Министерство науки и высшего образования Российской Федерации

Калужский филиал

федерального государственного бюджетного образовательного

учреждения высшего образования

**«Московский государственный технический университет**

**имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

(КФ МГТУ им. Н.Э. Баумана)

Ю.С.Белов, И.И.Ерохин

ЛИНЕЙНЫЕ СТРУКТУРЫ НА ЯЗЫКЕ PYTHON

Методические указания к лабораторной работе

по дисциплине «Высокоуровневое программирование»

Калуга – 2019

УДК 004.62

ББК 32.972.1

Б435

Методические указания составлены в соответствии с учебным планом КФ МГТУ им. Н.Э. Баумана по направлению подготовки 09.03.04 «Программная инженерия» кафедры «Программного обеспечения ЭВМ, информационные технологии».

Методические указания рассмотрены и одобрены:

|  |  |  |  |
| --- | --- | --- | --- |
| - Кафедрой «Программного обеспечения ЭВМ, информационных технологий» (ИУ4-КФ) протокол №  3  от « 24 »  октября  2018 г. | | | |
| Зав. кафедрой ИУ4-КФ | \_\_\_\_\_\_\_\_\_\_\_\_\_к.т.н., доцент Ю.Е. Гагарин | | |
|  |  | | |
| - Методической комиссией факультета ИУ-КФ протокол №\_\_\_ от «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2019 г. | | | |
|  | | | |
| Председатель методической комиссии факультета ИУ-КФ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_к.т.н., доцент М.Ю. Адкин | | |
|  |  | | |
| - Методической комиссией  КФ МГТУ им.Н.Э. Баумана протокол №\_\_\_ от «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2019 г. | | | |
|  | | | |
| Председатель методической комиссии КФ МГТУ им.Н.Э. Баумана | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_д.э.н., профессор О.Л. Перерва | |
|  | |  | |
| Рецензент:  зав. кафедрой ИУ2-КФ «Информационные системы и сети», к.т.н., доцент | | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.В. Чухраев |
|  | | |  |
| Авторы  к.ф.-м.н., доцент кафедры ИУ4-КФ  асс.кафедры ИУ4-КФ | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ю.С.Белов  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_И.И.Ерохин | |

Аннотация

Методические указания к выполнению лабораторной работы по курсу «Высокоуровневое программирование» содержат общие сведения о программах линейной структуры, описание типовых алгоритмов работы с основными типами данных, а также описание средств их реализации на языке программирования Python.

Предназначены для студентов 2-го курса бакалавриата КФ МГТУ им. Н.Э. Баумана, обучающихся по направлению подготовки 09.03.04 «Программная инженерия».

© Калужский филиал МГТУ им. Н.Э. Баумана, 2019 г.

© Ю.С.Белов, И.И.Ерохин, 2019 г.

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ……………………………..………………………………….4](#_Toc536561827)

[ЦЕЛЬ И ЗАДАЧИ РАБОТЫ, ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ЕЕ ВЫПОЛНЕНИЯ………………………………………………………..5](#_Toc536561828)

[КРАТКАЯ ХАРАКТЕРИСТИКА ОБЪЕКТА ИЗУЧЕНИЯ, ИССЛЕДОВАНИЯ………………………………………………………...6](#_Toc536561829)

[КОНСОЛЬНЫЙ ВВОД/ВЫВОД…………………………………………8](#_Toc536561830)

[ЛОГИЧЕСКИЕ И АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ………………..11](#_Toc536561831)

[ЗАДАЧИ И ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ……………………18](#_Toc536561832)

[ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ…………………………..24](#_Toc536561833)

[КОНТРОЛЬНЫЕ ВОПРОСЫ И ЗАДАНИЯ……………………………44](#_Toc536561834)

[ФОРМА ОТЧЕТА ПО ЛАБОРАТОРНОЙ РАБОТЕ…………………...45](#_Toc536561835)

[ОСНОВНАЯ ЛИТЕРАТУРА…………………………………………….46](#_Toc536561836)

[ДОПОЛНИТЕЛЬНАЯ ЛИТЕРАТУРА………………………………….46](#_Toc536561837)

**ВВЕДЕНИЕ**

Настоящие методические указания составлены в соответствии с программой проведения лабораторных работ по курсу «Высокоуровневое программирование» на кафедре «Программное обеспечение ЭВМ, информационные технологии» факультета «Информатика и управление» Калужского филиала МГТУ им. Н.Э. Баумана.

Методические указания, ориентированные на студентов 2-го курса направления подготовки 09.03.04 «Программная обеспечение ЭВМ, информационные технологии», содержат сведения об алгоритмах линейной структуры и средствах их реализации на языке Python.

Методические указания составлены для ознакомления студентов с основополагающими понятиями и принципами разработки программ, решение которых предполагает использование алгоритмов линейной структуры. Для выполнения лабораторной работы студенту необходимы минимальные теоретические знания курса алгебры.

**ЦЕЛЬ И ЗАДАЧИ РАБОТЫ, ТРЕБОВАНИЯ  
К РЕЗУЛЬТАТАМ ЕЕ ВЫПОЛНЕНИЯ**

Целью выполнения лабораторной работы является формирование практических навыков процедурного программирования, разработки и отладки программ, овладение методами и средствами разработки и оформления технической документации.

Основными задачами выполнения лабораторной работы являются:

1. Изучить структуру программы на языке Python.
2. Познакомиться с основными типами данных в Python.
3. Изучить операторы ввода, вывода и присвоения, логических и арифметических операций.

[Результатами](#ЗАДАНИЕ) работы являются:

1. Реализация разработанных алгоритмов на языке программирования Python;
2. Подготовленный отчет.

**КРАТКАЯ ХАРАКТЕРИСТИКА ОБЪЕКТА ИЗУЧЕНИЯ, ИССЛЕДОВАНИЯ**

Python представляет популярный высокоуровневый язык программирования, предназначенный для создания приложений различных типов: это и веб-приложения, и игры, и настольные программы, и работа с базами данных. Довольно большое распространение питон получил в области машинного обучения и исследований искусственного интеллекта.

Программа на языке Python состоит из набора инструкций. Каждая инструкция помещается на новую строку. Например:

print(2 + 3)   
print(**"Hello"**)

Большую роль в Python играют отступы, так как именно по их величине вложенные инструкции объединяются в блоки. Отступ может быть любым, главное, чтобы в пределах одного вложенного блока отступ был одинаков. Для отступа используйте 4 пробела (или знак табуляции). Наличие “лишнего” отступа приведёт к ошибке в компиляции программы:

print(2 + 3)   
 print(**"Hello"**)

Поэтому стоит помещать новые инструкции сначала строки. В этом одно из важных отличий Python от других языков программирования, как C# или Java.

Однако стоит учитывать, что некоторые конструкции языка могут состоять из нескольких строк. Например, условная конструкция if:

**if** 1 < 2:  
 print(**"Hello"**)

В данном случае если 1 меньше 2, то выводится строка "Hello". И здесь уже должен быть отступ, так как инструкция print("Hello") используется не сама по себе, а как часть условной конструкции if. Python - регистрозависимый язык, поэтому выражения print и Print или PRINT представляют разные выражения. И если вместо метода print для вывода на консоль использовать метод Print, то будет ошибка:

Print(**"Hello World"**)

Комментарии на Python обозначаются решеткой #. Их можно писать как над строкой, так и рядом. Например:

*# Вывод сообщения на консоль*print(**"Hello World"**)  
print(**"Hello World"**) *# Вывод сообщения на консоль*

**КОНСОЛЬНЫЙ ВВОД/ВЫВОД**

**Консольный вывод**

Основной функцией для вывода информации на консоль является функция print(). В качестве аргумента в эту функцию передается строка:

print(**"Hello Python"**)

Если же нам необходимо вывести несколько значений на консоль, то мы можем передать их в функцию print через запятую:

print(**"Full name:"**, **"Tom"**, **"Smith"**)

В итоге все переданные значения склеятся через пробелы в одну строку:

Full name: Tom Smith

Так же в print можно передать [математическое](#Математика) выражение:

print(6 + 2) *# 8*

На экран будет выводится полученное значение.

Для Python так же характерен форматированный вывод. Существуют спецификаторы. Они аналогичны тем, что используются в других языках программирования. Буква после процента обозначает [тип числа](#Типы). Например, 0.%f означает, что выводится 0 знаков за запятой:

* %d для вывода целого числа
* %f для вывода числа с точкой
* %s для вывода строки
* %c для вывода символа
* %e для вывода числа с e

**Консольный ввод**

Ввод данных с клавиатуры осуществляется с помощью функции input(). После выполнения данной функции программа ожидает ввода данных и после нажатия "Enter" записывает их в переменную или возвращает результат на экран.

Существуют 4 вида ввода в зависимости от [типа данных](#Типыданных):

1. a = int(input())

Это означает, что в переменную a будет записано целое число (int - integer). Для работы программы надо будет ввести любое целое число и нажать "Enter". После этого программа начнет работать и выведет введенное число.

a=int(input())  
print(a)

1. a = float(input())

"float" означает ввод действительного числа. При вводе нецелого числа обязательно разделить целую и дробную часть точкой. Например, "34.17"

1. a = str(input())

"str" означает ввод строки. Например, "qwerty" или "Hello, Python!" (str - string).

1. a,b=input().split()

Это означает ввод любых двух и более элементов. Скобки после "split" используются для того, чтобы задать, чем будут разделяться переменные (по умолчанию пробел). Например:

a,b=input().split(**'.'**)

С помощью этих функций можно преобразовать действительное число в дробное (при этом число округляется в меньшую сторону), число в строку и так далее:

a=float(input())  
a=int(a)  
print(a)

Например, введено число 2.9, но на экране выведется 2, т.к. округление в меньшую сторону.

В математических функциях есть функция [round()](#round), позволяющая округлить до заданного знака.

**ЛОГИЧЕСКИЕ И АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ**

**Типы данных**

Чтобы совершать [арифметические](#Математика) и [логические операции](#Логика), необходимо ознакомиться с основными типами данных.

В Python существует множество различных типов данных, которые подразделяются на категории: числа, последовательности, словари, наборы:

* boolean - логическое значение True или False
* int - представляет целое число, для хранения которого использует 4 байта в памяти компьютера.
* float - представляет число с плавающей точкой, для хранения которого используется 8 байт, например, 1.2 или 34.76
* complex - комплексные числа
* str - строки, например "hello". В Python 3.x строки представляют набор символов в кодировке Unicode
* bytes - последовательность чисел в диапазоне 0-255
* byte array - массив байтов, аналогичен bytes с тем отличием, что может изменяться
* list - список
* tuple - кортеж
* set - неупорядоченная коллекция уникальных объектов
* frozen set - то же самое, что и set, только не может изменяться (immutable)
* dict - словарь, где каждый элемент имеет ключ и значение

Python является языком с [динамической типизацией](#Преобразование). Он определяет тип данных переменной исходя из значения, которое ей присвоено. Так, при присвоении строки в двойных или одинарных кавычках переменная имеет тип str. При присвоении целого числа Python автоматически определяет тип переменной как int. Чтобы определить переменную как объект float, ей присваивается дробное число, в котором разделителем целой и дробной части является точка.

Число с плавающей точкой можно определять в экспоненциальной записи:

x = 3.9e3  
print(x) *# 3900.0*x = 3.9e-3  
print(x) *# 0.0039*

Число float может иметь только 18 значимых символов. Так, в данном случае используются только два символа - 3.9. И если число слишком велико или слишком мало, то можно записывать число в подобной нотации, используя экспоненту. Число после экспоненты указывает степень числа 10, на которое надо умножить основное число - 3.9.

При этом в процессе работы программы существует возможность изменить тип переменной, присвоив ей значение другого типа:

user\_id = **"12tomsmith438"** *# тип str*print(user\_id)  
user\_id = 234 *# тип int*print(user\_id)

С помощью функции type() динамически можно узнать текущий тип переменной:

user\_id = **"12tomsmith438"**print(type(user\_id)) *# <class 'str'>*user\_id = 234  
print(type(user\_id)) *# <class 'int'>*

**Арифметические операции**

Python поддерживает все распространенные арифметические операции:

print(6 + 2) *# сложение двух чисел, результат = 8*print(6 - 2) *# вычитание двух чисел, результат = 4*print(6 \* 2) *# умножение двух чисел, результат = 12*print(7 / 2) *# деление двух чисел, результат = 3.5*print(7 // 2) *# целочисленное деление, результат = 3*print(6 \*\* 2) *# возведение в степень (второе значение – показатель степени), результат = 36*print(7 % 2) *# деление по модулю, результат = 1*

**Арифметические операции с присвоением**

Ряд специальных операций позволяют использовать присвоить результат операции первому операнду:

number = 10  
number += 5 *# присвоение результата сложения*print(number) *# результат = 15*number -= 3 *# присвоение результата вычитания*print(number) *# результат = 12*number \*= 4 *# присвоение результата умножения*print(number) *# результат = 48*number /= 6 *# присвоение результата деления*print(number) *# результат = 8*number //= 3 *# присвоение результата целочисленного деления*print(number) *# результат = 2*number \*\*= 5 *# присвоение степени числа*print(number) *# результат = 32*number %= 7 *# присвоение остатка от деления*print(number) *# результат = 4*

Для того, чтобы не возникало конфликтов с более сложными математическими операциями, необходимо в программу импортировать библиотеку math. Она позволяет пользователю использовать тригонометрические функции, логарифмы и многое другое.

**Функции преобразования чисел**

Ряд встроенных функций в Python позволяют работать с числами. В частности, функции int() и float() позволяют привести значение к типу int и float соответственно.

Например, пусть будет следующий код:

first\_number = **"2"**second\_number = 3  
third\_number = first\_number + second\_number

Пользователь считает, что "2" + 3 будет равно 5. Однако этот код сгенерирует исключение, так как первое число на самом деле представляет строку. И чтобы все заработало как надо, необходимо привести строку к числу с помощью функции int():

first\_number = **"2"**second\_number = 3  
third\_number = int(first\_number) + second\_number  
print(third\_number) *# 5*

Аналогичным образом действует функция float(), которая преобразует в число с плавающей точкой. Но вообще с дробными числами надо учитывать, что результат операций с ними может быть не совсем точным. Например:

first\_number = 2.0001  
second\_number = 5  
third\_number = first\_number / second\_number  
print(third\_number) *# 0.40002000000000004*

В данном случае желательно получить число 0.40002, однако в конце через ряд нулей появляется еще какая-то четверка. Или еще одно выражение:

print(2.0001 + 0.1) *# 2.1001000000000003*

В этом случае для округления результата возможно использование функции round():

first\_number = 2.0001  
second\_number = 0.1  
third\_number = first\_number + second\_number  
print(round(third\_number, 4)) *# 2.1001*

Первый параметр функции - округляемое число, а второй - сколько знаков после запятой должно содержать получаемое число. Но можно использовать и [форматированный вывод](#Форматвывод).

**Представление числа**

При обычном определении числовой переменной она получает значение в десятичной системе. Но кроме десятичной в Python доступны двоичные, восьмеричные и шестнадцатеричные системы.

Для определения числа в двоичной системе перед его значением ставится 0 и префикс b:

x = 0b101 *# 101 в двоичной системе равно 5*Для определения числа в восьмеричной системе перед его значением ставится 0 и префикс o:  
  
a = 0011 *# 11 в восьмеричной системе равно 9*Для определения числа в шестнадцатеричной системе перед его значением ставится 0 и префикс x:  
  
y = 0x0a *# a в шестнадцатеричной системе равно 10*

И с числами в других системах измерения также можно проводить [арифметические операции](#Математика):

x = 0b101 *# 5*y = 0x0a *# 10*z = x + y *# 15*print(**"{0} in binary {0:08b} in hex {0:02x} in octal {0:02o}"**.format(z))

[Для вывода числа](#Вывод) в различных системах исчисления используются функция format, которая вызывается у строки. В эту строку передаются различные форматы. Для двоичной системы "{0:08b}", где число 8 указывает, сколько знаков должно быть в записи числа. Если знаков указано больше, чем требуется для числа, то ненужные позиции заполняются нулями. Для шестнадцатеричной системы применяется формат "{0:02x}". И здесь все аналогично - запись числа состоит из двух знаков, если один знак не нужен, то вместо него вставляется ноль. А для записи в восьмеричной системе используется формат "{0:02o}".

Результат работы: 15 in binary 00001111 in hex 0f in octal 17

**Логические операции**

В Python имеются следующие логические операторы, аналогичные в других языках программирования:

* and (логическое умножение)
* or (логическое сложение)
* not (логическое отрицание)

Если в одном выражении одновременно используется несколько или даже все логические операторы, то следует учитывать, что они имеют разные приоритеты. Вначале выполняется оператор not, затем оператор and, а в конце оператор or.

**ЗАДАЧИ И ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ**

Рассмотрим выполнение лабораторной работы на примере следующей задачи: даны действительные числа x,y,z. Вычислить:

A=;

B=.

Для начала работы необходимо создать новый файл (название любое, не содержащее кириллицы и знаков) в среде разработки IDLE Python.

start-idle

Рис.1 Cреда разработки на компьютере

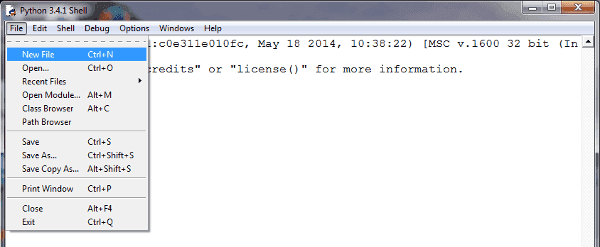


Рис.2. Создание нового файла программы

В появившемся окне писать код программы. Запуск осуществляется с помощью кнопки Run в панели.

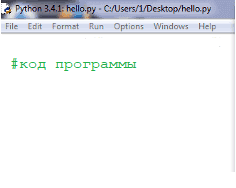


Рис.3. Поле для ввода кода

Имеется сложное алгебраическое выражение. Для этого нужно подключить библиотеку math. Она помогает избавится от конфликтов, происходящих во время [математических операций](#Математика).

**import** math

Затем создается [приглашение пользователю](#Вывод) ввести заданные числа.

Следует использовать методы [для ввода числа](#Ввод), в данном случае здесь будет float (по условию):

print(**"Введите х"**)  
x=float(input())  
print(**"Введите у"**)  
y=float(input())  
print(**"Введите z"**)  
z=float(input())

Далее перечислен список функций из библиотеки math, необходимых для решения задачи:

* pow(аргумент, степень) – аналог возведения степени;
* sin(),cos(),tan() – тригонометрические функции;
* log(выражение) – десятичный логарифм.

Следует отметить, что при возникновении ошибки желательно писать функции таким образом: math.pow.

Выражения в Python выглядят так:

A=((1+pow(math.sin(x+y),2))/(2+abs(x-2\*x/(1+pow(x\*y,2)))))+z\*\*3  
B=pow(x+y,2)/math.log(pow(x,2)+1)-pow(math.cos(x),2)+math.tan(z)

Далее выводятся полученные выражения на экран. Для корректного вывода используется [форматирование](#Форматвывод):

print(**"A= "**,**"%.2f"** % A)  
print(**"B= "**,**"%.2f"** % B)

Итоговый код:

**import** math  
print(**"Введите х"**)  
x=float(input())  
print(**"Введите у"**)  
y=float(input())  
print(**"Введите z"**)  
z=float(input())  
A=((1+pow(math.sin(x+y),2))/(2+abs(x-2\*x/(1+pow(x\*y,2)))))+z\*\*3  
B=pow(x+y,2)/math.log(pow(x,2)+1)-pow(math.cos(x),2)+math.tan(z)  
print(**"A= "**,**"%.2f"** % A)  
print(**"B= "**,**"%.2f"** % B)

Результат работы: Введите х

2

Введите y

4

Введите z

5

A=125,27;

B=18,81

Следующая задача посвящена [операциям деления](#Математика). Условие таково: написать программу для определения числа, получаемого выписыванием в обратном порядке цифр заданного трехзначного числа.

Здесь снова необходим импорт математической библиотеки

**import** math

Приглашение ввести трехзначное целое число, [используя методы ввода](#Ввод):

print(**"Введите трехзначное число"**)  
olddig=int(input())

Далее записывается выражение, позволяющее записать цифры в обратном порядке:

newdig =(olddig%10)\*100+((olddig%100)/10)\*10+olddig/100 # первая цифра – это единица, она записывается на место сотни, т.е. старое число делится на 10, его остаток умножается на 100. Аналогично делаются остальные.

Далее полученное значение выводится на экран:

print(**"Число, получившееся в результате преобразования "**,**"%.0f"** % newdig)  
  
Итоговый код:

**import** math  
print(**"Введите трехзначное число"**)  
olddig=int(input())  
newdig =(olddig%10)\*100+((olddig%100)/10)\*10+olddig/100  
print(**"Число, получившееся в результате преобразования "**,**"%.0f"** % newdig)

Результат работы: Введите трехзначное число

234

Число, получившееся в результате преобразования 432

На примере следующей задачи будут рассмотрены логические операции. Условие: точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит. 

Снова надо подключить библиотеку math.

**import** math

Вводимые координаты х и у будут действительными числами:

x=float(input())  
print(**"Введите у"**)  
y=float(input())

Далее следует использовать знания из области математики и составить логическое выражение для определения точки, находящейся в заданной фигуре:

isinarea=bool(((x+y>=3)**and**(pow(x,2)+pow(y,2)<=9)**and**(x>=0))**or**((-x+y<=-3)**and**(pow(x,2)+pow(y,2)<=9)**and**(x>=0)))  
  
Полученный результат выводится на экран:  
  
print(isinarea)  
  
Итоговый код:  
  
**import** math  
print(**"Введите х"**)  
x=float(input())  
print(**"Введите у"**)  
y=float(input())  
isinarea=bool(((x+y>=3)**and**(pow(x,2)+pow(y,2)<=9)**and**(x>=0))**or**((-x+y<=-3)**and**(pow(x,2)+pow(y,2)<=9)**and**(x>=0)))  
print(isinarea)

Результат работы: Введите х

0

Введите y

1

true

**ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ**

**Вариант 1**

Задача 1.Даны целое число n, действительные числа x, y. Вычислить:

A=;

B=;

C=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Даны целые числа m и n. Определить являются ли они делителями значения суммы первой и предпоследней цифры заданного целого четырехзначного числа.

**Вариант 2**

Задача 1.Даны действительные числа x, y, z. Вычислить:

A=;

B=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Дано действительное число a.Написать программу, печатающую число сотен в целой части значения выражения: 17\*a3/(a+1)2.

**Вариант 3**

Задача 1.Даны действительные числа а, b, целое число c. Вычислить:

X=;

Y=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Дано целое число k, действительное число x. Написать программу, определяющую количество десятков в округленном значении выражения: \*k2.

**Вариант 4**

Задача  1.Даны целые числа k, l, действительное число x. Вычислить:

A=;

B=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Написать программу вычисления суммы цифр заданного четырехзначного целого числа.

**Вариант 5**

Задача  1.Даны действительные числа a, b, целое число k. Вычислить:

X= ;

Y=;

Z=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Дано действительное число a.Написать программу, печатающую число сотен в целой части значения выражения: 17\*a3/(a+1)2.

**Вариант 6**

Задача 1.Даны целые числа i, k, действительное число a. Вычислить:

X=;

Y=.

Задача  2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Дано целое двузначное число k. Написать программу для вычисления R = , где m

- число, получаемое из заданного двузначного числа k выписыванием каждой цифры два раза. (Например: 23-2233, 80 - 8800).

**Вариант 7**

Задача  1.Даны целое число n, действительные числа x, y. Вычислить:

A=;

B=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Найти остаток от деления значения выражения c=k \* (a-b) на 5.

**Вариант 8**

Задача  1.Даны целые числа i, j, l. Вычислить:

A=;

B=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Дано целое число a и действительное число m.Написать программу для вычисления последней цифры в записи целой части выражения .

**Вариант 9**

Задача  1.Даны целое число kи действительное число a. Вычислить:

X=;

Y=;

Z=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит заштрихованной области и FALSE - если не принадлежит.



Задача 3.Дано натуральное число n.Вычислить сумму квадрата этого числа и квадратов цифр этого числа.

**Вариант 10**

Задача  1.Даны целые числа а, b, действительное число c. Вычислить:

X=;

Y=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Дано действительное число a.Написать программу для определения третьей цифры с конца в записи целой части выражения .

**Вариант 11**

Задача  1.Даны действительные числа s, t, r. Вычислить:

P=;

Q=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Дано целое число k, действительное число x.Вычислить произведение трех младших цифр в записи округленного значения выражения xk+k2.

**Вариант 12**

Задача  1.Даны целые числа n, k, действительное число a. Вычислить:

D=;

F=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Дано действительное число x.Написать программу для определения первой цифры дробной части в записи значения выражения .

**Вариант 13**

Задача 1.Даны целое k, действительные числа a, b. Вычислить:

;

.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая используя логическое выражение выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Найти разность первой цифры дробной части и второй цифры с конца целой части значения выражения .

**Вариант 14**

Задача  1.Даны целые числа a, b, c.Вычислить:

D=;

F=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Даны целые числа m,b.Написать программу для вычисления суммы цифр целой части значения выраженияm2 / b2 - .

**Вариант 15**

Задача  1.Даны целые числа n, k, действительные число a,b. Вычислить:

X=;

Y=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Даны действительные числа a, b. Найти сумму последней цифры целой части числа a и первой цифры дробной части числа b.

**Вариант 16**

Задача  1.Даны целое число k., действительные числа x, y, z. Вычислить:

A=;

B=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Даны два целых четырехзначных числа. Найти сумму последних двух цифр первого числа и произведение первых двух цифр второго числа.

**Вариант 17**

Задача 1.Даны целое число n, действительные числа i,j,k. Вычислить:

D=;

T=;

R=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит v области и FALSE - если не принадлежит.



Задача  3.Проверить кратно ли 6 третья цифра от конца в записи целого положительного четырехзначного числа.

**Вариант 18**

Задача  1.Даны целые числа n, k, действительные числа x,y. Вычислить:

A=;

B=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Присвоить целой переменной d первую цифру из дробной части положительного числа x.

**Вариант 19**

Задача  1.Даны действительные числа a, b, целые числа i, j. Вычислить:

C=;

D=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Найти остаток от деления значения выражения c=k(a+b) на 4.

**Вариант 20**

Задача  1.Даны действительные числа a, b, целое число k. Вычислить:

X= ;

Y=;

Z=.

Задача 2.Точка задана своими координатами (x, y). Написать программу, которая, используя логическое выражение, выводит TRUE, если точка принадлежит закрашенной области и FALSE - если не принадлежит.



Задача 3.Найти целую часть от деления на 7 целой части значения выражения .

КОНТРОЛЬНЫЕ ВОПРОСЫ И ЗАДАНИЯ

1. Расскажите об [особенностях](#Особенности) кода на Python.
2. Как можно расположить комментарии в программе?
3. Для чего нужны [спецификаторы](#Форматвывод)? Перечислите их.
4. Опишите консольный ввод Python. Приведите примеры ввода.
5. Расскажите [об операторах деления](#Математика). В чем их отличия?
6. Зачем нужны арифметические операции с присваиванием? Приведите примеры.
7. Для чего предназначена функция type()?
8. Как работает преобразование типов?
9. Напишите программу для арифметических операций чисел в разных системах счисления.
10. Запишите на языке Python законы де Моргана и составьте для них таблицы истинности.

ФОРМА ОТЧЕТА ПО ЛАБОРАТОРНОЙ РАБОТЕ

На выполнение лабораторной работы отводится 2 академических часа: 1 час на выполнение и сдачу лабораторной работы и 1 час на подготовку отчета. Отчет на защиту предоставляется в печатном виде.

Порядок выполнения:

1. Изучить теоретический материал.
2. Получить вариант у преподавателя.
3. Разработать программы согласно варианту.
4. Выполнить тестирование программы.
5. Продемонстрировать работу программы преподавателю.
6. Оформить отчет.
7. Защитить выполненную работу у преподавателя.

ОСНОВНАЯ ЛИТЕРАТУРА

1. Буйначев, С.К. Основы программирования на языке Python: учебное пособие / С.К. Буйначев, Н.Ю. Боклаг; Министерство образования и науки Российской Федерации, Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. - Екатеринбург: Издательство Уральского университета, 2014. - 92 с. – Режим доступа: <http://biblioclub.ru/index.php?page=book&id=275962>

2. Саммерфилд, М. Python на практике [Электронный ресурс] : учебное пособие / М. Саммерфилд ; пер. с англ. Слинкин А.А.. — Электрон. дан. — Москва: ДМК Пресс, 2014. — 338 с. — Режим доступа: <https://e.lanbook.com/book/66480>

ДОПОЛНИТЕЛЬНАЯ ЛИТЕРАТУРА

3. Сузи, Р.А. Язык программирования Python : курс / Р.А. Сузи. - 2-е изд., испр. - Москва : Интернет-Университет Информационных Технологий, 2007. - 327 с. - (Основы информационных технологий). – Режим доступа: <http://biblioclub.ru/index.php?page=book&id=233288>

4. Хахаев, И.А. Практикум по алгоритмизации и программированию на Python : курс / И.А. Хахаев. - 2-е изд., исправ. - Москва: Национальный Открытый Университет «ИНТУИТ», 2016. - 179 с. – Режим доступа: <http://biblioclub.ru/index.php?page=book&id=429256>

**Электронные ресурсы:**

5. Электронно-библиотечная система <http://biblioclub.ru/>

6. Электронно-библиотечная система <http://e.lanbook.com>

7. Сайт о программировании <https://metanit.com/>