# **ФГБОУ ВО «Московский политехнический университет»**

## **Лабораторная работа №6**

Файлы

**Задание 1 Вариант №19**

По дисциплине:

Программирование

Выполнил

студент 1 курса

группы 201-321

Сметанина А.А.

Проверил:

\_\_\_\_\_\_\_\_\_\_\_\_ Никишина И.Н.

# **МОСКВА 2020**

**Цель работы**

Сформировать практический навык в написании программ, в которых выполняются

операции с текстовыми файлами – чтение, запись.

**Постановка задачи**

В предыдущих заданиях необходимые для программы данные, вводились с клавиатуры, а

результат выводился на экран монитора. Очевидно, что и при отладке программ, и при вводе большого объема данных в программу такой подход трудозатратен и потому непригоден. Наиболее подходящее решение – это ввод данных из файла и вывод результатов работы в файл. При этом входные данные подготавливаются однократно и с должным многообразием, а результаты работы можно анализировать многократно.

Напишем несколько программ, которые будут считывать входные данные из заранее

подготовленного текстового файла, и выводить результат в текстовый файл. Для этой цели воспользуемся программами, написанными в предыдущих заданиях.

**Теоретическая часть**

Файловые типы данных введены в языках программирования для работы с внешними

устройствами – файлами на диске, портами, принтерами и т.д. Файловые типы подразделяются на стандартные типы и типы, определяемые программистом. К стандартному типу относится текстовый тип файлов, с которым мы и будем работать.

Доступ к файлам может быть последовательным или прямым. При последовательном

доступе каждый следующий элемент может быть прочитан только после выполнения аналогичной операции с предыдущим элементом. При прямом доступе операция чтения (записи) может быть выполнена для произвольного элемента с заданным адресом.

**Описание программы**

Программа написана на алгоритмическом языке Python 3.6, реализована в среде macOS

Catalina и состоит из частей, отвечающих за ввод данных, вычисление и представление данных на экране монитора.

**Описание алгоритма**

Алгоритмы работают так же, как и в соответствующих лабораторных работах, однако

данные вводятся из файла и выводятся в файл.

**Описание входных и выходных данных**

Входные и выходные данные такие же, как и в соответствующих лабораторных

работах.

**Листинг программы (по з. №1)**

from math import sin, cos, pi

def f():

with open('in.txt', 'r', encoding='utf-8') as f:

a = float(f.read())

z1 = (((1 + a + a\*\*2) / (2 \* a + a\*\*2)) + 2 - ((1 - a + a\*\*2) / (2 \* a - a\*\*2))) \* (5 - 2 \* a\*\*2)

z2 = (4 - a\*\*2) / 2

with open('out.txt', 'w', encoding='utf-8') as f:

f.write(f"z1 = {z1:.4f}\n")

f.write(f"z2 = {z2:.4f}\n")

try:

f()

except FileNotFoundError:

with open('out.txt', 'w', encoding='utf-8') as f:

f.write("Файл со входными данными не найден\n")

except ZeroDivisionError:

with open('out.txt', 'w', encoding='utf-8') as f:

f.write("Введено неверное число, произошло деление на 0\n")

except Exception:

with open('out.txt', 'w', encoding='utf-8') as f:

f.write("Ожидалось вещественное число")

**Результат работы программы (файл out.txt по з. №1)**

z1 = -192.8571

z2 = -10.5000

**Листинг программы (по з. №4)**

import random

try:

with open('in.txt', 'r', encoding='utf-8') as f:

N = int(f.read())

assert 0 < N <= 30

except FileNotFoundError:

with open('out.txt', 'w', encoding='utf-8') as f:

f.write('Файл с входными данными не существует')

exit(-1)

except Exception:

with open('out.txt', 'w', encoding='utf-8') as f:

f.write("Ожидалось натуральное число")

exit(-1)

values = [random.uniform(-5, 5) for i in range(N)]

mult\_of\_negative = 1

for item in values:

if item < 0:

mult\_of\_negative \*= item

max\_ = max(values)

max\_index = values.index(max\_)

sum\_after\_max = 0

for item in values[:max\_index]:

if item > 0:

sum\_after\_max += item

reversed\_values = values[::-1]

with open('out.txt', 'w', encoding='utf-8') as f:

f.write(f"Длина массива: {N}\n")

f.write(f"Массив:\n{values}\n")

f.write(f"Произведение отрицательных элементов: {mult\_of\_negative}\n")

f.write(f"Сумма положительных до максимума: {sum\_after\_max}\n")

f.write(f"Перевёрнутый массив\n{reversed\_values}\n")

**Результат работы программы (файл out.txt для з. №4)**

Длина массива: 5

Массив:

[-4.577582678105396, 2.2101448348796238, 1.9210123663589362, -0.30374448322543746, -2.9242708054227395]

Произведение отрицательных элементов: -4.065951410143011

Сумма положительных до максимума: 0

Перевёрнутый массив

[-2.9242708054227395, -0.30374448322543746, 1.9210123663589362, 2.2101448348796238, -4.577582678105396]

**Листинг программы (по з. №5)**

import numpy as np

def min\_diagonal\_sum(matrix):

\_, size = matrix.shape

min\_d, offset = sum(matrix.diagonal()), 0

for i in range(-size + 2, size - 1):

if sum(matrix.diagonal(i)) < min\_d:

min\_d = sum(matrix.diagonal(i))

offset = i

return offset, min\_d

def summ(matrix):

res = 0

for row in matrix:

if all(i >= 0 for i in row):

res += sum(row)

return res

try:

with open('in.txt', 'r', encoding='utf-8') as f:

size = int(f.read())

assert size > 0

except FileNotFoundError:

with open('out.txt', 'w', encoding='utf-8') as f:

f.write("Файл с входными данными не существует\n")

exit(-1)

except Exception:

with open('out.txt', 'w', encoding='utf-8') as f:

f.write("Неверный ввод, ожидалось натуральное число\n")

exit(-1)

with open('out.txt', 'w', encoding='utf-8') as f:

matrix = np.random.randint(low=-100, high=100, size=(size, size))

f.write(f"Матрица {size}x{size}:\n")

f.write(f"{matrix}\n")

f.write(f"Сумма строк с элементами >= 0: {summ(matrix)}\n")

offset, min\_d = min\_diagonal\_sum(matrix)

f.write(f"Min сумма элементов диагоналей, || главной: {min\_d}\n")

f.write(f"{matrix.diagonal(offset)}\n")

**Результат работы программы (файл out.txt по з. №5)**

Матрица 5x5:

[[-33 26 -10 -93 38]

[-49 78 -86 11 -11]

[-80 54 -93 99 89]

[-23 -18 15 -84 54]

[ 76 -82 -5 6 -22]]

Сумма строк с элементами >= 0: 0

Минимальная сумма элементов диагоналей, параллельных главной: -154

[-33 78 -93 -84 -22]

**Список используемой литературы**

1. Лутц М. Программирование на Python, том I, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с.
2. Доусон М. Программируем на Python. – СПб.: Питер, 2014. – 416 с.