**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение высшего образования**

**«Российский Университет Транспорта (МИИТ)»**

**Институт экономики и финансов**

**Кафедра «Информационные системы цифровой экономики»**

**О Т Ч Е Т**

**по дисциплине: «Программирование»**

**о выполнении практических работ**

**на тему «Программирование в Python»**

**Вариант 4**

**Выполнила: студент ЭПИ-111**

**Ган Екатерина**

**Проверила: ст. преп. Дмитриева Т.М.**

**Москва 2022 г.**

**Содержание**

Возможности модуля math……………………………………………………………………3-6

Списки………………………………………………………………………………………….6-7

Массивы как списки……………………………………………………………………..…...8-10

Матрицы………………………………………………………………………………………...10

Строки………………………………………………………………………………………..10-12

Словари……………………………………………………………………………………….…12

Черепашка…………………………………………………………………………………...12-13

Кортежи и множества…………………………………………………………….…………13-14

Файлы………………………………………………………………………………………..14-15

Функции и исключения………………………………………………………..……………15-16

Типовые алгоритмы…………………………...…………………………………………….17-20

**Возможности модуля math**

Пример 1

Постановка задачи:

Напишите программу для решения примера (по вариантам). Предусмотрите проверку деления на ноль. Все необходимые переменные пользователь вводит через консоль.

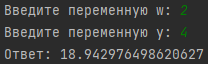
Изображение выглядит как текст

Автоматически созданное описание

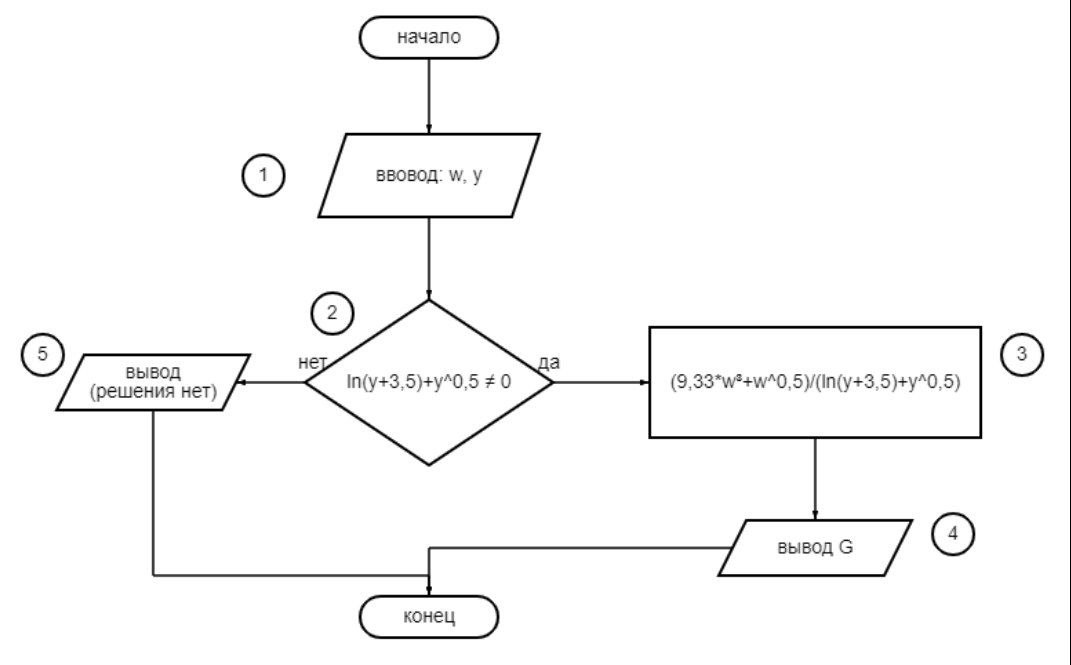
Текст программы:

import math  
  
w = float(input('Введите переменную w: '))  
y = float(input('Введите переменную y: '))  
a = (9.33\*(w\*\*3)) + math.sqrt(w)  
b = math.log(y+3.5)+math.sqrt(y)  
if b == 0:  
 print('Деление на 0 невозможно.')  
else:  
 G = a / b  
 print(f'Ответ: {G}')

Результаты выполнения:



Блок-схемы:



Пример 2

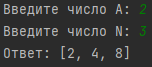
Постановка задачи:

Дано вещественное число A и целое число N (>0). Найдите все целые степени числа A от 1 до N.

Текст программы:

import math  
  
A = float(input('Введите число A: '))  
N = int(input('Введите число N: '))  
a = []  
  
for i in range(1, N+1):  
 a.append(int(math.pow(A, i)))  
  
print(f'Ответ: {a}')

Результаты выполнения:



Пример 3

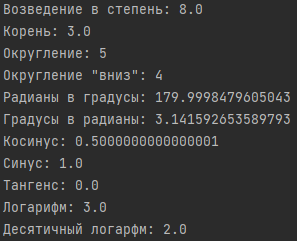
Постановка задачи:

К заданным числовым значениям применить функции по выполнению арифметических действий с помощью модуля math.

Текст программы:

import math  
  
print(f'Возведение в степень: {math.pow(2, 3)}')  
print(f'Корень: {math.sqrt(9)}')  
print(f'Округление: {math.ceil(4.56)}')  
print(f'Округление "вниз": {math.floor(4.56)}')  
print(f'Радианы в градусы: {math.degrees(3.14159)}')  
print(f'Градусы в радианы: {math.radians(180)}')  
print(f'Косинус: {math.cos(math.radians(60))}')  
print(f'Синус: {math.sin(math.radians(90))}')  
print(f'Тангенс: {math.tan(math.radians(0))}')  
print(f'Логарифм: {math.log(8, 2)}')  
print(f'Десятичный логарифм: {math.log10(100)}')

Результаты выполнения:



Пример 4

Постановка задачи:

Траектория снаряда, вылетающего из орудия под углом, а с начальной скоростью v0, задается уравнениями:

X = v0 \* t \* cos(a)

Y = v0 \* t \* sin(a) – (g \* t2)/2

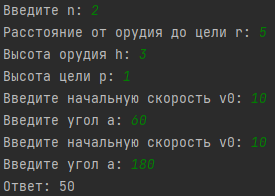
где g = 9,8 м/с2 — ускорение свободного падения, t — время.

Даны n пар значений и v0. Определить процент попадания снарядов в цель высотой Р, расположенную в вертикальной плоскости ствола орудия на расстоянии R на высоте H

Текст программы:

import math  
  
popal = 0  
n = int(input('Введите n: ')) #1  
r = int(input('Расстояние от орудия до цели r: ')) #5  
h = int(input('Высота орудия h: ')) #3  
p = int(input('Высота цели p: ')) #1  
g = 9.8  
  
for i in range(n):  
 v0 = int(input('Введите начальную скорость v0: ')) #10  
 a = int(input('Введите угол a: ')) #60  
 t = r / (v0 \* math.cos(math.radians(a))) # 1  
 x = v0 \* math.cos(math.radians(a)) \* t # 5  
 y = (v0 \* math.sin(math.radians(a)) \* t) - ((g \* t\*\*2) / 2) # 3,76  
  
 if (x >= r) and (y > h) and (y <= h+p):  
 popal += 1  
  
print(f'Ответ: {int((popal/n)\*100)}')

Результаты выполнения:



**Списки**

Пример 1

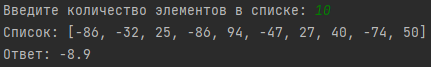
Постановка задачи:

Решить задачу поиска среднего значения в списке из N элементов. Использовать метод добавления элементов списка и суммирования элементов.

Текст программы:

import random  
  
a = []  
N = int(input('Введите количество элементов в списке: '))  
for i in range(N):  
 a.append(random.randint(-100, 100))  
  
print(f'Список: {a}')  
print(f'Ответ: {sum(a)/len(a)}')

Результаты выполнения:



Пример 2

Постановка задачи:

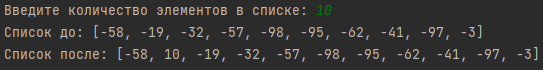
Создать линейный однонаправленный список из целых чисел. Вставить в список число 10

после первого элемента с отрицательным значением.

Текст программы:

import random  
  
a = []  
n = int(input('Введите количество элементов в списке: '))  
for i in range(n):  
 a.append(random.randint(-100, 9))  
print(f'Список до: {a}')  
  
for i in range(n):  
 if a[i] < 0:  
 a.insert(i+1, 10)  
 break  
print(f'Список после: {a}')

Результаты выполнения:



Пример 3

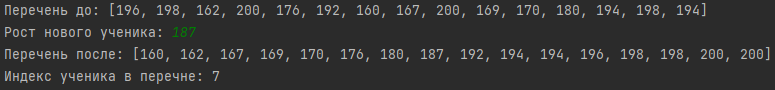
Постановка задачи:

Известны данные о росте 15 юношей класса, упорядоченные по убыванию. Нет ни одной пары учеников, имеющих одинаковый рост. В начале учебного года в класс поступил новый ученик. Какое место в перечне ростов займет рост этого ученика?

Текст программы:

import random  
a = []  
n = 15  
for i in range(n):  
 a.append(random.randint(150, 200))  
  
print(f'Перечень до: {a}')  
b = int(input('Рост нового ученика: '))  
a.append(b)  
a.sort()  
  
print(f'Перечень после: {a}')  
print(f'Индекс ученика в перечне: {a.index(b)}')

Результаты выполнения:



**Массивы как списки**

Пример 1

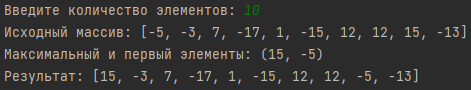
Постановка задачи:

Ввести целочисленный массив, состоящий из 10 элементов. Поменять местами максимальный и первый элементы.

Текст программы:

import random  
  
a = []  
n = int(input('Введите количество элементов: '))  
for i in range(n):  
 a.append(random.randint(-20, 20))  
  
print(f'Исходный массив: {a}')  
print(f'Максимальный и первый элементы: {max(a), a[0]}')  
  
i = a.index(max(a))  
a[0], a[i] = a[i], a[0]  
  
print(f'Результат: {a}')

Результаты выполнения:



Пример 2

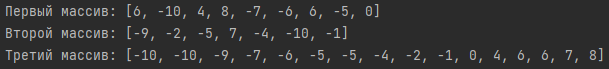
Постановка задачи:

Ввести два массива действительных чисел, состоящих из 9 и 7 элементов.Сформировать третий массив из упорядоченных по возрастанию значений обоих массивов.

Текст программы:

import random  
  
a = []  
n1 = 9  
for i in range(n1):  
 a.append(random.randint(-10, 10))  
print(f'Первый массив: {a}')  
  
b = []  
n2 = 7  
for i in range(n2):  
 b.append(random.randint(-10, 10))  
print(f'Второй массив: {b}')  
  
c = a+b  
c.sort()  
print(f'Третий массив: {c}')

Результаты выполнения:



Пример 3

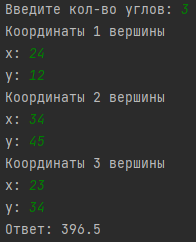
Постановка задачи:

Многоугольник на плоскости задан целочисленными координатами своих N вершин в декартовой системе координат. Требуется найти площадь многоугольника. Стороны многоугольника не соприкасаются (за исключением соседних – в вершинах) и не пересекаются.

Текст программы:

n = int(input("Введите кол-во углов: "))  
res = 0  
a = []  
b = []  
j = 0  
  
for i in range(n):  
 j += 1  
 print(f'Координаты {j} вершины')  
 a.append(int(input("x: ")))  
 b.append(int(input("y: ")))  
 s = (a[i-1]\*b[i]-b[i-1]\*a[i])/2  
 res = res + s  
 if res < 0:  
 res = res \* (-1)  
  
print(f'Ответ: {res}')

Результаты выполнения:



**Матрицы**

Пример 1

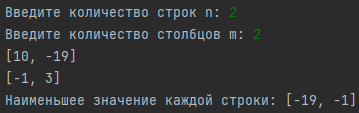
Постановка задачи:

Выполнить обработку элементов прямоугольной матрицы , имеющей  строк и  столбцов. Найти наименьшее значение среди значений для каждой строки матрицы.

Текст программы:

import random  
  
n = int(input('Введите количество строк n: '))  
m = int(input('Введите количество столбцов m: '))  
a = [[random.randint(-20, 20) for i in range(m)]for i in range(n)]  
  
for i in a:  
 print(i)  
  
print(f'Наименьшее значение каждой строки: {[min(r) for r in a]}')

Результаты выполнения:



**Строки**

Пример 1

Постановка задачи:

В заданной строке удалить первый символ.

Текст программы:

a = list(input())  
a.pop(0)  
print(''.join(a))

Результаты выполнения:



Пример 2

Постановка задачи:

В заданной строке заменить каждый символ «\*» числом, соответствующим номеру по

порядку вхождения этого символа в строку.

Текст программы:

a = list(input())  
for i in range(len(a)):  
 if a[i] == '\*':  
 a[i] = str(i)  
print(''.join(a))

Результаты выполнения:



Пример 3.

Постановка задачи:

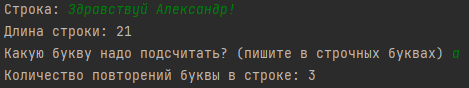
Задана строка символов. Вычислить количество символов в строке и подсчитать, сколько

раз в строке повторяется тот или иной символ.

Текст программы:

s = str(input('Строка: ').lower())  
print(f'Длина строки: {len(s)}')  
print(f"Количество повторений буквы в строке: {s.count(str(input('Какую букву надо подсчитать? (пишите в строчных буквах) ')))}")

Результаты выполнения:



**Словари**

Пример 1

Постановка задачи:

Дан текст: в первой строке задано число строк, далее идут сами строки. Выведите слово, которое в этом тексте встречается чаще всего. Если таких слов несколько, выведите то, которое меньше в лексикографическом порядке.

Текст программы:

counter = {}  
line = input('Введите строку: ').split()  
  
for word in line:  
 counter[word] = counter.get(word, 0) + 1  
  
max\_count = max(counter.values())  
most\_frequent = [k for k, v in counter.items() if v == max\_count]  
print(f'Самое повторяющиеся слово: {min(most\_frequent)}')

Результаты выполнения:



**Черепашка.**

Пример 1.

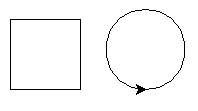
Постановка задачи:

Нарисовать произвольный квадрат и круг.

Текст программы:

import turtle  
  
turtle.penup()  
turtle.goto(35, 35)  
turtle.right(90)  
turtle.pendown()  
turtle.forward(70)  
turtle.right(90)  
turtle.forward(70)  
turtle.right(90)  
turtle.forward(70)  
turtle.right(90)  
turtle.forward(70)  
turtle.penup()  
turtle.goto(45, 35)  
turtle.goto(100, -35)  
turtle.pendown()  
turtle.circle(40)  
turtle.done()

Результаты выполнения:



**Кортежи и множества**

Пример 1

Постановка задачи:

На входе функция to\_set() получает строку или список чисел. Преобразуйте их в множество.

Текст программы:

def to\_set(element):  
 st = set(element)  
 return st, len(st)  
  
print(f'Ответ: {to\_set(input("Введите строку: "))}')

Результаты выполнения:



Пример 2

Постановка задачи:

Напишите функцию tpl\_sort(), которая сортирует кортеж, состоящий из целых чисел по возрастанию и возвращает его. Если хотя бы один элемент не является целым числом, то функция возвращает исходный кортеж.

Текст программы:

def tpl\_sort(tpl):  
 for element in tpl:  
 if not isinstance(element, int):  
 return tpl  
 return tuple(sorted(tpl))  
print(f'Ответ: {tpl\_sort((10, 9, 8, 7, 6, 5, 4, 3, 2, "y"))}')

def tpl\_sort(tpl):  
 for element in tpl:  
 if not isinstance(element, int):  
 return tpl  
 return tuple(sorted(tpl))  
print(f'Ответ: {tpl\_sort((10, 9, 8, 7, 6, 5, 4, 3, 2, 1))}')

Результаты выполнения:





Пример 3

Постановка задачи:

На вход функции sieve() поступает список целых чисел. В результате выполнения этой функции будет получен кортеж уникальных элементов списка в обратном порядке.

Текст программы:

def sieve(lst):  
 unique = []  
 a = [unique.append(item) for item in reversed(lst) if item not in unique]  
 return tuple(unique)  
  
print(f'Ответ: {sieve(list(map(int, input("Введите массив чисел: ").split())))}')

Результаты выполнения:



**Файлы**

Пример 1

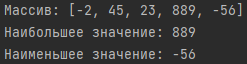
Постановка задачи:



Текст программы:

F = open("1234.txt")  
s = [int(i) for i in F.read().splitlines()]  
print(f'Массив: {s}\nНаибольшее значение: {max(s)}\nНаименьшее значение: {min(s)}')

Результаты выполнения:



Пример 2

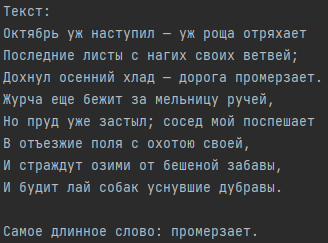
Постановка задачи:

Требуется реализовать функцию longest\_words(file), которая выводит слово, имеющее максимальную длину (или список слов, если таковых несколько).

Текст программы:

def longest\_words(file):  
 with open(file, encoding='utf-8') as text:  
 words = text.read().split()  
 max\_length = len(max(words, key=len))  
 sought\_words = [word for word in words if len(word) == max\_length]  
 if len(sought\_words) == 1:  
 return sought\_words[0]  
 return sought\_words  
  
f = open('1234.txt', encoding='utf-8')  
print(f'Текст:\n{f.read()}\n\nCамое длинное слово:', longest\_words('1234.txt'))

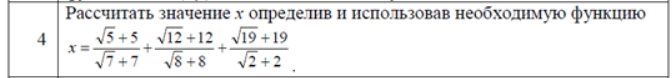
Результаты выполнения:



**Функции и исключения**

Пример 1

Постановка задачи:



Текст программы:

import math  
  
def x (a, b, c):  
 return (a + b + c)  
  
a = (math.sqrt(5) + 5) / (math.sqrt(7) + 7)  
b = (math.sqrt(12) + 12) / (math.sqrt(8) + 8)  
c = (math.sqrt(19) + 19) / (math.sqrt(2) + 2)  
  
print(f'Ответ: {x(a, b, c)}')

Результаты выполнения:

****

Пример 2

Постановка задачи:

«Обернуть» функцию в блок try… except…

Текст программы:

import math  
  
try:  
 b = 0  
 a = (math.sqrt(12) + 12) / b  
except ZeroDivisionError:  
 a = 0  
print(f'Ответ: {a}')

Результаты выполнения:



**Типовые алгоритмы**

Пример 1

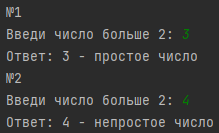
Постановка задачи:

Поиск простого числа

Текст программы:

print('№1')  
count = 0  
k = 2  
j = int(input('Введи число больше 2: '))  
  
while k\*k <= j:  
 if j % k == 0:  
 print(f'Ответ: {j} - непростое число')  
 k += 1  
if k\*k > j:  
 print(f'Ответ: {j} - простое число')  
  
print('№2')  
def prostoe(x):  
 s = int(x\*\*0.5)  
 for i in range(2, s+1):  
 if x % i == 0:  
 return f'Ответ: {x} - непростое число'  
 return f'Ответ: {x} - простое число'  
  
print(prostoe(int(input('Введи число больше 2: '))))

Результаты выполнения:



Пример 2

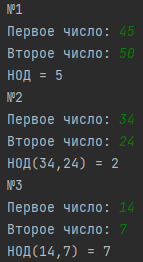
Постановка задачи:

Алгоритм Евклида (НОД)

Текст программы:

print('№1')  
q = int(input('Первое число: '))  
w = int(input('Второе число: '))  
  
while q != w:  
 if q > w:  
 q -= w  
 else:  
 w -= q  
  
print(f'НОД = {q}')  
  
print('№2')  
a = int(input('Первое число: '))  
b = int(input('Второе число: '))  
  
def NOD(a, b):  
 if a == 0 or b == 0:  
 return a+b  
 if a > b:  
 return NOD(a - b, b)  
 else:  
 return NOD(a, b - a)  
  
print(f'НОД({a},{b}) = {NOD(a, b)}')  
  
print('№3')  
a = int(input('Первое число: '))  
b = int(input('Второе число: '))  
  
def NOD(a, b):  
 if b == 0: return a  
 return NOD(b, a % b)  
  
print(f'НОД({a},{b}) = {NOD(a, b)}')

Результаты выполнения:



Пример 3

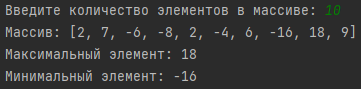
Постановка задачи:

Поиск максимального и минимального элемента в списке

Текст программы:

import random  
a = []  
n = int(input('Введите количество элементов в массиве: '))  
for i in range(n):  
 a.append(random.randint(-20, 20))  
print(f'Массив: {a}')  
  
maxi = a[0]  
for i in range(n):  
 if a[i] > maxi:  
 maxi = a[i]  
print(f'Максимальный элемент: {maxi}')  
  
mini = a[0]  
for i in range(n):  
 if mini > a[i]:  
 mini = a[i]  
print(f'Минимальный элемент: {mini}')

Результаты выполнения:

****

Пример 4

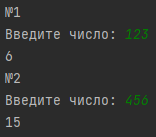
Постановка задачи:

Сумма цифр чисел

Текст программы:

print('№1')  
def sumDigits(e):  
 sumi = 0  
 while e != 0:  
 sumi += e % 10  
 e = e // 10  
 return sumi  
  
print(sumDigits(int(input('Введите число: '))))  
  
print('№2')  
def sumDigits(n):  
 if n < 10:  
 return n  
 d = n % 10  
 sumi = d + sumDigits(n // 10)  
 return sumi  
  
print(sumDigits(int(input('Введите число: '))))

Результаты выполнения:

****

Пример 5

Постановка задачи:

Фибиначи

Текст программы:

f1 = 1  
f2 = 1  
a = [1, 1]  
n = int(input('Введите n: '))  
for i in range(3, n):  
 f3 = f1 + f2  
 a.append(f3)  
 f1, f2, = f2, f3  
print(f'Ответ: {a}')

Результаты выполнения:

****