**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №2**

**по дисциплине** «Алгоритмы и Структуры Данных»

**Тема: Сортировки**

|  |  |  |
| --- | --- | --- |
| Студент гр. 1384 |  | Усачева Д.В. |
| Преподаватель |  | Иванов Д.В. |
|  |  |  |

Санкт-Петербург 2022

**Цель работы.**

Освоение алгоритма сортировки слиянием.

**Задание.**

Сортировка слиянием.

На вход программе подаются квадратные матрицы чисел. Напишите программу, которая сортирует матрицы по возрастанию суммы чисел на главной диагонали**с использованием алгоритма сортировки слиянием**.

Формат входа.

Первая строка содержит натуральное число n - количество матриц. Далее на вход подаются n матриц, каждая из которых описана в формате: сначала отдельной строкой число mi - размерность i-й по счету матрицы. После m строк по m чисел в каждой строке - значения элементов матрицы.

Формат выхода.

* Порядковые номера тех матриц, которые участвуют в слиянии на очередной итерации алгоритма. Вывод с новой строки для каждой итерации.
* Массив, в котором содержатся порядковые номера матриц, отсортированных по возрастанию суммы элементов на диагонали. Порядковый номер матрицы - это её номер по счету, в котором она была подана на вход программе, нумерация начинается с нуля.

**Выполнение работы.**

Для выполнения задания была реализована функция merge, в которой происходит сортировка слиянием, так же в этой функции предусмотрен вывод порядковых номеров тех матриц, которые участвуют в слиянии на очередной итерации алгоритма.

В основном теле программы мы получаем необходимые сведения о введённых матрицах и записываем ее в массив, в котором будет выполнена непосредственнно сортировка слиянием при помощи функции merge. Итоговый результат также выводится в основном теле функции.

Разработанный программный код см. в приложении 1.

**Тестирование.**

Результаты тестирования представлены в таблице 1.

Для проверки работы программы был разработан код тестовой

программы.

Всего 3 теста:

- test\_1. Данный тест был взят с условия лабораторной с сайта.

Вход: 3 различные матрицы.

- test\_2. Тест для проверки работы программы без ввода данных.

- test\_3. Тест для проверки особого случая. Вход: 3 матрицы с

одинаковой суммой на главной диагонали.

- test\_4. На вход подаётся несколько идентичных матриц и одной не схожей

с ними;

- test\_5. На вход подаются матрицы, след которых убывает;

- test\_6. На вход подаются матрицы с возрастающим следом, за исключением

матрицы в середине ввода – у неё наибольший среди всех след.

**Выводы.**

В ходе выполнения был освоен алгоритм сортировки слиянием, были написаны тесты, проверяющие различные случаи задачи. Был визуализирован ход выполнения сортировки – для каждого вызова функции сортировки выводится положение объектов в массиве.

**ПРИЛОЖЕНИЕ 1**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

Название файла: main.py

#python

def sum\_ii(matrix):

work = 0

for i in range(len(matrix)):

work += matrix[i][i]

return work

def merge(arr):

if len(arr) == 1:

return

middle = len(arr) // 2

left, right = arr[:middle], arr[middle:]

merge(left)

merge(right)

index\_left = index\_right = index = 0

result = [0] \* (len(left) + len(right))

while index\_left < len(left) and index\_right < len(right):

if left[index\_left][0] <= right[index\_right][0]:

result[index] = left[index\_left]

index\_left += 1

else:

result[index] = right[index\_right]

index\_right += 1

index += 1

while index\_left < len(left):

result[index] = left[index\_left]

index\_left += 1

index += 1

while index\_right < len(right):

result[index] = right[index\_right]

index\_right += 1

index += 1

res = [str(k[1]) for k in result]

print(" ".join(res))

for i in range(len(arr)):

arr[i] = result[i]

return arr

matrix\_res = []

count = int(input())

for i in range(count):

matrix = []

mi = int(input())

for j in range(mi):

line = list(map(int, input().split()))

matrix.append(line)

matrix\_res.append((sum\_ii(matrix), i))

result = merge(matrix\_res)

res = [str(k[1]) for k in result]

print(" ".join(res))

Название файла: tests.py

import unittest

from main import \*

class TestMethods(unittest.TestCase):

def test\_1(self):

arr1 = [(32, 0), (11, 1), (3, 2)]

self.assertEqual(merge(arr1),[(32, 0), (11, 1), (3, 2)])

def test\_2(self):

arr2 = []

self.assertEqual(merge(arr2), None)

def test\_3(self):

arr3 = [(10, 0), (10, 1), (10, 2)]

self.assertEqual(merge(arr3), [(10, 0), (10, 1), (10, 2)])

def test\_4(self):

arr4 = [(10, 0), (10, 1), (10, 2), (11, 3)]

self.assertEqual(merge(arr4), [(10, 0), (10, 1), (10, 2), (11, 3)])

def test\_5(self):

arr5 = [(10, 0), (19, 1), (8, 2)]

self.assertEqual(merge(arr5), [(10, 0), (9, 1), (8, 2)])

def test\_6(self):

arr6 = [(10, 0), (11, 1), (14, 2), (12, 3), (13, 4)]

self.assertEqual(merge(arr6), [(10, 0), (11, 1), (14, 2), (12, 3), (13, 4)])

if \_\_name\_\_ == "\_\_main\_\_":

unittest.main()