text{если } X>0{,}5\end{cases}$

7) $y=\begin{cases}\pi x^2-7/\sqrt{x}, & \text{если } X<1{,}3\\ ax^3+7\sqrt{x}, & \text{если } X=1{,}3\\ \ln(x-7\sqrt{x}), & \text{если } X>1{,}3\end{cases}$

8) $y=\begin{cases}(a+b)/(e^x+\cos x), & \text{если } X<2\\ a/(bx+1), & \text{если } 2\le X<8\\ e^x+\sin 2x, & \text{если } X\ge 8\end{cases}$

9) $y=\begin{cases} a\lg x+\sqrt[3]{x}, \textit{если } X>1 \\ 2a\cos x+3x^2, \textit{ если } X=1 \\ (a+5)/\ln(x-1)^2, \textit{если } X<1 \end{cases}$

10) $y=\begin{cases} 1/\lg(x+1), \textit{если } X<2 \\ 2\sin^2\sqrt{|ax|}, \textit{ если } X>2 \\ ax^3, \textit{если } X=2 \end{cases}$

11) $y=\begin{cases} a/(e^{0.1x}+\cos^2 x), \textit{если } X<3 \\ (a+b)/\ln(x+1), \textit{если } 3\le X<10 \\ |\sin x|, \quad \textit{если } X\ge 10 \end{cases}$

12) $y=\begin{cases} b/(0.4*5^x+\sin^2 x), \textit{если } a<b \\ (a+b)/\ln(a-b)+x, \quad \textit{если } a>b \\ |\sin^3 x|, \qquad \textit{если } a=b \end{cases}$

13) $y=\begin{cases} \dfrac{a+b}{c}-\sin^3(2x+c), \textit{ если } c>0 \\ \ln(x+2c)-\sqrt[3]{a-bc}, \textit{ если } c\le 0 \textit{ и } a>b \\ ctg\dfrac{ac}{x}+2xb, \textit{ иначе} \end{cases}$

14) $y=\begin{cases} ac-x\sin\dfrac{x^3}{ca^2}, \textit{ если } a,c>0 \\ \log_5(acx)^3+\dfrac{x^2+c}{ab}, \textit{ если } a,b\ne 0 \\ \sqrt[3]{ax-c\sin(bx-c)}, \textit{ иначе} \end{cases}$

15) $y=\begin{cases} \dfrac{ax^3}{|c^2+bx|}-\cos(ax+b), \textit{ если } a,b,c>0 \\ \ln(3x+c)+\sqrt[3]{abx}, \textit{ если } c,b<0 \\ (ax+cb)^3+\cos^2(abx), \textit{ иначе} \end{cases}$

16) $y=\begin{cases} ax^3-tg\dfrac{cx^3}{b}, \textit{ если } b>0 \\ \sin(ax^3)+\dfrac{x^{\ln 5}+c}{a^2b}, \textit{ если } a,b\ne 0 \\ \sqrt[3]{ax-bx-c}+\dfrac{5x}{12}, \textit{ иначе} \end{cases}$

Задача 2

1. Если сумма трех попарно различных натуральных чисел меньше 10, то минимальное из них заменить полусуммой двух других, иначе меньшее из двух первых - полусуммой оставшихся.

2. Даны натуральные числа a, b, c, d. Если $a\le b\le c\le d$, то каждое из них заменить наибольшим. Если a>b>c>d, то числа оставить без изменения, иначе заменить их квадратами.

3. Даны действительные числа x, y. Если x,y <0, то $x=|x|, y=|y|$, если меньше нуля только одно число, то x = 0,5x, y=0,5y; если x и y≥0 и ни одно не принадлежит отрезку [0;2], то x = x/10, y = y/10; иначе -оставить без изменения.

4. Даны действительные положительные числа a,b,c,x,y. Выяснить, пройдет ли кирпич с ребрами a,b,c в прямоугольное отверстие x, y. Кирпич просовывается в отверстие так, что каждое его ребро параллельно или перпендикулярно каждой стороне отверстия.

5. Дано натуральное число $N \le 100$, определяющее возраст человека в годах. Дать для этого числа наименование “год”, “года”, “лет”: 1 год,23 года,45 лет и т.д.

6. Даны действительные числа x, y ($x \neq y$). Меньшее из этих двух чисел заменить их полусуммой, а большее- их удвоенным произведением.

7. Если сумма трех попарно различных действительных чисел x, y, z меньше единицы, то наименьшее из этих трех чисел заменить полусуммой двух других; в противном случае заменить меньшее из x и y полусуммой двух оставшихся значений.

8. Даны действительные числа a, b, c, d. Если a?b?c?d , то каждое число заменить наибольшим из них; если a>b>c>d, то числа оставить без изменения; в противном случае все числа заменяются их квадратами.

9. Даны действительные числа x, y. Если x и y отрицательны, то каждое значение заменить его модулем; если отрицательно только одно из них, то оба значения увеличить на 0.5; если оба значения неотрицательны и ни одно из них не принадлежит отрезку [0.5, 2.0] то оба значения уменьшить в 10 раз; в остальных случаях x и y оставить без изменения.

10. Даны действительные числа a,b,c ($a \neq 0$). Полностью исследовать биквадратное уравнение $ax^4+bx^2+c=0$, т.е. если действительных корней нет, то должно быть выдано сообщение об этом, иначе должны быть выданы два или четыре корня.

11. Даны числа a, b, c. Упорядочить их следующим образом: в переменную a поместить максимальное из 3-х чисел, в c - минимальное, в b - оставшееся. Если среди чисел есть равные, выдать соответствующее сообщение.

12. Известно, что из четырех чисел, веденных с клавиатуры, одно отлично от трех других, равных между собой. Определить порядковый номер этого числа при вводе.

13. Написать программу для вычисления числа дней в месяце. Даны: номер месяца (целое число), признак високосного года (1 для високосного, 0 - иначе).

14. Дано натуральное трехзначное число. Проверить, является ли оно палиндром (читается одинаково справа налево и наоборот), а также определить наибольшую и наименьшую цифру в его записи.

Контрольные вопросы

1. Назовите основные логические операции в языке Python
2. Постройте блок-схему оператора ветвления (условия)
3. Постройте блок-схему множественного ветвления (из 4 условий)
4. Каким образом оператор выбора реализуется в Python

Лабораторная работа №4

Цикл FOR

Цель работы: изучить синтаксис, особенности и правила работы оператора For.

Краткие теоретические сведения

Инструкция for в языке Python отличается от того, что используется в таких языках как C или Pascal. Вместо того, чтобы всегда перебирать числа арифметической прогрессии (как в Pascal), или предоставлять пользователю полную свободу выбора итератора и условия выхода из цикла (как в C), перебирает элементы произвольной последовательности (например, списка или строки) в порядке их следования:

>>> # Измерение нескольких строк:

... a = ['кот', 'окно', 'выбросить']

>>> for x in a: ... print x, len(x)

кот 3

окно 4

выбросить 9

Небезопасно изменять в цикле итерируемую последовательность (такое возможно только для последовательностей, допускающих изменение, например, списков).

Если Вы собираетесь вносить изменения в список, элементы которого перебираете, например, продублировать избранные элементы, следует перебирать элементы копии исходного списка.

Запись в виде среза делает это особенно удобным:

>>>for x in a[:]: # сделать копию (среза) всего списка ...

If len(x) > 4: a.insert(0, x)

>>> for x in a:

... print x,

выбросить кот окно выбросить

Используя средства функционального программирования, можно одновременно перебирать элементы нескольких последовательностей.

Функции range() и xrange()

Если Вам необходимо перебирать последовательность чисел, то пригодится встроенная функция **range()**.

Она создает список, содержащий арифметическую прогрессию:

>>>range(10)

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Указанная верхняя граница никогда не входит в созданную последовательность. Например, range(10) создает список из 10 значений, точно соответствующих допустимым индексам для элементов последовательности, имеющей длину 10.

Можно указать другую нижнюю границу или другое приращение (шаг), в том числе и отрицательное: (в том числе и инструкция for) требует возможности получить произвольный элемент по индексу.

>>>range(5, 10)

[5, 6, 7, 8, 9]

>>>range(0, 10, 3)

[0, 3, 6, 9]

>>>range(-10, -100, -30)

[-10, -40, -70]

Для того, чтобы перебрать индексы последовательности, используйте совместно range() и len() :

```
>>>a = ['У', 'Марии', 'есть', 'маленькая', 'овечка']
>>> for i in range(len(a)):
        print i, a[i]
0   У
1   Марии
2   есть
3   маленькая
4   овечка
```

Оператор continue

Оператор continue начинает следующий проход цикла, минуя оставшееся тело цикла (for или while)

```
>>> for i in 'hello world':
    if i == 'o':
      continue
    print(i * 2, end='')
hheellll  wwrrlldd
```

Оператор BREAK

Оператор break досрочно прерывает цикл.

```
>>> for i in 'hello world':
      if i == 'o':
        break
      print(i * 2, end='')
hheellll
```

Использование ELSE в циклах

Оператор else, примененный в цикле for или while, проверяет, был ли произведен выход из цикла инструкцией break, или же был выполнен до конца. Блок инструкций внутри else выполнится только в том случае, если выход из цикла произошел без помощи break.

```
>>> for i in 'hello world':
      if i == 'a':
        break
    else:
      print('Буквы а в строке нет')
Буквы а в строке нет
```

Варианты заданий

Задача 1.

1. Дано натуральное N. Вычислить $\frac{\cos 1}{1}+\frac{\cos 2}{1+2}+\frac{\cos 3}{1+2+3}...+\frac{\cos N}{1+2+...+N}$
2. Даны натуральное N и действительное a. Вычислить

$a(a+\sin 1)(a+\sin 1+\sin 2)...(a+\sin 1+...+\sin N)$

3. Даны натуральное N и действительное a. Вычислить $\frac{1}{a}+\frac{1}{a(a+1)}+...+\frac{1}{a(a+1)...(a+N)}$
4. Дано натуральное N. Вычислить $1+(1+\frac{1}{2})+(1+\frac{1}{2}+\frac{1}{3})+...+(1+\frac{1}{2}+...\frac{1}{N})$
5. Дано натуральное N. Вычислить $\frac{2}{\sin 1}+\frac{3}{\sin 1+\sin 2}+...+\frac{N+1}{\sin 1+...+\sin N}$
6. Даны натуральное N и действительное a. Вычислить $\sin a+\sin a^2+\sin a^3+...+\sin a^N$.
7. Даны натуральное N и действительное a. Вычислить $a+(a(a+1))+(a(a+1)(a+2))+...+(a(a+1)...(a+N))$.
8. Дано натуральное N. Вычислить $1+\frac{1}{1*2}+\frac{1}{1*2*3}+...+\frac{1}{N!}$
9. Даны натуральное N и действительное a. Вычислить $1+\frac{1}{a^2}+\frac{1}{a^4}+...+\frac{1}{a^{2N}}$
10. Даны натуральное N и действительное x. Вычислить $x+x(x+1)+x(x+1)(x+2)+...+x(x+1)...(x+N)$.
11. Дано натуральное N. Вычислить $\frac{\cos 1}{\sin 1}+\frac{\cos 1+\cos 2}{\sin 1+\sin 2}+...+\frac{\cos 1+...+\cos N}{\sin 1+...+\sin N}$
12. Дано натуральное N. Вычислить 1+1*2+1*2*3+…+1*2*…*N.
13. Даны натуральное N и действительное a. Вычислить $\frac{1}{a+1}+\frac{2}{(a+1)(a+2)}+...+\frac{N}{(a+1)...(a+n)}$.
14. Дано натуральное N. Вычислить $\frac{1}{\cos 1}+\frac{1+2}{\cos 2}+\frac{1+2+3}{\cos 3}+...+\frac{1+...+N}{\cos N}$

Задача 2. Сочетание цикла и разветвления

1. Даны натуральные числа n, $q_1,\ldots,q_n$. Найти те члены q_i последовательности $q_1,\ldots,q_n$, которые являются удвоенными нечётными числами.

2. Даны целые числа $a_1,\ldots,a_{50}$. Получить:

а) S1 – сумму тех чисел последовательности, которые кратны 5;

б) S2 – сумму тех членов последовательности, которые нечётны и отрицательны;

в) S3 – сумму тех членов последовательности, которые удовлетворяют условию $|a_i| < i^2$.

3 Даны натуральное число n, целые числа $a_1,\ldots,a_n$. Найти количество и сумму тех членов данной последовательности, которые делятся на 5 и не делятся на 7.

4. Даны натуральное n, целые числа p, q, $a_1,\ldots,a_n$ $(p>q\geq 0)$. В последовательности $a_1,\ldots,a_n$ заменить нулями члены, модуль которых при делении на p даёт в остатке q.

5. Даны натуральное число n, действительные числа $a_1,\ldots,a_n$. Вычислить обратную величину произведения тех членов a_i последовательности $a_1,\ldots,a_n$, для которых выполнено $i+1<a_i<i!$.

6. Даны натуральное число n, действительные числа $a_1,\ldots,a_n$. В последовательности $a_1,\ldots,a_n$. Получить сумму членов последовательности значения, которых меньше их порядкового номера.

7. Даны натуральное число n, действительные числа $a_1,\ldots,a_n$. В последовательности $a_1,\ldots,a_n$ получить сумму отрицательных членов и число членов, принадлежащих отрезку [1, 2].

8 .Даны натуральное число n, целые числа $a_1,\ldots,a_n$. Вычислить среднее арифметическое тех членов последовательности, значения которых больше 7.

9. Даны натуральное число n, целые числа a, $x_1,\ldots,x_n$. Если в последовательности $x_1,\ldots,x_n$ есть хотя бы один член, равный a, то получить сумму всех членов, следующих за первым таким членом, иначе – выдать соответствующее сообщение.

10. Даны натуральное число n, целые числа $a_1,\ldots,a_n$. Вывести на экран последовательность $b_1,\ldots,b_n$, которая отличается от исходной тем, что все нечётные члены удвоены.

11. Даны целые числа a, n, $x_1,\ldots,x_n$ $(n>0)$. Определить, каким по счёту идёт в последовательности $x_1,\ldots,x_n$ член, равный a. Если такого члена нет, то выдать соответствующее сообщение.

12. Даны натуральное число n, действительные числа $a_1,\ldots,a_n$. Верно ли, что отрицательных членов в последовательности $a_1,\ldots,a_n$ больше, чем положительных?

13. Даны натуральное число n, действительные числа $a_1,\ldots,a_n$. Верно ли, что наибольший член последовательности $a_1,\ldots,a_n$ по модулю больше единицы?

14. Даны натуральные числа n, $q_1,\ldots,q_n$. Найти те члены q_i последовательности $q_1,\ldots,q_n$, которые при делении на 7 дают остаток 1, 2 или 5.

Задача 3

1. Дано натуральное число N. Определить количество правильных делителей числа. Правильными считаются все делители числа, кроме самого числа, например для числа 6 это 1, 2, 3.
2. Дано натуральное число N. Определить количество правильных делителей данного числа, которые делятся на 3, но не делятся на 4.
3. Дано натуральное число N. Найти наибольший правильный делитель данного числа.
4. Дано натуральное число N. Если число равно сумме всех своих правильных делителей, то оно называется совершенным. Определить, является ли число N совершенным.
5. Дано натуральное число N. Определить, является ли оно простым.
6. Даны натуральные числа N и M. Вывести на экран все делители числа N, которые являются одновременно делителями числа M.
7. Даны натуральные числа N и M. Вычислить количество общих делителей данных чисел.
8. Даны натуральные числа N и M. Вычислить сумму общих делителей данных чисел.
9. Дано натуральное число N. Определить количество правильных делителей данного числа, которые являются четными числами.
10. Дано натуральное число N. Вывести на экран все нечетные делители данного числа.
11. Дано натуральное число N. Вывести на экран все делители данного числа, которые заканчиваются цифрой 3.
12. Даны натуральные числа N, K, L (K<L). Вывести на экран все делители числа N, которые лежат в диапазоне от K до L.
13. Даны натуральные числа N, K, L (K<L). Вывести на экран все делители числа N, которые меньше K или больше L.

Дано натуральное число N. Вывести на экран все делители данного числа, которые заканчиваются цифрой 0.

Контрольные вопросы

1. Нарисуйте бок-схему циклического оператора For
2. Использование операторов range() и xrange() в Python
3. Обход списка с помощью оператора for

Лабораторная работа №5

Цикл с предусловием (пока)

Цель работы: изучить синтаксис, особенности и правила работы оператора WHILE.

Краткие теоретические сведения

В цикле с условием while цикл выполняется, пока истинно задающее его условие. Поэтому этот цикл также иногда называют циклом "пока". Часто цикл while используется в том случае, когда невозможно заранее предсказать, сколько раз необходимо выполнить тело цикла. В повседневной жизни цикл while можно встретить в алгоритмах, например"Пока в пределах видимости есть машины, стоять на месте" или "Пока в ящике есть детали, достать деталь из ящика".

В следующей программе цикл while используется, подобно циклу for, для вывода на экран всех чисел от 1 до 10:

```
i=1
while i<=10:
    print i
    i=i+1
```

В этой программе переменной i присваивается значение 1. Затем начинается цикл (ключевое слово while) с проверяемым условием i<=10. Тело цикла содержит две инструкции: вывод на экран значения переменной i и увеличение значения переменной i на 1.

При выполнении этого цикла проверяется условие i<=10. Поскольку значение i изначально равно 1, то условие верно и выполняется тело цикла: на экран выводится значение переменной i, то есть 1 и переменной i присваивается значение i+1, то есть 2. Снова проверяется условие, поскольку оно верно, то выполняется блок цикла: на экран выводится число 2 и переменной i присваивается значение 3. Опять проверяется значение цикла, и так далее до тех пор, **пока** проверяемое условие истинно.

Как только проверяемое условие станет ложно (это произойдет, когда переменная i станет равна 11), цикл завершит работу и управление будет передано следующей инструкции после блока цикла. Поэтому после завершения цикла переменная i будет иметь значение 11.

В общем виде синтаксис цикла с условием в языке Питон такой:

while *условие*:

инструкция 1

инструкция 2

...

инструкция n

В каждой инструкции while должны присутствовать:

Условие, определяющее, будет ли выполняться тело цикла. Это условие записывается после слова while и может быть произвольным арифметическим выражением, в котором должен быть хотя бы один из операторов ==, !=, <, >, <=, >= и могут использоваться логические операторы and, or, not. После условия ставится двоеточие.

Тело цикла, состоящее из одной или нескольких инструкций, записанных с отступом одинаковой величины.

Инструкции, изменяющие значения переменных, входящих в проверяемое условие. В рассмотренном примере это инструкция i=i+1. Если бы этой инструкции не было, то значение переменной i не менялось бы и проверяемое условие всегда было бы истинным, что привело бы к бесконечному циклу. Для прерывания работы программы, попавшей в бесконечный цикл, используется комбинация клавиш Ctrl+C.

Цикл while в Питоне всегда можно использовать вместо цикла for. Однако иногда цикл for удобней использовать, а иногда удобней применить цикл while, как в следующем примере, где вычисляется наименьшая степень двойки, которая превосходит данное число n:

```
n=input("Введите натуральное число")
i=0
    while 2**i<=n:
        i=i+1
print ("2 в степени",i,"превосходит данное число")
```

В этом примере переменная i внутри цикла увеличивается на 1, пока значение 2**i не превосходит n. После окончания цикла величина 2**i будет больше n, и соответствующее значение i будет напечатано на экране.

Внутри цикла могут быть различные другие инструкции, в том числе инструкции if, while и for. В этом случае говорят о вложенных циклах, или об

условной инструкции, вложенной в цикл. Тело вложенного цикла выделяется от цикла, в который оно вложено, большей величиной отступа.

Рассмотрим два примера. В первом примере программа печатает на экран все натуральные делители данного натурального числа n. Для этого используется цикл, в котором переменная i меняется от 1 до n, а внутри цикла проверяется условие, и если остаток от деления n на i равен 0, то печатается значение i:

```
n=input("Введите число, для которого необходимо вывести делители")
i=1
    while i<=n:
        if n%i==0:
                print( i)
        i=i+1
```

В следующем примере на экран печатается таблица умножения всех однозначных чисел. Для этого организовано два цикла: в одном переменная i меняется от 1 до 9, внутри этого цикла (то есть при каждом новом значении i) переменная j также меняется от 1 до 9. В блоке вложенного цикла на экран печатаются значения переменных i, j и их произведение:

```
    i=1
    while i<10:        # Условие внешнего цикла по i
        j=1
        while j<10:    # Условие внутреннего цикла по j
            print i, "*", j, "=", i*j
            j=j+1      # Инструкция-итератор вложенного цикла
        i=i+1          # Инструкция-итератор внешнего цикла
```

С использованием циклов for этот пример можно записать короче:

```
for i in range(1,10):
        for j in range(1,10):
            print i, "*", j, "=", i*j
```

Варианты заданий

Задача 1

1. Ввести на экран таблицу перевода миль в километры для расстояния от 5 до 75 миль с шагом 5 миль. 1 миля составляет 1,609 км.

2. Дано натуральное число n. Получить наименьшее число вида 2^k, превосходящее n (k - целое положительное).

3. Ввести на экран таблицу значений функции y = $4x^2-2x+5$ для значений x, изменяющихся от -3 до 1, с шагом 0,1.

4. Получить набольшее положительное n, при котором неравенство $-4n+\frac{84}{\sqrt{n}+3}\geq 0$ еще истинно.

5. Ввести на экран таблицу перевода температуры из градусов по шкале Цельсия (C) в градусы шкалы Фаренгейта (F) для значений от a до b градусов с шагом 1° C. Перевод осуществлять по формуле F = 1,8C +32.

6. Получить набольшее положительное n, при котором неравенство $7n^3-103<0$ еще истинно.

7. Ввести на экран таблицу перевода веса в фунтах в вес в килограммах для значений от 1 до 20 фунтов с шагом 2 фунта. (1 фунт = 400 г).

8. Получить наименьшее положительное n, при котором неравенство $3n^3-730>0$ истинно.

9. Вывести на экран таблицу значений функции $y=x^3-x$ на интервале от a до b с шагом 0<h<1.

10. Получить наименьшее положительное n, при котором неравенство $n^4-\frac{1500}{n}>0$ истинно.

11. Ввести на экран таблицу перевода расстояний в дюймах в сантиметры для значений от a до b дюймов в шагом 1 дюйм. (1 дюйм = 2,54 см).

12. Получить набольшее положительное n, при котором неравенство $\ln n-e^n>-1000$ еще истинно.

13. Плотность воздуха убывает с высотой по закону $p=p_0e^{-hz}$. Ввести на экран таблицу зависимости плотности воздуха от высоты для значений от 0 до 1000м с шагом 100м. Считать, что p_0 = 1,29 кг/м^3, z = 1,25*10^{-4} 1/м.

14. Получить набольшее положительное n, при котором неравенство $\ln n-e^n<-1000$ становится истинно.

Задача 2. Разбор натурального числа на цифры

Варианты заданий

1. Дано натуральное N. Найти первую цифру в его записи.

2. Дано натуральное N. Выяснить, входит ли цифра 3 в его запись числа N^2.

3. Дано натуральное N. Определить количество четных цифр в его записи.

4. Дано натуральное N. Поменять порядок цифр числа на обратный.

5. Натуральное число из n цифр является числом Армстронга, если сумма его цифр, возведенных в n-ю степень, равна самому числу (как, например, $153= 1^3 + 5^3 + 3^3$). Дано натуральное N. Выяснить, является ли оно числом Армстронга.

6. Дано натуральное N. Определить количество нечетных цифр в его записи.

7. Даны натуральные N,M.Выяснить, равна ли сумма цифр числа N числу M.

8. Дано натуральное N. Определить, является ли оно палиндромом, т.е. читается одинаково слева направо и справа налево.

9. Дано натуральное N. Определить количество нулей в его записи.

10. Даны натуральные N,M. Выяснить, кратна ли сумма цифр числа N числу M.

11. Дано натуральное N. Удалить из записи числа все четные цифры.

12. Дано натуральное N. Определить сколько в его записи цифр, кратных 3.

13. Дано натуральное N. Определить, совпадает ли запись числа N с последними цифрами записи числа N^2.

14. Даны натуральные N, M ($0 \le M < 10$). Выяснить, есть ли в записи числа N число M.

Задача 3

Варианты заданий

1. Вычислить бесконечную сумму с заданной точностью ε ($\varepsilon > 0$). Считать, что требуемая точность достигнута, если вычислена сумма нескольких первых слагаемых и очередное слагаемое оказалось по модулю меньше, чем ε, - это и все последующие слагаемые можно уже не учитывать. Вычислить: $\sum_{i=1}^{\infty} \frac{(-1)^i}{i!}$;

2. Пусть $X_i = \frac{2i}{(i+1)!}$ для $i \ge 1$. Найти квадрат суммы членов этой последовательности, которые больше заданного числа $\varepsilon (0 < \varepsilon < 1)$. Указать также количество членов последовательности, вошедших в сумму.

3. Вычислить бесконечную сумму с заданной точностьюε ($\varepsilon > 0$).Считать, что требуемая точность достигнута, если вычислена сумма нескольких первых слагаемых и очередное слагаемое оказалось по модулю

меньше, чем ε , - это и все последующие слагаемые можно уже не учитывать. Вычислить: $\sum_{i=0}^{\infty}\frac{(-2)^i}{i!}$;

4. Даны положительные вещественные числа a, x, ε. В последовательности $y_1, y_2, \ldots,$ образованной по закону $y_0 = a;\ y_i = \frac{1}{2}\left(y_{i-1} + \frac{x}{y_{i-1}}\right),\ i = 1,2,\ldots,$ найти первый член y_n, для которого выполнено неравенство $\left|y_n^2 - y_{n-1}^2\right| < \varepsilon$.

5. Пусть $X_i = \frac{i+1}{i+2}$ для i≥1. Найти удвоенную сумму членов этой последовательности, которые больше заданного числа $\varepsilon\ (0 < \varepsilon < 1)$. Указать также количество членов последовательности, вошедших в сумму.

6. Пусть $x_0 = 1;\ x_k = \frac{2 - x_{k-1}^3}{5},\ k = 1,2,\ldots$. Найти первый член x_n, для которого $\left|x_n - x_{n-1}\right| < 10^{-5}$.

7. Пусть $X_i = \frac{1}{2(i+1)}$ для i≥1. Найти квадрат суммы членов этой последовательности, которые больше заданного числа $\varepsilon\ (0 < \varepsilon < 1)$. Указать также количество членов последовательности, вошедших в сумму.

8. Пусть $y_0 = 0;\ y_k = \frac{y_{k-1} + 1}{y_{k-1} + 2},\ k = 1,2,\ldots$. Дано вещественное $\varepsilon > 0$. Найти первый член y_n, для которого выполнено $y_n - y_{n-1} < \varepsilon$.

9. Пусть $X_i = \frac{i}{i^2 + 1}$ для i≥1. Найти сумму и количество членов этой последовательности, которые больше заданного числа $\varepsilon\ (0 < \varepsilon < 1)$.

10. Дано действительное $a > 0$. Последовательность $x_0, x_1, \ldots$ образована по закону $x_0 = \begin{cases} min(2a, 0,95) \bullet при\ \ a \le 1; \\ \frac{a}{5} \quad при \quad 1 < a < 25; \\ \frac{a}{25} \quad при\ остальных\ \ a; \end{cases}$, $x_n = \frac{4}{5}x_{n-1} + \frac{a}{5x_{n-1}^4},\ n = 1,2,\ldots$

Найти первый член x_n, для которого $\left|x_{n+1} - x_n\right| < 10^{-6}$. Вычислить для найденного значения x_n разность $a - x_n^5$.

11. Пусть $X_i = \frac{1}{2(i+1)!}$ для i≥1. Найти удвоенную сумму членов этой последовательности, которые больше заданного числа $\varepsilon\ (0 < \varepsilon < 1)$. Указать также количество членов последовательности, вошедших в сумму.

12. Пусть N - натуральное число и пусть N!! означает $1\cdot3\cdot5\cdot\ldots\cdot N$ для нечётного N и $2\cdot4\cdot\ldots\cdot N$ для чётного N. Для заданного натурального N вычислить $(-1)^{N+1}N!!$.

13. Пусть $X_i = \frac{i}{2i+1}$ для i≥1. Найти сумму и количество членов этой последовательности, которые больше заданного числа $\varepsilon(0<\varepsilon<1)$.

14. Вычислить бесконечную сумму с заданной точностью ε ($\varepsilon>0$). Считать, что требуемая точность достигнута, если вычислена сумма нескольких первых слагаемых и очередное слагаемое оказалось по модулю меньше, чем ε, - это и все последующие слагаемые можно уже не учитывать. Вычислить: $\sum_{i=1}^{\infty}\frac{(-1)^{i+1}}{2i!}$;

Контрольные вопросы

1. Как изображается на блок-схемах цикл с предусловием?
2. Синтаксис оператора WHILE.
3. Правила работы оператора WHILE.
4. Особенности оператора WHILE.
5. Почему цикл с предусловием является самым универсальным циклом?

Лабораторная работа №6

Функции в языке Python

Цель работы: изучить структуру и особенности функций языка Python, научиться создавать и использовать их в программах.

Краткие теоретические сведения

В структурном программировании функция представляет собой именованную последовательность выражений, выполняющих необходимую операцию. В Питоне существует специальный оператор определения функции, имеющий следующий синтаксис:

def имя_функции (список_параметров)

последовательность выражений

Правила выбора имен функций полностью аналогичны правилам выбора имен переменных. Список параметров определяет набор значений, которые могут быть переданы функции в качестве исходных данных – параметры перечисляются через запятую.

Первая строка определения обычно называется заголовком функции, обозначенным ключевым словом def (от англ. «define» – «определить»). Заголовок функции в Питоне завершается двоеточием. После него может следовать любое количество выражений, но они должны быть записаны со смещением относительно начала строки. Для такого смещения можно использовать один или несколько пробелов, или символ табуляции.

```
>>>def printAnything(object):
...     print object
>>> printAnything("Some string")
Some string
>>> number = 15
>>> printAnything(number)
15
```

В первых двух строках примера мы определили функцию PrintAnything(), которая печатает объект, переданный в качестве параметра. Мы можем передать этой функции строку или число, что мы и попробовали сделать в четвертой и седьмой строках примера.

Задача 1

Указания к решению. Выполнить задание, предварительно обосновав целесообразность выбора той или иной процедуры, функции и их параметров.

Варианты заданий

1. Даны действительные числа *s* и t. Получить: *f(t, -2s, 1.17) + f(2.2, t, s-t)*, где $f(a,b,c)=\frac{2a-b-\sin c}{5+|c|}$

2. Даны действительные числа *s* и t. Получить *g(1.2, s) + g(t, s) – g(2s-1, st)*, где $g(a,b)=\frac{a^2+b^2}{a^2+2ab+3b^2+4}$

3. Дано действительное числа y. Получить $\frac{1.7t(0.25)+2t(1+y)}{6-t(y^2-1)}$, где $t(x)=\frac{\sum_{k=0}^{10}\frac{x^{2k+1}}{(2k+1)!}}{\sum_{k=0}^{10}\frac{x^{2k}}{(2k)!}}$

4. Даны действительные числа a,b,c. Получить $\frac{\max(a,a+b)+\max(a,b+c)}{1+\max(a+bc,1,15)}$

5. Даны действительные числа a,b,c,d. Получить max *(A,B,C)*, max*(A,B,D)*, max*(A,C,D)*.

6. Даны действительные числа *s* и t. Получить *h (s,t)*+max*(h²(s-t, st), h⁴(s-t, s+t))+ h(1,1)*, где $h(a,b)=\frac{a}{1+b^2}+\frac{b}{1+a^2}-(a-b)^3$

7. Даны действительные числа a,b,c,d. Получить значение *F* = min *(A,B,C)* + min*(A,B,D)*+ min*(A,C,D)*.

8. Даны координаты четырех точек на плоскости. Для каждой точки вывести на экран: 1) в какой координатный угол она попадает; 2) расстояние от нее до центра координат.

9. Даны действительные числа a,b. Получить *f* = min*(u+v²,3.14)*, где*u* = min*(a,b)*,
v = min*(ab,a+b)*,

10. Даны координаты точек плоскости A, B, C, D. Найти длины отрезков AB, AC, AD.

11. Даны координаты точек плоскости A, B, C. Найти площадь треугольника ABC по формуле Герона: $S_{ABC} = \sqrt{p(p-|AB|)(p-|AC|)(p-|BC|)}$, где p — полупериметр.

12. Даны координаты точек плоскости A, B. Отрезок AB является диагональю прямоугольника. Вычислить площадь и периметр этого прямоугольника.

13. Дано целое число N. Для каждого из N целых чисел, вводимых с клавиатуры, определить, является ли оно простым.

14. Дано целое число N. Для каждого из N целых чисел, вводимых с клавиатуры, найти сумму цифр в его записи.

Задача 2

Задание. Написать программу для решения задачи №3 «Вложенные циклы» лабораторной работы № 5, создав и использовав в программе процедуру или функцию.

Контрольные вопросы

1. Что такое подпрограмма?
2. Какие средства реализации подпрограмм имеют языки программирования?
3. Как определить, что нужно использовать для реализации подпрограммы – процедуру или функцию?
4. Структура функции.
5. Из чего состоит заголовок функции?
6. Каков порядок действий при написании программы с использованием функций?
7. Что такое сфера видимости (область действия) идентификатора?
8. Что такое формальные параметры?
9. Что такое фактические параметры?
10. Что такое локальные переменные?
11. Что такое глобальные переменные?
12. Какие типы параметров Python позволяют реализовывать входные данные процедур и функций и какие – выходные?

Лабораторная работа №7

Работа со строками в языке Python

Цель работы: изучить особенности строк, функции и операторы Pythondля работы с ними, рассмотреть алгоритмы обработки строк .

Краткие теоретические сведения

Записи строк в программном коде Python:

- Строки в апострофах: ‘spam’
- Строки в кавычках: “spa’m”
- Строки в тройных кавычках: ‘’’... spam ...’’’, “””... spam ...”””
- Экранированные последовательности: “s\tp\na\0m”
- Неформатированные строки: r”C:\new\test.spm”
- Строки байтов в версии 3.0: b’sp\x01am’

Наиболее часто используется первые два способа задания строки, двойные и одинарные кавычки являются равнозначными. Экранированные последовательности позволяют вставлять в строки символы, которые сложно ввести с клавиатуры. В конечном строковом объекте символ \ и один или более следующих за ним символов замещаются единственным символом, который имеет двоичное значение, определяемое экранированной последовательностью. Например, ниже приводится строка из пяти символов, в которую вставлены символ новой строки и табуляции:

```
>>> s = ‘a\nb\tc’
```

Последовательность \n образует единственный символ – байт, содержащий двоичное значение кода символа новой строки в используемом наборе символов (обычно ASCII-код 10). Аналогично последовательность \t замещается символом табуляции. Как будет выглядеть такая строка при печати, зависит от того, как она выводится (см. приложение 2).

Если перед открывающей кавычкой стоит символ 'r' (в любом регистре), то механизм экранирования отключается.

```
S = r'C:\newt.txt'
```

Главное достоинство строк в тройных кавычках в том, что их можно использовать для записи многострочных блоков текста. Внутри такой строки возможно присутствие кавычек и апострофов, главное, чтобы не было трех кавычек подряд.

```
>>> c = '''это очень большая
строка, многострочный
блок текста'''
>>> c
'это очень большая\n строка, многострочный\n блок текста'
>>> print(c)
это очень большая
строка, многострочный
блок текста
```

Базовые операции над строками

Конкатенация (сложение)

```
>>> S1 = 'spam'
>>> S2 = 'eggs'
>>> print(S1 + S2)
'spameggs'
```

Дублирование строки

```
>>> print('spam' * 3)
Spamspamspam
```

Длина строки (функция len)

```
>>> len('spam')
4
```

Доступ по индексу

```
>>> S = 'spam'
>>> S[0]
's'
>>> S[2]
'a'
>>> S[-2]
'a'
```

Как видно из примера, в Python возможен и доступ по отрицательному индексу, при этом отсчет идет от конца строки.

Извлечение среза

Оператор извлечения среза: [x:y]. X – начало среза, а Y – окончание; символ с номером y в срез не входит. По умолчанию первый индекс равен 0, а второй - длине строки.

```
>>> s = 'spameggs'
>>> s[3:5]
'me'
>>> s[2:-2]
'ameg'
>>> s[:6]
'spameg'
>>> s[1:]
'pameggs'
>>> s[:]
'spameggs'
```

Кроме того, можно задать шаг, с которым нужно извлекать срез.

```
>>> s[::-1]
'sggemaps'
>>> s[3:5:-1]
''
>>> s[2::2]
'aeg'
```

При вызове методов необходимо помнить, что строки в Python относятся к категории неизменяемых последовательностей, то есть все функции и методы могут лишь создавать новую строку.

```
>>> s = 'spam'
>>> s[1] = 'b'
Traceback (most recent call last):
  File "", line 1, in
```

```
  s[1] = 'b'
TypeError: 'str' object does not support item assignment
>>> s = s[0] + 'b' + s[2:]
>>> s
'sbam'
```

Поэтому все строковые методы возвращают новую строку, которую потом следует присвоить переменной.

Таблица 7.1. Функции и методы строк

Функция или метод	Назначение
S = 'str'; S = "str"; S = '''str'''; S = """str"""	Литералы строк
S = "s\np\ta\nbbb"	Экранированные последовательности
S = r"C:\temp\new"	Неформатированные строки (подавляют экранирование)
S = b"byte"	Строка байтов
S1 + S2	Конкатенация (сложение строк)
S1 * 3	Повторение строки
S[i]	Обращение по индексу
S[i:j:step]	Извлечение среза
len(S)	Длина строки
str in S	Проверка на вхождение подстроки в строку
S.find(str, [start],[end])	Поиск подстроки в строке. Возвращает номер первого вхождения или -1
S.rfind(str, [start],[end])	Поиск подстроки в строке. Возвращает номер последнего 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.isdigit()	Состоит ли строка из цифр
S.isalpha()	Состоит ли строка из букв
S.isalnum()	Состоит ли строка из цифр или букв
S.islower()	Состоит ли строка из символов в нижнем регистре
S.isupper()	Состоит ли строка из символов в верхнем регистре
S.isspace()	Состоит ли строка из неотображаемых символов (пробел, символ перевода страницы ('\f'), "новая строка" ('\n'), "перевод каретки" ('\r'), "горизонтальная табуляция" ('\t') и "вертикальная табуляция" ('\v'))
S.istitle()	Начинаются ли слова в строке с заглавной буквы
S.upper()	Преобразование строки к верхнему регистру
S.lower()	Преобразование строки к нижнему регистру
S.startswith(str)	Начинается ли строка S с шаблона str

Функция или метод	Назначение
S.endswith(str)	Заканчивается ли строка S шаблоном str
S.join(список)	Сборка строки из списка с разделителем S
ord(символ)	Символ в его код ASCII
chr(число)	Код ASCII в символ
S.capitalize()	Переводит первый символ строки в верхний регистр, а все остальные - в нижний
S.center(width, [fill])	возвращает отцентрованную строку, по краям которой символ fill (пробел по умолчанию)
S.count(str, [start],[end])	Возвращает количество непересекающихся вхождений подстроки в диапазоне [начало, конец] (0 и длина строки по умолчанию)
S.expandtabs([tabsize])	Возвращает копию строки, в которой все символы табуляции заменяются одним или несколькими пробелами, в зависимости от текущего столбца. Если TabSize не указан, размер табуляции полагается равным 8 пробелов
S.lstrip([chars])	Удаление пробельных символов в начале строки
S.rstrip([chars])	Удаление пробельных символов в конце строки
S.strip([chars])	Удаление пробельных символов в начале и в конце строки
S.partition(шаблон)	Возвращает кортеж, содержащий часть перед первым шаблоном, сам шаблон, и часть после шаблона. Если шаблон не найден, возвращается кортеж, содержащий саму строку, а затем две пустых строки
S.rpartition(sep)	Возвращает кортеж, содержащий часть перед последним шаблоном, сам шаблон, и часть после шаблона. Если шаблон не найден, возвращается кортеж, содержащий две пустых строки, а затем саму строку
S.swapcase()	Переводит символы нижнего регистра в верхний, а верхнего – в нижний
S.title()	Первую букву каждого слова переводит в верхний регистр, а все остальные - в нижний
S.zfill(width)	Делает длину строки не меньшей width, по необходимости заполняя первые символы нулями
S.ljust(width, fillchar=" ")	Делает длину строки не меньшей width, по необходимости заполняя последние символы символом fillchar
S.rjust(width, fillchar=" ")	Делает длину строки не меньшей width, по необходимости заполняя первые символы символом fillchar
S.format(*args, **kwargs)	Форматирование строки

Варианты заданий

Задача 1.

1. Дана последовательность символов (строка). Заменить все фрагменты вида «начало» на текст «start», все фрагменты вида «конец» на «end» (кавычки не являются частью фрагмента). Большие и малые буквы кириллицы считаются эквивалентными.

2. Задан текст. Определить количество повторений для каждой буквы кириллицы. Данные о буквах вывести в алфавитном порядке. Малые и большие буквы считаются эквивалентными.

3. Дана последовательность символов (строка). Удалит из текста все символы, которые не являются буквами латинского алфавита или кириллицы.

4. Задан текст. Определить, какая буква кириллицы встречается в строке наибольшее количество раз. Если таких букв несколько, то вывести любую.

5. Дана последовательность символов (строка). Если в строке четное количество символов, то удалить из нее все вхождения первого символа, кроме его самого.

6. Задан текст. Вывести в алфавитном порядке все буквы латинского алфавита, которые встречаются в последовательности один раз. Малые и большие буквы считаются эквивалентными.

7. Задан текст. Определить количество повторений для каждой латинской буквы. Данные о буквах вывести в алфавитном порядке. Малые и большие буквы считаются разными.

8. Задан текст. Для каждой цифры, имеющейся в тексте, определить количество повторений.

9. Задан текст. Определить, чего в тексте больше – латинских букв, букв кириллицы или цифр. Большие и малые буквы считаются разными.

10. Задан текст. Найти количество небуквенных символов в этом тексте.

11. Задан текст. Найти количество больших букв (латинских и кириллицы) в этом тексте.

12. Задан текст. Определить, сколько различных небуквенных символов содержит данный текст, и вывести их на экран. Пример: «Ура! Погода – просто чудо!» - 3 различных символа (восклицательный знак, пробел и тире).

Задача 2.

1. Дана последовательность букв латинского алфавита и пробелов. Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Заменить все буквы на большие в тех словах, которые начинаются с заданного символа. Подсчитать количество таких слов.

2. Дана последовательность букв латинского алфавита и пробелов. Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Удалить все слова текста, в которых заданная с клавиатуры буква встречается N раз. Подсчитать количество таких слов.

3. Дана строка, состоящая из букв кириллицы, цифр и пробелов. Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Будем называть особым слово, в котором количество букв и цифр совпадает. Из каждого такого слова удалить цифры и подсчитать количество удаленных символов.

4. Дана последовательность символов латинского алфавита и пробелов. Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Поменять порядок следования букв на обратный в тех словах, которые начинаются с заданного символа. Подсчитать количество таких слов.

5. Дана последовательность символов кириллицы и пробелов. Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Будем называть особым слово, в котором гласных букв больше, чем согласных. Перед каждым таким словом вставить символ *. Подсчитать количество вставленных символов.

6. Дана последовательность символов (строка). Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Словом текста является последовательность букв. Удалить все повторы каждого слова, оставив каждое слово по одному разу. Подсчитать количество удаленных слов.

7. Дана последовательность символов (строка). Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Характеристикой слова считается количество различных букв в слове. Удалить из текста все слова с наибольшей характеристикой.

8. Дана последовательность символов латинского алфавита и пробелов. Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Поменять порядок следования букв на обратный в тех словах, которые имеют наибольшую длину. Подсчитать количество таких слов.

9. Дана последовательность символов. Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Определить сумму цифр, присутствующих в каждом слове.

10. Дана последовательность символов (строка). Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Перед теми словами, которые имеют наибольшую длину, вставить номер по порядку. Подсчитать количество таких слов. Пример: исходный текст – «Мама утром мыла раму мылом», результат: «Мама 1)утром мыла раму 2)мылом».

11. Дана последовательность символов латинского алфавита и пробелов. Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Поменять порядок следования букв на обратный в тех словах, которые имеют четную длину. После каждого такого слова вставить число, равное длине слова.

12. Дана последовательность символов латинского алфавита и пробелов. Группу символов, разделенную с одной или обеих сторон пробелами и не

содержащую внутри себя пробелов, назовем словом. В каждом слове, имеющем нечетную длину, подсчитать количество гласных букв. После каждого такого слова вставить число, равное этому количеству.

13. Дана последовательность символов кириллицы и пробелов. Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Перед каждым словом вставить номер по порядку и найти самое короткое слово.

Пример: исходный текст «Сегодня хорошая погода».
Вывод на экран:
1) Сегодня 2) хорошая 3) погода
Самое короткое слово - погода

14. Дана последовательность символов латинского алфавита и пробелов. Группу символов, разделенную с одной или обеих сторон пробелами и не содержащую внутри себя пробелов, назовем словом. Удалить из слов, имеющих нечетную длину, гласные буквы. После каждого такого слова вставить число, равное длине слова после удаления гласных.

Контрольные вопросы

1. Способы реализации строк в языках программирования и их отличия.
2. Достоинства и недостатки каждого способа реализации строк.
3. Синтаксис описания строк.
4. Экранированные строки.
5. Ввод текста большого объема.
6. Срезы строк.
7. Стандартные процедуры и функции обработки строк: назначение и параметры.
8. Как организовать преобразование малых латинских букв в большие?
9. Как организовать преобразование малых букв кириллицы в большие?
10. На какие группы можно разделить алгоритмы обработки строк?
11. На какие группы можно разделить формальные правила выделения слов в тексте?
12. Модель логического строения текста.
13. Алгоритм выделения слов в тексте и его особенности.

Лабораторная работа №8

Списки в языке программирования Python. Обработка массивов.

Цель: изучить особенности работы со списками в языке Python, познакомиться с основными функциями и операторами работы со списками, изучить алгоритмы сортировки массивов.

Для группировки множества элементов в Python используется список **list**, который может быть записан как индексированная последовательность значений, разделенных запятыми, заключенная в квадратные скобки. Списки имеют произвольную вложенность, т.е. могут включать в себя любые вложенные списки. Физически список представляет собой массив указателей (адресов) на его элементы.

Элементы списка не обязательно должны быть одного типа. Приведем вариант статического определения списка:

>> lst = ['spam', 'drums', 100, 1234]

Как и для строк, для списков нумерация индексов начинается с нуля. Для списка можно получить срез, объединить несколько списков и так далее:

Таблица 8.1. Методы и функции списков

Метод	Результат выполнения
L = []	Пустой список
L = [0, 1, 2, 3]	Четыре элемента с индексами 0..3
L = [‘abc’, [‘def’, ghi’]]	Вложенные списки
L = list(‘spam’)	Создание списка из итерируемого объекта.
L = list(range(-4, 4))	Создание списка из непрерывной последовательности целых чисел
L[i] L[i][j] L[i:j] len(L)	Индекс, индекс индекса, срез, длина
L1 + L2 L * 3	Конкатенация, дублирование
for x in L: print(x) 3 in L	Обход в цикле, проверка вхождения
L.append(4) L.extend([5,6,7]) L.insert(I, X)	Методы: добавление элементов в список
L.index(1) L.count()	Методы: поиск
L.sort() L.reverse()	Методы: сортировка, изменение порядка следования элементов на обратный
del L[k] del L[i:j]	Уменьшение списка

L.pop() L.remove(2) L[i:j] = []	
L[i] = 1 L[i:j] = [4,5,6]	Присваивание по индексу, присваивание срезу

Генераторы списков

Для создания списка, заполненного одинаковыми элементами, можно использовать оператор повторения списка, например:

*A = [0] * n*

Для создания списков, заполненных по более сложным формулам можно использовать *генераторы*: выражения, позволяющие заполнить список некоторой формулой. Общий вид генератора следующий:

[*выражение* for *переменная* in *список*]

где *переменная* — идентификатор некоторой переменной, *список* — список значений, который принимает данная переменная (как правило, полученный при помощи функции range),*выражение* — некоторое выражение, которым будут заполнены элементы списка, как правило, зависящее от использованной в генераторе переменной.

Примеры:

Создать список, состоящий из n нулей можно и при помощи генератора:

A = [0 for i in range(n)]

Создать список, заполненный квадратами целых чисел можно так:

*A = [i ** 2 for i in range(n)]*

Если нужно заполнить список квадратами чисел от 1 до n, то можно изменить параметры функции range на range(1, n + 1):

*A = [i ** 2 for i in range(1, n + 1)]*

Вот так можно получить список, заполненный случайными числами от 1 до 9 (используя функцию randint из модуля random):

A = [randint(1, 9) for i in range(n)]

В этом примере список будет состоять из строк, считанных со стандартного ввода: сначала нужно ввести число элементов списка (это значение будет использовано в качестве аргумента функции range), потом — заданное количество строк:

A = [input() for i in range(int(input()))]

Варианты заданий

Задача 1

1. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [200,1000]. Найти: а) сумму элементов массива, которые являются простыми числами б) максимальный элемент среди тех, которые не являются простыми.
2. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [100,900]. Найти: а) количество элементов массива, которые не являются простыми числами б) минимальный элемент среди тех, цифровая запись которых содержит цифры 3 и 5.
3. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [500,1500]. Найти а) сумму элементов массива, цифровая запись которых дает четную сумму цифр б) максимальный элемент среди тех, которые не являются простыми.
4. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [1000,2000]. Найти: а) количество элементов массива, которые имеют четную сумму делителей б) минимальный элемент среди тех, цифровая запись которых не содержит цифру 0 и содержит цифру 2.
5. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [150,2000]. Найти: а) сумму элементов массива, которые являются четными и имеют два нуля в своей записи б) максимальный элемент среди тех, которые не являются простыми.
6. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [1000,1500]. Найти: а) количество элементов массива, которые не содержат в своей записи цифру 5 б) минимальный элемент среди тех, которые имеют заданное количество делителей.
7. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [1500,2500]. Найти: а) сумму элементов массива, цифровая запись которых дает нечетную сумму цифр б) максимальный элемент среди тех, которые являются простыми.
8. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [2000,3000]. Найти: а) количество элементов массива, которые являются палиндромами (т.е. читаются одинаково слева направо и справа налево) б) минимальный элемент среди тех, цифровая запись которых содержит цифру 0 и не содержит цифру 8.

9. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [200,1000]. Найти: а) сумму элементов массива, цифровая записи которых начинается с цифры 4 б) максимальный элемент среди тех, которые не являются простыми.

10. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [1000,9000]. Найти: а) количество элементов массива, которые являются палиндромами (см.задачу 8) б) минимальный элемент среди тех, цифровая запись которых дает сумму, кратную трем .

11. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [800,1600]. Найти: а) сумму элементов массива, цифровая запись которых дает сумму цифр, кратную 8 б) максимальный элемент среди тех, которые не являются простыми.

12. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [1300,2400]. Найти: а) количество элементов массива, которые имеют нечетную сумму делителей б) минимальный элемент среди тех, цифровая запись которых не содержит цифры 2 и 7.

13. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [1500,3000]. Найти: а) сумму тех элементов массива, которые являются нечетными и имеют в своей записи цифру 7 б) максимальный элемент среди тех, которые являются простыми.

14. Дан массив натуральных чисел A(N), значения элементов которого лежат в диапазоне [1000,5000]. Найти: а) количество тех элементов массива, которые содержат в своей записи цифры 9 и 4 б) минимальный элемент среди тех, которые имеют заданное количество делителей.

Задача 2

Указания к решению:

При решении задачи использовать алгоритмы: сортировка, поиск первого/последнего элемента, удовлетворяющего условию.

1. Ввод элементов массива выполнить с помощью датчика случайных чисел с одновременным выводом его на экран в строку.

2. Для визуальной проверки результатов решения вывод массива выполнить в строку и выделить цветом ту его часть, которая подвергается сортировке.

3. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по возрастанию элементы массива, находящиеся между первым отрицательным и последним положительным. Использовать алгоритм сортировки обменом.

4. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по убыванию элементы массива, находящиеся между последним отрицательным и первым положительным. Использовать алгоритм сортировки вставкой.

5. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по невозрастанию элементы массива, находящиеся между первым и последним элементами, в которых первой цифрой после запятой является 8. Использовать алгоритм сортировки выбором.

6. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по неубыванию элементы массива, находящиеся между первым и последним отрицательными. Использовать алгоритм сортировки обменом.

7. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по возрастанию абсолютных величин элементы массива, находящиеся между первым отрицательным и последним, абсолютное значение которого больше 100. Использовать алгоритм сортировки вставкой.

8. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по убыванию абсолютных величин элементы массива, находящиеся между последним, равным 0 и первым, абсолютное значение которого меньше 100. Использовать алгоритм сортировки выбором.

9. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по невозрастанию абсолютных величин элементы массива, находящиеся между первым и последним элементами, в которых 2 первые цифры после запятой четные. Использовать алгоритм сортировки обменом.

10. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по неубыванию абсолютных величин элементы массива, находящиеся между последним, дробная часть которого больше 0.5 и первым, абсолютное значение которого больше 100. Использовать алгоритм сортировки вставкой.

11. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по возрастанию элементы массива, находящиеся между первым и последним элементами, в которых первой цифрой после запятой является 3. Использовать алгоритм сортировки выбором.

12. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по убыванию элементы массива, находящиеся между первым элементом, в котором первой цифрой после запятой является 3, и последним, в котором после запятой первая цифра 0. Использовать алгоритм сортировки обменом.

13. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по неубыванию элементы массива, находящиеся между первым элементом, в котором первой цифрой после запятой является 5, и первым, в котором после запятой первая цифра 2. Использовать алгоритм сортировки вставкой.

14. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по невозрастанию элементы массива, находящиеся между первым элементом, который в разряде единиц содержит цифру 3, и первым, который в разряде единиц содержит цифру 2. Использовать алгоритм сортировки выбором.

15. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по убыванию абсолютных величин элементы массива, находящиеся между последним элементом, который в разряде единиц не содержит цифру 2, и последним, который в разряде единиц не содержит цифру 1. Использовать алгоритм сортировки обменом.

16. Дан массив действительных чисел A(N), значения элементов которого лежат в диапазоне [a,b]. Отсортировать по возрастанию абсолютных величин элементы массива, находящиеся между первым элементом, который в разряде единиц не содержит цифры 2 и 3, и первым, который в разряде единиц не содержит цифры 7 и 9. Использовать алгоритм сортировки вставкой.

Контрольные вопросы

1. Определения массива и индекса.
2. Особенности одномерного массива.
3. Синтаксис описания одномерного массива.
4. Привести пример типизированного описания массива.
5. Привести пример анонимного описания массива.
6. Главная особенность алгоритмов обработки массивов.
7. Алгоритм ввода одномерного массива с клавиатуры.
8. Особенности применения датчика псевдослучайных чисел для ввода элементов массива.
9. Алгоритм ввода одномерного массива с помощью датчика псевдослучайгых чисел.

10. Алгоритм вывода одномерного массива.
11. Перечислить базовые алгоритмы обработки массивов.
12. Алгоритм поиска элементов массива, удовлетворяющих условию и его особенности.
13. Ошибка «выход за границы массива»: когда возникает и как с ней бороться.
14. Алгоритм вставки элемента массива.
15. Алгоритм удаления элемента массива.
16. Алгоритм поиска максимума-минимума и варианты его модификации.
17. Что такое сортировка элементов массива и какая она бывает?
18. Показатели оценки качества алгоритмов сортировки.
19. Методы и группы алгоритмов сортировки.
20. Алгоритм сортировки обменом (пузырьковой сортировки) и оценка его эффективности.
21. Алгоритм сортировки выбором (линейная сортировки) и оценка его эффективности.
22. Алгоритм сортировки вставкой и оценка его эффективности.
23. Идея сортировки Шелла.
24. Идея сортировки Хоара.
25. Сравнение алгоритмов сортировки.

Лабораторная работа №9. Двумерные массивы

Обработка и вывод вложенных списков

Цель: изучить особенности обработки многомерных массивов в Python

Краткие теоретические сведения

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].

То есть A[0][0] == 1, значение A[0][1] == 2, A[0][2] == 3, A[1][0] == 4, A[1][1] == 5, A[1][2] == 6.

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(len(A)):
    for j in range(len(A[i]):
        print(A[i][j], end = ' ')
    print()
```

То же самое, но циклы не по индексу, а по значениям списка:

```
for row in A:
    for elem in row:
        print(elem, end = ' ')
    print()
```

Естественно для вывода одной строки можно воспользоваться методом join:

```
for row in A:
    print(' '.join(list(map(str, row))))
```

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
```

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. Что же делать?

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
```

Другой (но похожий) 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)
```

Ввод списка

Пусть программа получает на вход двумерный массив, в виде n строк, каждая из которых содержит m чисел, разделенных пробелами. Как их считать? Например, так:

```
A = []
for i in range(n):
    A.append(list(map(int, input().split())))
```

Или, без использования сложных вложенных вызовов функций:

```
A = []
for i in range(n):
    row = input().split()
    for i in range(len(row)):
        row[i] = int(row[i])
    A.append(row)
```

Сложный пример обработки массива

Пусть дан квадратный массив 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 1, элементам, находящимся выше главной диагонали – значение 0, элементам, находящимся ниже главной диагонали – значение 2. То есть получить такой массив (пример для n==4):

```
1 0 0 0
2 1 0 0
2 2 1 0
2 2 2 1
```

Рассмотрим несколько способов решения этой задачи. 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]. Получаем следующий алгоритм:

```
for i in range(n):
    for j in range(n):
```

```
if i < j:
    A[i][j] = 0
elif i > j:
    A[i][j] = 2
else:
    A[i][j] = 1
```

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:

```
for i in range(n):
    for j in range(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
```

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)
```

Задания для выполнения

1. Найдите индексы первого вхождения максимального элемента. Выведите два числа: номер строки и номер столбца, в которых стоит наибольший элемент в двумерном массиве. Если таких элементов несколько, то выводится тот, у которого меньше номер строки, а если номера строк равны то тот, у которого меньше номер столбца. Программа получает на вход размеры массива n и m, затем n строк по m чисел в каждой.

2. Дано нечетное число n. Создайте двумерный массив из n×n элементов, заполнив его символами "." (каждый элемент массива является строкой из одного символа). Затем заполните символами "*" среднюю строку массива, средний столбец массива, главную диагональ и побочную диагональ. В результате единицы в массиве должны образовывать изображение звездочки. Выведите полученный массив на экран, разделяя элементы массива пробелами.

Ввод	Вывод
5	* . * . * . * * * . * * * * * . * * * . * . * . *

3. Даны два числа n и m. Создайте двумерный массив размером n×m и заполните его символами "." и "*" в шахматном порядке. В левом верхнем углу должна стоять точка.

Ввод	Вывод
3 4	. * . * * . * . . * . *

4. Дано число n. Создайте массив размером n×n и заполните его по следующему правилу. На главной диагонали должны быть записаны числа 0. На двух диагоналях, прилегающих к главной, числа 1. На следующих двух диагоналях числа 2, и т.д.

Ввод	Вывод
5	0 1 2 3 4 1 0 1 2 3 2 1 0 1 2 3 2 1 0 1 4 3 2 1 0

5. Дано число n. Создайте массив размером n×n и заполните его по следующему правилу: Числа на диагонали, идущей из правого верхнего в левый нижний угол равны 1. Числа, стоящие выше этой диагонали, равны 0.

Числа, стоящие ниже этой диагонали, равны 2. Полученный массив выведите на экран. Числа в строке разделяйте одним пробелом.

6. Дан двумерный массив и два числа: i и j. Поменяйте в массиве столбцы с номерами i и j и выведите результат. Программа получает на вход размеры массива n и m, затем элементы массива, затем числа i и j. Решение оформите в виде функции

7. Дано число n и массив размером n×n. Проверьте, является ли этот массив симметричным относительно главной диагонали. Выведите слово “YES”, если массив симметричный, и слово “NO” в противном случае. Решение оформите в виде функции.

8. Дан квадратный двумерный массив размером n×n и число k. Выведите элементы k-й по счету диагонали ниже главной диагонали (т.е. если k == 1, то нужно вывести элементы первой диагонали, лежащей ниже главной, если k == 2, то второй диагонали и т.д.). Значение k может быть отрицательным, например, если k == -1, то нужно вывести значение первой диагонали лежащей *выше* главной. Если k == 0, то нужно вывести элементы главной диагонали. Программа получает на вход число n, затем массив размером n×n, затем число k.

9. Дан двумерный массив размером n×m. Симметричный ему относительно главной диагонали массив называется транспонированным к данному. Он имеет размеры m×n: строки исходного массива становятся столбцами транспонированного, столбцы исходного массива становятся строками транспонированного. Для данного массива постройте транспонированный массив и выведите его на экран. Решение оформите в виде функции .

10. Дан двумерный массив размером n×n. Транспонируйте его и результат запишите в этот же массив. Вспомогательный массив использовать нельзя. Решение оформите в виде функции.

11. Дан квадратный массив. Поменяйте местами элементы, стоящие на главной и побочной диагонали, при этом каждый элемент должен остаться в том же столбце (то есть в каждом столбце нужно поменять местами элемент на главной диагонали и на побочной диагонали). Решение оформите в виде функции.

12. В кинотеатре n рядов по m мест в каждом. В двумерном массиве хранится информация о проданных билетах, число 1 означает, что билет на данное место уже продано, число 0 означает, что место свободно. Поступил запрос на продажу k билетов на соседние места в одном ряду. Определите, можно ли выполнить такой запрос.

Программа получает на вход числа n и m. Далее идет n строк, содержащих m чисел (0 или 1), разделенных пробелами. Затем дано число k.

Программа должна вывести номер ряда, в котором есть k подряд идущих свободных мест. Если таких рядов несколько, то выведите номер наименьшего подходящего ряда. Если подходящего ряда нет, выведите число 0.

13. **.** Даны два числа n и m. Создайте массив n×m и заполните его по следующим правилам: Числа, стоящие в строке 0 или в столбце 0 равны 1 (A[0][j] = 1, A[i][0] = 1). Для всех остальных элементов массива A[i][j] = A[i-1][j] + A[i][j-1], то есть каждый элемент равен сумме двух элементов, стоящих слева и сверху от него. Выведите данный массив на экран, отводя на вывод каждого элемента массива ровно 6 символов.

Задача 2.

1. Треугольник Паскаля состоит из чисел, где каждое число равно двум числам, стоящим над ним. Если перенумеровать строки треугольника Паскаля с нуля, то *i*-я строка содержит *i*+1 число, которые равны C_{ij}, где *j* изменяется от 0 до 1 включительно.

По данному числу *n* создайте список из *n* строк, где *i*-й элемент списка должен быть списком, содержащим *i*+1 число — элементы *i*-й строки треугольника Паскаля. Заполните этот массив числами треугольника Паскаля. Выведите результат на экран отводя на вывод одного числа ровно 6 символов.

2. На шахматной доске стоит конь. Отметьте положение коня на доске и все клетки, которые бьет конь. Программа получает на вход координаты коня на шахматной доске в шахматной нотации (то есть в виде “e4”, где сначала записывается номер столбца (буква от “a” до “h”, слева направо), затем номеру строки (цифра от 1 до 8, снизу вверх). Клетку, где стоит конь, отметьте буквой “K”, клетки, которые бьет конь, отметьте символами “*”, остальные клетки заполните точками. Выведите на экран изображение доски.

3. Решите предыдущую задачу для ферзя. Ферзь обозначается буквой “Q”.

4. По данным числам n и m заполните двумерный массив размером n×m числами от 1 до n×m “змейкой”, как показано в примере. Выведите полученный массив, отводя на вывод каждого элемента ровно 4 символа.

5. По данным числам n и m заполните двумерный массив размером n×m числами от 1 до n×m “диагоналями”, как показано в примере. Выведите полученный массив, отводя на вывод каждого элемента ровно 4 символа.

6. Дан прямоугольный массив размером n×m. Поверните его на 90 градусов по часовой стрелке, записав результат в новый массив размером m×n. Выведите получившийся массив. Числа при выводе разделяйте одним пробелом.

7. Дан квадратный массив. Поверните его на 90 градусов по часовой стрелке. Результат запишите в этот же массив, вспомогательный массив использовать нельзя. Выведите результат на экран, разделяя числа одним пробелом.

8. Даны числа n и m. Создайте двумерный массив размером n×m и заполните его таблицей умножения по формуле A[i][j] = i * j. При заполнении массива нельзя использовать вложенные циклы. Выведите получившийся

массив на экран (при выводе можно использовать вложенные циклы), отводя на вывод каждого числа ровно 4 символа.

9. . Даны числа n и m. Заполните массив размером n×m в шахматном порядке: клетки одного цвета заполнены нулями, а другого цвета - заполнены числами натурального ряда сверху вниз, слева направо. В левом верхнем углу записано число 1. Выведите полученный массив на экран, отводя на вывод каждого элемента ровно 4 символа.

10. По данным числам n и m заполните двумерный массив размером n×m числами от 1 до n×m по спирали, выходящей из левого верхнего угла и закрученной по часовой стрелке, как показано в примере. Выведите полученный массив, отводя на вывод каждого элемента ровно 4 символа.

11. : На поле для игры в сапер клеточки с минами обозначаются символом “*”, а в каждой пустой клеточке записано число от 0 до 8, равное количеству мин в 8 клетках, соседних с данной. Дан список мин на поле. Постройте по данному списку изображение поля. Программа получает на вход числа N и M - количество строк и столбцов на поле, а также количество мин на поле K. Далее идет K пар чисел - координат мин. Первое число - номер строки, второе число - номер столбца. Выведите изображение поля на экран, клетки при выводе разделяйте одним пробелом.

12. Дано натуральное число *k*. Сделайте *k*-мерный список размера 2 по каждому измерению, то есть общее число элементов в списке должно быть 2*k*. Заполните список нулями. Выведите результат при помощи функции print без дополнительного форматирования.

13. Дано натуральное число *k*. Сделайте *k*-мерный список размера 2 по каждому измерению, то есть общее число элементов в списке должно быть 2*k*. Список заполните строковыми значениями по формуле: *A*[*i*1][*i*2]...[*ik*]=str(*i*1)+str(*i*2)+...+str(*ik*) Например, если k == 4, то A[0][0][1][0] == '0010'. Выведите результат при помощи функции print без дополнительного форматирования.

Задача 3.

1. Дана матрица A(N,M) натуральных чисел. Найти количество строк, сумма элементов которых является простым числом.

2. Дана матрица B(N,M) натуральных чисел. Найти количество строк, все элементы которых принадлежат отрезку [a,b] и кратны заданному числу X.

3. Дана матрица натуральных чисел A(N,N). Найти среднее арифметическое тех ее элементов, которые являются простыми числами.

4. Дана матрица натуральных чисел A(N,N). Найти количество строк, среднее арифметическое которых больше заданного числа.

5. Дана матрица B(N,M) натуральных чисел. Найти количество строк, в которых меньше половины элементов начинается с заданной цифры X.

6. Дана матрица натуральных чисел A(N,N). Найти количество строк, в которых все элементы, стоящие на четных местах, больше среднего арифметического элементов, стоящих на нечетных местах.

7. Дана матрица натуральных чисел A(N,N). Найти количество строк, в которых сумма элементов, кратных 3, меньше максимального элемента строки.

8. Дана матрица A(N,M) натуральных чисел. Найти количество столбцов, сумма элементов которых является не простым числом.

9. Дана матрица B(N,M) натуральных чисел. Найти количество столбцов, все элементы которых принадлежат отрезку [a,b] и кратны заданному числу X.

10. Дана матрица натуральных чисел A(N,N). Найти среднее арифметическое тех ее элементов, в цифровой записи которых все цифры четные.

11. Дана матрица натуральных чисел A(N,N). Найти количество столбцов, в которых на четных строках стоят четные числа.

12. Дана матрица B(N,M) натуральных чисел. Найти количество столбцов, в которых больше половины элементов начинается с заданной цифры X.

13. Дана матрица натуральных чисел A(N,N). Найти количество столбцов, в которых все элементы, стоящие на четных местах, больше среднего арифметического элементов, стоящих на нечетных местах.

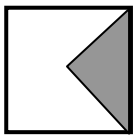
14. Дана матрица натуральных чисел A(N,N). Найти количество столбцов, в которых сумма элементов, кратных 5, меньше максимального элемента строки.

Задача 4

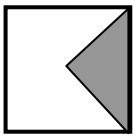
Указания к решению:

Ввод и вывод элементов матрицы оформить в виде функций. Ввод выполнить с помощью датчика случайных чисел.

1. Дана квадратная матрица B(K,K) натуральных чисел. Найти среди элементов, лежащих в выделенной области и имеющих в своей записи более 4 цифр, максимальный. Элементы, расположенные на границах области, входят в заштрихованную часть.

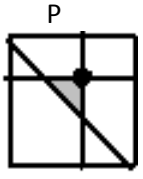


2. Дана квадратная матрица B(K,K) натуральных чисел. Найти среди элементов, лежащих в выделенной области и имеющих в своей записи больше четных цифр, минимальный. Элементы, расположенные на границах области, не входят в заштрихованную часть.



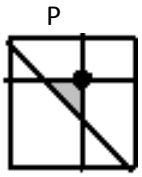
3. Дана квадратная матрица B(K,K) натуральных чисел. Найти среди элементов, лежащих в выделенной области такой, который содержит максимальное количество цифр 5. Элементы, расположенные на границах области, не входят в заштрихованную часть.

4. Дана квадратная матрица B(K,K) натуральных чисел. Найти среди элементов, лежащих в выделенной области, максимальный по модулю элемент. Область задается побочной

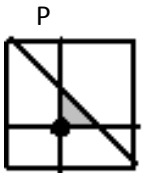


диагональю и координатами элемента B(S,P), через который проводятся горизонтальная и вертикальная линии. Элементы, расположенные на границах области, входят в заштрихованную часть.

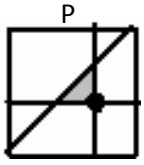
5. Дана квадратная матрица B(K,K) натуральных чисел. Найти среди элементов, лежащих в выделенной области, максимальный среди отрицательных элемент. Область задается главной диагональю и координатами элемента B(S,P), через который проводятся горизонтальная и вертикальная линии. Элементы, расположенные на границах области, не входят в заштрихованную часть.



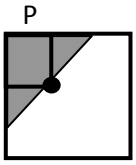
6. Дана квадратная матрица B(K,K) натуральных чисел. Среди положительных элементов, лежащих в выделенной области, найти минимальный. Область задается главной диагональю и координатами элемента B(S,P), через который проводятся горизонтальная и вертикальная линии. Элементы, расположенные на границах области, входят в заштрихованную часть.



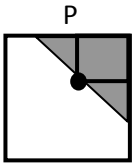
7. Дана квадратная матрица B(K,K) натуральных чисел. Среди нечетных отрицательных элементов, лежащих в выделенной области, найти минимальный. Область задается главной диагональю и координатами элемента B(S,P), через который проводятся горизонтальная и вертикальная линии. Элементы, расположенные на границах области, не входят в заштрихованную часть.



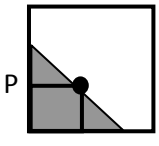
8. Дана квадратная матрица B(K,K) натуральных чисел. Найти максимальный элемент среди тех, который лежат в выделенной области и являются квадратом какого-либо натурального числа. Область задается координатами элемента B(S,P), через который проводится косая линия, параллельная побочной диагонали. Элементы, расположенные на границах области, входят в заштрихованную часть.



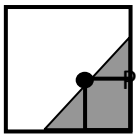
9. Дана квадратная матрица B(K,K) натуральных чисел. Найти максимальный элемент среди тех, который лежат в выделенной области и являются простыми числами. Область задается координатами элемента B(S,P), через который проводится косая линия, параллельная главной диагонали. Элементы, расположенные на границах области, не входят в заштрихованную часть.



10. Дана квадратная матрица B(K,K) натуральных чисел. Найти максимальный элемент среди тех, который лежат в выделенной области и не являются квадратом какого-либо натурального числа. Область задается координатами элемента B(S,P), через который проводится косая линия, параллельная главной диагонали. Элементы, расположенные на границах области, входят в заштрихованную часть.



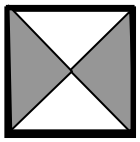
11. Дана квадратная матрица B(K,K) натуральных чисел. Найти минимальный элемент среди тех, который лежат в выделенной области и не являются простыми числами. Область задается координатами элемента B(S,P), через который проводится косая линия, параллельная побочной диагонали. Элементы, расположенные на границах области, не входят в заштрихованную часть.



12. Дана квадратная матрица B(K,K) натуральных чисел. Для каждой косой линии, параллельной главной диагонали (и для самой диагонали) найти максимальный по модулю элемент.

13. Дана квадратная матрица B(K,K) натуральных чисел. Для каждой косой линии, параллельной побочной диагонали (и для самой диагонали) найти минимальный по модулю элемент.

14. Дана квадратная матрица B(K,K) натуральных чисел. Найти среди элементов, лежащих в выделенной области и имеющих в своей записи более двух цифр 0, максимальный. Элементы, расположенные на границах области, входят в заштрихованную часть.



Задача 5

Указания к решению:

Ввод и вывод элементов матрицы оформить в виде функций. Ввод выполнить с помощью датчика случайных чисел.

Варианты заданий

1. Дана квадратная матрица A(N,N) натуральных чисел. Повернуть ее относительно центра симметрии на 180◦ по часовой стрелке.

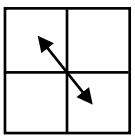
2. Дана квадратная матрица A(N,N) натуральных чисел. Повернуть ее относительно центра симметрии на 270◦ по часовой стрелке.

3. Дана квадратная матрица A(N,N) натуральных чисел. Повернуть ее относительно центра симметрии на 90◦ против часовой стрелки.

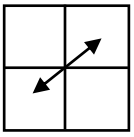
4. Дана квадратная матрица A(N,N) натуральных чисел. Повернуть ее относительно центра симметрии на 180◦ против часовой стрелки.

5. Дана квадратная матрица A(N,N) натуральных чисел. Повернуть ее относительно центра симметрии на 270◦ против часовой стрелки.

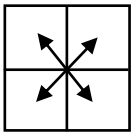
6. Дана квадратная матрица B(2K,2K) натуральных чисел. Матрицы разбита на квадраты. Поменять местами квадраты матрицы согласно рисунку.



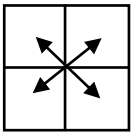
7. Дана квадратная матрица B(2K,2K) натуральных чисел. Матрицы разбита на квадраты. Поменять местами квадраты матрицы согласно рисунку.



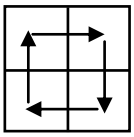
8. Дана квадратная матрица B(2K,2K) натуральных чисел. Матрицы разбита на квадраты. Поменять местами квадраты матрицы согласно рисунку.



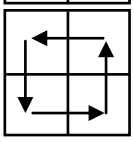
9. Дана квадратная матрица B(2K,2K) натуральных чисел. Матрицы разбита на квадраты. Поменять местами квадраты матрицы согласно рисунку.



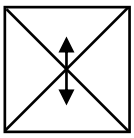
10. Дана квадратная матрица B(2K,2K) натуральных чисел. Матрицы разбита на квадраты. Поменять местами квадраты матрицы согласно рисунку



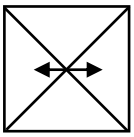
11. Дана квадратная матрица B(2K,2K) натуральных чисел. Матрицы разбита на квадраты. Поменять местами квадраты матрицы согласно рисунку



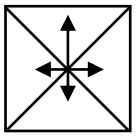
12. Дана квадратная матрица B(2K,2K) натуральных чисел. Матрицы разбита диагоналями на треугольники. Поменять их местами согласно рисунку



13. Дана квадратная матрица B(2K,2K) натуральных чисел. Матрицы разбита диагоналями на треугольники. Поменять их местами согласно рисунку



14. Дана квадратная матрица B(2K,2K) натуральных чисел. Матрицы разбита диагоналями на треугольники. Поменять их местами согласно рисунку



Контрольные вопросы

1. Определения матрицы.
2. Особенности двумерного массива.
3. Синтаксис описания двумерного массива.
4. Привести пример типизированного описания матрицы.
5. Привести пример анонимного описания матрицы.
6. Главная особенность алгоритмов обработки матриц.
7. Алгоритм ввода матрицы с помощью датчика псевдослучайных чисел.
8. Перечислить базовые алгоритмы обработки матриц.
9. Алгоритм поиска элементов матрицы, удовлетворяющих условию и его особенности.
10. Подходы к решению задач на заполнение и преобразование матриц.
11. Вывести логические выражения для элементов главной и побочной диагоналей матрицы.

Лабораторная работа №10

Работа с файлами в PYTHON

Цель работы: освоить технологию работы с файлами в PYTHON

Краткие теоретические сведения

Файлом называется логически связанная именованная область внешней памяти ЭВМ. Файловая система связывает имя файла с цепочкой байт размещенных в областях внешней памяти ЭВМ. Для работы с файлом необходимо создать некоторую переменную, которая будет связана с объектом файла. Например, нам необходимо открыть файл .txt, располагающийся в корневом каталоге диска D: для чтения

```
#coding: utf-8;
c=open('d:\\ .txt','r')
```

Встроенная функция open(name [,mode [,bufsize]]) (аналогична функции file() в предыдущих версиях) открывает файл и создает объект файла, как показано ниже:

```
f = open("foo")     # Откроет "foo" для чтения
f = open("foo",'r') # Откроет "foo" для чтения (как и выше)
f = open("foo",'w') # Откроет "foo" для записи
```

Файл может быть открыт в режиме 'r' – для чтения, в режиме 'w' – для записи и в режиме 'a' – для добавления в конец.

В версии Python 3 в функцию open() было добавлено четыре дополнительных аргумента – она вызывается как open(name [,mode [,bufsize [, encoding [, errors [, newline [, closefd]]]]]]). Аргумент encoding определяет имя кодировки символов, например: 'utf-8' или 'ascii'. Аргумент errors определяет политику обработки ошибок, связанных с кодировкой символов (дополнительная информация о Юникоде приводится в следующих разделах этой главы). Аргумент newline определяет порядок работы в режиме поддержки универсального символа перевода строки и может принимать значения None, '', '\n', '\r' или '\r\n'. Если имеет значение None, любые символы окончания строки, такие как '\n', '\r' или '\r\n', преобразуются в '\n'.

Закрывается файл оператором close()

```
#coding: utf-8;
c=open('d:\\ .txt','r')
x=c.read()
print(x)

c.close()
```

При работе с файлами используется включенная по умолчанию буферизация, при буферизации данные в файл непосредственно при вызове метода записи не записываются, они записываются когда все действия с файлами прекращены или вызвана функция close(). Необходимо отметить, что она является не обязательной.

Таблица 10.1. Операции над файлам в Python

Операция	Интерпретация
output = open(r'C:\spam', 'w')	Открывает файл для записи ('w' означает write – запись)
input = open('data', 'r')	Открывает файл для чтения ('r' означает read – чтение)
input = open('data')	То же самое, что и в предыдущей строке (режим 'r' используется по умолчанию)
aString = input.read()	Чтение файла целиком в единственную строку
aString = input.read(N)	Чтение следующих N символов (или байтов) в строку
aString = input.readline()	Чтение следующей текстовой строки (включая символ конца строки) в строку
aList = input.readlines()	Чтение файла целиком в список строк (включая символ конца строки)
output.write(aString)	Запись строки символов (или байтов) в файл
output.writelines(aList)	Запись всех строк из списка в файл
output.close()	Закрытие файла вручную (выполняется по окончании работы с файлом)
output.flush()	Выталкивает выходные буферы на диск, файл остается открытым
anyFile.seek(N)	Изменяет текущую позицию в файле для следующей операции, смещая ее на N байтов от начала файла.
for line in open('data'):	операции над line Итерации по файлу, построчное чтение
open('f.txt', encoding='latin-1')	Файлы с текстом Юникода в Python 3.0 (строки типа str)
open('f.bin', 'rb')	Файлы с двоичными данными в Python 3.0 (строки типа bytes)

Пример:

Написать программу, которая выводит в файл квадраты целых чисел от одного до 100

```
#coding: utf-8;
c=open('d:\\try .dat','w')
z=list(range(1,101,1))
print(z)
```

```
for x in range (1,101,1):
   x=x**2
   x=str(x)+'  '
   c.write(x)
   if int(x) %10==0:
      c.write('\n')
```

Вывести на экран содержимое файла .dat ранящего целые числа

```
#coding: utf-8;
c=open('d:\\ try.dat','r')
k=c.read()
print (k)
```

Задания на выполнение

Задача 1

l Дано имя файла и целые положительные числа *N* и K. Создать текстовый файл с указанным именем и записать в него *N* строк, каждая из которых состоит из *K* символов «*» (звездочка).

2. Дано имя файла и целое число *N* (0 < *N* < 27). Создать текстовый файл с указанным именем и записать в него *N* строк: первая строка должна содержать *строчную* (то есть маленькую) латинскую букву «a», вторая — буквы «ab», третья — буквы «abc» и т. д.; последняя строка должна содержать *N* начальных строчных латинских букв в алфавитном порядке.

3. Дано имя файла и целое число *N* (0 < *N* < 27). Создать текстовый файл с указанным именем и записать в него *N* строк длины *N;* строка с номером *K* (K = 1, ..., *N)* должна содержать *K* начальных *прописных* (то есть заглавных) латинских букв, дополненных справа символами «*» (звездочка). Например, для *N* = 4 файл должен содержать строки «A***», «AB**», «ABC*», «ABCD».

4. Дан текстовый файл. Вывести количество содержащихся в нем символов и строк (маркеры концов строк EOLN и конца файла EOF при подсчете количества символов не учитывать).

5. Дана строка *S* и текстовый файл. Добавить строку *S* в конец файла.

6. Даны два текстовых файла. Добавить в конец первого файла содержимое второго файла.

7. Дана строка *S* и текстовый файл. Добавить строку *S* в начало файла.

8. Даны два текстовых файла. Добавить в начало первого файла содержимое второго файла.

9. Дано целое число *K* и текстовый файл. Вставить пустую строку перед строкой файла с номером K. Если строки с таким номером нет, то оставить файл без изменений.

10. Дано целое число *K* и текстовый файл. Вставить пустую строку после строки файла с номером *K*. Если строки с таким номером нет, то оставить файл без изменений.

11. Дан текстовый файл. Продублировать в нем все пустые строки.

12. Дана строка *S* и текстовый файл. Заменить в файле все пустые строки на строку S.

13. Дан непустой текстовый файл. Удалить из него первую строку.

Задача 2

1. Дан непустой текстовый файл. Удалить из него последнюю строку.
2. Дано целое число *K* и текстовый файл. Удалить из файла строку с номером *K*. Если строки с таким номером нет, то оставить файл без изменений.
3. Дан текстовый файл. Удалить из него все пустые строки.
4. Даны два текстовых файла. Добавить в конец каждой строки первого файла соответствующую строку второго файла. Если второй файл короче первого, то оставшиеся строки первого файла не изменять.
5. Дано целое число *K* и текстовый файл. Удалить из каждой строки файла первые *K* символов (если длина строки меньше *K*, то удалить из нее все символы).
6. Дан текстовый файл. Заменить в нем все прописные русские буквы на строчные, а все строчные — на прописные.
7. Дан текстовый файл. Заменить в нем все подряд идущие пробелы на один пробел.
8. Дан текстовый файл, содержащий более трех строк. Удалить из него последние три строки.
9. Дано целое число *K* ($0 < K < 10$) и текстовый файл, содержащий более *K* строк. Удалить из файла последние *K* строк.
10. Дано целое число *K* ($0 < K < 10$) и текстовый файл, содержащий более *K* строк. Создать новый текстовый файл, содержащий *K* последних строк исходного файла.
11. Дан текстовый файл. Найти количество абзацев в тексте, если абзацы отделяются друг от друга одной или несколькими пустыми строками.
12. Дано целое число *K* и текстовый файл. Удалить из файла абзац с номером *K* (абзацы отделяются друг от друга одной или несколькими пустыми строками). Пустые строки, предшествующие и следующие за удаляемым абзацем, не удалять. Если абзац с данным номером отсутствует, то оставить файл без изменений.
13. Дан текстовый файл. Найти количество абзацев в тексте, если первая строка каждого абзаца начинается с 5 пробелов («*красная строка*»). Пустые строки между абзацами не учитывать.
14. . Дано целое число *K* и текстовый файл. Удалить из файла абзац с номером *K* (абзацы выделяются с помощью *красной строки* — см. задание 26). Пустые строки между абзацами не учитывать и не удалять. Если абзац с данным номером отсутствует, то оставить файл без изменений.

Задача 3.

1. Дан текстовый файл. Абзацы выделяются в нем с помощью *красной строки* (см. задание 14), а пустых строк нет. Вставить между соседними абзацами по одной пустой строке (в начало и конец файла пустые строки не добавлять).

2. Дан текстовый файл. Вывести первое слово текста наибольшей длины. *Словом* считать набор символов, не содержащий пробелов и ограниченный пробелами или началом/концом строки.

3. Дан текстовый файл. Вывести последнее слово текста наименьшей длины. *Словом* считать набор символов, не содержащий пробелов и ограниченный пробелами или началом/концом строки.

4. Дано целое число *K* и текстовый файл. Создать строковый файл и записать в него все слова длины *K* из исходного файла. *Словом* считать набор символов, не содержащий пробелов, знаков препинания и ограниченный пробелами, знаками препинания или началом/концом строки. Если исходный файл не содержит слов длины *K*, то оставить результирующий файл пустым.

5. Дан символ *C* — *прописная* (заглавная) русская буква и текстовый файл. Создать строковый файл и записать в него все слова из исходного файла, начинающиеся на эту букву (прописную или строчную). *Словом* считать набор символов, не содержащий пробелов, знаков препинания и ограниченный пробелами, знаками препинания или началом/концом строки. Если исходный файл не содержит подходящих слов, то оставить результирующий файл пустым.

6. Дан символ *C* — *строчная* (маленькая) русская буква и текстовый файл. Создать строковый файл и записать в него все слова из исходного файла, содержащие хотя бы одну букву *C* (прописную или строчную). *Словом* считать набор символов, не содержащий пробелов, знаков препинания и ограниченный пробелами, знаками препинания или началом/концом строки. Если исходный файл не содержит подходящих слов, то оставить результирующий файл пустым.

7. Дан текстовый файл, содержащий текст, выровненный по левому краю. Выровнять текст по правому краю, добавив в начало каждой непустой строки нужное количество пробелов (ширину текста считать равной 50).

8. Дан текстовый файл, содержащий текст, выровненный по левому краю. Выровнять текст по центру, добавив в начало каждой непустой строки нужное количество пробелов (ширину текста считать равной 50). Строки нечетной длины перед центрированием дополнять слева пробелом.

9. Дан текстовый файл, содержащий текст, выровненный по правому краю. Выровнять текст по центру, удалив из каждой непустой строки половину начальных пробелов. В строках с нечетным количеством начальных пробелов перед центрированием удалять первый начальный пробел.

10. Дан текстовый файл, содержащий текст, выровненный по левому краю. Абзацы текста разделяются одной пустой строкой. Выровнять текст *по ширине* (то есть и по левому, и по правому краю), увеличив в каждой непустой строке (кроме последних строк абзацев) количество пробелов между словами, начиная с последнего пробела в строке (ширину текста считать равной 50).

11. Дано целое число *K* (> 25) и текстовый файл, содержащий текст, выровненный по левому краю. Абзацы текста отделяются друг от друга одной пустой строкой. Отформатировать текст так, чтобы его ширина не превосходила *K* позиций, и выровнять текст по левому краю, сохранив деление

на абзацы. Пробелы в конце строк удалить. Сохранить отформатированный текст в новом текстовом файле.

12. Дано целое число *K* (> 25) и текстовый файл, содержащий текст, выровненный по левому краю. Абзацы выделяются в нем с помощью *красной строки* (5 начальных пробелов), а пустых строк нет. Отформатировать текст так, чтобы его ширина не превосходила *K* позиций, и выровнять текст по левому краю, сохранив деление на абзацы. Пробелы в конце строк удалить. Сохранить отформатированный текст в новом текстовом файле.

13. Даны два файла целых чисел одинакового размера. Создать текстовый файл, содержащий эти числа, расположенные в два столбца шириной по 30 символов (в первом столбце содержатся числа из первого исходного файла, во втором — из второго файла). В начало и конец каждой строки текстового файла добавить разделитель «|» (код 124). Числа выравниваются по правому краю столбца.

14. . Даны три файла целых чисел одинакового размера. Создать текстовый файл, содержащий эти числа, расположенные в три столбца шириной по 20 символов (в каждом столбце содержатся числа из соответствующего исходного файла). В начало и конец каждой строки текстового файла добавить разделитель «|» (код 124). Числа выравниваются по левому краю столбца.

Задача 4

1. Даны вещественные числа *A, B* и целое число N. Создать текстовый файл, содержащий таблицу значений функции y/X на промежутке [A, B] с шагом (B - A)/N. Таблица состоит из двух столбцов: с аргументами *x* (10 позиций, из них 4 под дробную часть) и со значениями л/X (15 позиций, из них 8 под дробную часть). Столбцы выравниваются по правому краю.

2. Даны вещественные числа A, *B* и целое число N. Создать текстовый файл, содержащий таблицу значений функций sin(x) и cos(x) на промежутке [A, B] с шагом (B - A)/N. Таблица состоит из трех столбцов: с аргументами *x* (8 позиций, из них 4 под дробную часть) и со значениями sin*(x)* и cos*(x)* (по 12 позиций, из них 8 под дробную часть). Столбцы выравниваются по правому краю.

3. Дан текстовый файл, каждая строка которого изображает целое число, дополненное слева и справа несколькими пробелами. Вывести количество этих чисел и их сумму.

4. Дан текстовый файл, каждая строка которого изображает целое или вещественное число, дополненное слева и справа несколькими пробелами (вещественные числа имеют ненулевую дробную часть). Вывести количество чисел с ненулевой дробной частью и их сумму.

5. Дан текстовый файл, каждая строка которого содержит изображения нескольких чисел, разделенные пробелами (вещественные числа имеют ненулевую дробную часть). Создать файл вещественных чисел, содержащий (в том же порядке) все числа из исходного файла, имеющие ненулевую дробную часть.

6. Дан текстовый файл, каждая строка которого изображает целое или вещественное число, дополненное слева и справа несколькими пробелами (вещественные числа имеют ненулевую дробную часть). Вывести количество целых чисел и их сумму.

7. Дан текстовый файл, каждая строка которого содержит изображения нескольких чисел, разделенные пробелами (вещественные числа имеют ненулевую дробную часть). Создать файл целых чисел, содержащий все целые числа из исходного файла (в том же порядке).

8. Дан текстовый файл и файл целых чисел. Добавить в конец каждой строки текстового файла изображение соответствующего числа из файла целых чисел. Если файл целых чисел короче текстового файла, то оставшиеся строки текстового файла не изменять.

9. Дан текстовый файл. В каждой его строке первые 30 позиций отводятся под текст, а оставшаяся часть — под вещественное число. Создать два файла: строковый файл, содержащий текстовую часть исходного файла, и файл вещественных чисел, содержащий числа из исходного файла (в том же порядке).

10. Дан текстовый файл, содержащий таблицу из трех столбцов вещественных чисел. Ширина столбцов таблицы и способ их выравнивания являются произвольными, специальных символов-разделителей таблица не содержит. Создать три файла вещественных чисел, каждый из которых содержит числа из соответствующего столбца таблицы (в том же порядке).

11. Дан текстовый файл, содержащий таблицу из трех столбцов целых чисел. В начале и в конце каждой строки таблицы, а также между ее столбцами располагается *символ-разделитель*. Ширина столбцов таблицы, способ их выравнивания и вид символа-разделителя являются произвольными. Создать файл целых чисел, содержащий сумму чисел из каждой строки исходной таблицы.

12. Дан текстовый файл. Создать файл, содержащий все знаки препинания, встретившиеся в текстовом файле (в том же порядке).

13. Дан текстовый файл. Создать файл, содержащий все символы, встретившиеся в тексте, включая пробел и знаки препинания (без повторений). Символы располагать в порядке их первого появления в тексте.

14. Дан текстовый файл. Создать файл, содержащий все символы, встретившиеся в тексте, включая пробел и знаки препинания (без повторений). Символы располагать в порядке возрастания их кодов.

Контрольные вопросы

1. Дайте определение понятию «Файл»
2. Режим буферизации при работе с файлами
3. Опишите общую технологию работы с файлами в Python
4. Назовите основные способы открытия файлов
5. Приведите пример вывода на экран данных из файла с числами

Список литературы

1. "Python programming language" - официальный сайт. Новости языка программирования Python. Документация, дистрибутивы для скачивания. - www.python.org
2. Уроки программирования на Python. www.pythonworld.ru
3. Python для новичков. - http://python.su/

Приложения

Приложение 1

Таблица операции преобразования типов данных.

Функция	Описание
int(x [,base])	Преобразует объект x в целое число. Если x является строкой, аргумент base определяет основание системы счисления.
float(x)	Преобразует объект x в число с плавающей точкой
complex(real [,imag])	Преобразует объект x в комплексное число.
str(x)	Преобразует объект x в строковое представление.
repr(x)	Преобразует объект x в строковое выражение.
format(x [,format_spec])	Преобразует объект x в форматированную строку.
eval(str)	Вычисляет выражение в строке str и возвращает объект.
tuple(s)	Преобразует объект s в кортеж.
list(s)	Преобразует объект s в список
set(s)	Преобразует объект s в множество.
dict(d)	Создает словарь. Объект d должен быть последовательностью кортежей (key, value).
frozenset(s)	Преобразует объект s в множество типа frozenset.
chr(x)	Преобразует целое число x в символ.
ord(x)	Преобразует одиночный символ в целое число.
oct(x)	Преобразует целое число x в строку с восьмеричным представлением.
hex(x)	Преобразует целое число x в строку с шестнадцатеричным представлением.
bin(x)	Преобразует целое число x в строку с двоичным представлением.

Приложение 2.

Операторы работы с экранированными последовательностями

Экранированная последовательность	Назначение
\n	Перевод строки
\a	Звонок
\b	Забой
\f	Перевод страницы
\r	Возврат каретки
\t	Горизонтальная табуляция

Экранированная последовательность	Назначение
\v	Вертикальная табуляция
\N{id}	Идентификатор ID базы данных Юникода
\uhhhh	16-битовый символ Юникода в 16-ричном представлении
\Uhhhh…	32-битовый символ Юникода в 32-ричном представлении
\xhh	16-ричное значение символа
\ooo	8-ричное значение символа
\0	Символ Null (не является признаком конца строки)

Приложение 3. Операторы модуля Random

Модуль random предоставляет функции для генерации случайных чисел, букв, случайного выбора элементов последовательности.

random.seed([X], version=2) - инициализация генератора случайных чисел. Если X не указан, используется системное время.

random.getstate() - внутреннее состояние генератора.

random.setstate(state) - восстанавливает внутреннее состояние генератора. Параметр state должен быть получен функцией getstate().

random.getrandbits(N) - возвращает N случайных бит.

random.randrange(start, stop, step) - возвращает случайно выбранное число из последовательности.

random.randint(A, B) - случайное целое число N, $A \le N \le B$.

random.choice(sequence) - случайный элемент непустой последовательности.

random.shuffle(sequence, [rand]) - перемешивает последовательность (изменяется сама последовательность). Поэтому функция не работает для неизменяемых объектов.

random.sample(population, k) - список длиной k из последовательности population.

random.random() - случайное число от 0 до 1.

random.uniform(A, B) - случайное число с плавающей точкой, $A \le N \le B$ (или $B \le N \le A$).

random.triangular(low, high, mode) - случайное число с плавающей точкой, $low \le N \le high$. Mode - распределение.

random.betavariate(alpha, beta) - бета-распределение. alpha>0, beta>0. Возвращает от 0 до 1.

random.expovariate(lambd) - экспоненциальное распределение. lambd равен 1/среднее желаемое. Lambd должен быть отличным от нуля. Возвращаемые значения от 0 до плюс бесконечности, если lambd положительно, и от минус бесконечности до 0, если lambd отрицательный.

random.gammavariate(alpha, beta) - гамма-распределение. Условия на параметры alpha>0 и beta>0.

random.gauss(значение, стандартное отклонение) - распределение Гаусса.

random.lognormvariate(mu, sigma) - логарифм нормального распределения. Если взять натуральный логарифм этого распределения, то Вы получите нормальное распределение со средним mu и стандартным отклонением sigma. mu может иметь любое значение, и sigma должна быть больше нуля.

random.normalvariate(mu, sigma) - нормальное распределение. mu - среднее значение, sigma - стандартное отклонение.

random.vonmisesvariate(mu, kappa) - mu - средний угол, выраженный в радианах от 0 до 2π, и kappa - параметр концентрации, который должен быть больше или равен нулю. Если каппа равна нулю, это распределение сводится к случайному углу в диапазоне от 0 до 2π.

random.paretovariate(alpha) - распределение Парето.

random.weibullvariate(alpha, beta) - распределение Вейбулла.