ФГАОУ ВО "МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ"

Лабораторная работа №6

Файлы

Вариант №11

по дисциплине:

Основы программирования

Выполнил

студент 1 курса
группы 211-321
Журавлев Д.А.

Проверил
_____ Никишина И.Н.

МОСКВА 2021

Постановка задачи

Выполнить корректировку программ, написанных для лабораторных работ 1,4 ,5, с таким условием, что бы ввод данных и вывод результатов осуществлялся с использованием файлов.

Теоретическая часть

Python содержит в себе функцию, под названием «open», которую можно использовать для открытия файлов для чтения.

Режимм «только чтение». Это стандартный режим функции открытия файлов. Обратите внимание на то, что мы не пропускаем весь путь к файлу, который мы собираемся открыть в первом примере. Python автоматически просмотрит папку, в которой запущен скрипт для text.txt. Если его не удается найти, вы получите уведомление об ошибке IOError. Это значит, что мы указываем Python, чтобы строка обрабатывалась как исходная. Давайте посмотрим на разницу между исходной строкой и обычной:

```
handle = open("test.txt")
handle = open(r"C:\Users\mike\py101book\data\test.txt", "r")
```

Существуют определенные специальные символы, которые должны быть отображены, такие как “n” или “t”. В нашем случае присутствует “t” (иными словами, вкладка), так что строка послушно добавляет вкладку в наш путь и портит её для нас. Второй аргумент во втором примере это буква “r”. Данное значение указывает на то, что мы хотим открыть файл в режиме «только чтение». Иными словами, происходит то же самое, что и в первом примере, но более явно. Теперь давайте, наконец, прочтем файл!

```
handle = open("test.txt", "r")
data = handle.readline() # read just one line
print(data)
handle.close()
```

После запуска, файл откроется и будет прочитан как строка в переменную. После этого мы печатаем данные и закрываем дескриптор файла. Следует всегда закрывать дескриптор файла, так как неизвестно, когда и какая именно программа захочет получить к нему доступ. Закрытие файла также поможет сохранить память и избежать появления странных багов в программе. Вы можете указать Python читать строку только раз, чтобы прочитать все строки в списке Python, или прочесть файл по частям. Последняя опция очень полезная, если вы работаете с большими фалами и вам не нужно читать все его содержимое, на что может потребоваться вся память компьютера.

```
with open('newfile.txt', 'w', encoding='utf-8') as g:

    d = int(input())

    print('1 / {} = {}'.format(d, 1 / d), file=g)
```

Описание программы

Для реализации поставленной задачи необходимо все вызовы функций `scanf` и `printf` заменить на чтение и запись в файл.

Описание алгоритма

1. Копируем необходимые файлы в папку с лабораторной работой.
2. Заменяем все вызовы функций `scanf` и `printf на чтение и запись в файл.`

Описание входных и выходных данных

Входные и выходные данные поступают из файлов в виде строк, далее приводятся к программой к необходимому виду.

Листинг программы

```
from math import *

def tg(a):
    return sin(a) / cos(a)

try:
    a = 0
    with open('res00', 'r') as f:
        a = float(f.readline())
    res1 = (cos(2 * a)) / (1 + sin(2 * a))
    res2 = (1 - tg(a)) / (1 + tg(a))
    with open('res00', 'w') as f:
        f.write("First  formula: {}\n".format(res1))
        f.write("Second formula: {}\n".format(res2))
except:
    print("Value error")
```

```
from math import *

try:
    x = 0
    with open('res01', 'r') as f:
        x = float(f.readline())

    if x <= -2:
        y = -x - 2
    if x > -2 and x < -1:
        y = sqrt(1 - pow(x + 1, 2))
    if x >= -1 and x <= 1:
        y = 1
    if x > 1 and x < 2:
        y = -2 * (x - 2) - 1
    if x >= 2:
        y = -1
    with open('res01', 'w') as f:
        f.write("X: {}  Y: {}".format(x, float(y)))
except:
    print("Value error")
```

```
from math import *

try:

    #r = float(input("Input R: "))
    r = 10

    x = 0
    y = 0
    with open('res02', 'r') as f:
        x = float(f.readline())
        y = float(f.readline())

    if x >= 0 and y >= 0 and sqrt(x*x + y*y) <= r:
        res = ("[{};{}] belongs to the region".format(x , y))
    elif x <= 0 and y <= 0 and -x - r <= y:
        res = ("[{};{}] belongs to the region".format(x , y))
    else:
        res = ("[{};{}] not belongs to the region".format(x , y))
    with open('res02', 'w') as f:
        f.write(res)
except:
    print("Value error")

from random import *
from math import *
import numpy as np

def gen_array(m: int, n: int):
    rng = np.random.default_rng(145)
    arr = np.array([(rng.integers(low=0, high=10, size=n)) for i in
range(m)])
    return arr

def swap_rows(arr, a, b):
    arr = list(arr)
    temp = arr[a]
    arr[a] = arr[b]
    arr[b] = temp
    arr = np.array(arr)
    return arr

def sum_rows(a ,b, weight):
    temp_a = [x for x in a]
    temp_b = [x * weight for x in b]
    a = [temp_a[i] + temp_b[i] for i in range(len(b))]
    return a

def go_triangle(arr_src: np.ndarray, n) -> np.ndarray:
    rng = np.random.default_rng(145)
    arr =  np.array([(rng.integers(low=0, high=10, size=n)) for i in
 range(n)])
    np.copyto(arr, arr_src)
```

```
    try:
        swaps = 0
        depth = 0
        for x in range(n - 1):
            main = arr[depth][depth]
            if (main == 0):
                toswap = depth + 1
                while (toswap < n):
                    if (arr[toswap][depth] != 0):
                        arr = swap_rows(arr, depth, toswap)
                        swaps += 1
                        break
                    toswap += 1
                main = arr[depth][depth]
            for i in range (depth + 1, n):
                devisor = -arr[i][depth] / main
                arr[i] = sum_rows(arr[i], arr[depth], devisor)
            depth += 1
        if arr[n-1][n-1] == 0:
            print("Данная матрица не может быть преведена к треуголь
ной")
            return None
        if swaps % 2 == 1:
            return arr * -1
        return arr
    except:
        print("Данная матрица не может быть преведена к треугольной"
)
        return None

def find_lines(arr, av):
    count = 0
    for line in arr:
        if sum(line) / len(line) < av:
            count += 1
    return count

try:
    n = 0
    av = 0
    with open('inp03', 'r') as f:
        n = int(f.readline())
        av = int(f.readline())
except:
    print("Value Error")
    exit()

arr = gen_array(n, n)
arr_tri = go_triangle(arr, n)

with open('res03', 'w') as f:
    f.write(np.array2string(arr))
    f.write("\n")
```

```
    f.write(np.array2string(arr_tri))
    f.write("\n")
    f.write("Lines with average less than {} - {}".format(av, find_l
ines(arr, av)))
```

Результат работы программы

Созданные файлы в файловой системе

Список используемой литературы

1. В.П. Рядченко, Методическое пособие по выполнению лабораторных работ
2. https://pythonworld.ru/