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. 1303 _____ Беззубов Д.В.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург
2022

Цель работы.

Изучение различных алгоритмов сортировки. Реализация программы, использующей сортировку слиянием.

Задание.

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 m_i - 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 реализации данной задачи был реализован класс *Matrix*. Полями матрицы являются *_id_* (уникальный номер матрицы, который увеличивается при создании нового объекта данного класса) и поле *_data_*, в котором хранится сама матрица. В данном классе реализованы следующие методы:

- *getID()* – возвращает id данной матрицы
- *get_dSum()* – возвращает сумму чисел на главной диагонали матрицы

Так же были перегружены оператор сравнения «<» и метод *str()*.

Кроме того, была реализована функция, выполняющая сортировку слиянием. Список делится пополам, затем, для каждой половины так же вызывается функция *merge_sort()*. После этого полученные списки начинают «сливаться». В результате функция возвращает отсортированный список матриц.

Инициализация матриц и вызов функции сортировки производится в функции *main()*.

Исходный код программы изложен в Приложении А, код тестов – в Приложении Б.

Тестирование программы.

Тестирование функции сортировки производится с помощью unit-тестов, описанных в файле *test.py*. Данные тесты покрывают следующие случаи:

- На вход подается пустая матрица
- На вход подается 1 матрица размерности 1
- На вход подаются 3 «обычные» матрицы с разными суммами чисел на главной диагонали

Вывод.

В ходе выполнения данной лабораторной работы был освоен алгоритм сортировки слиянием. Реализована программа, выполняющая сортировку матриц по увеличению суммы чисел на главной диагонали. Ход сортировки визуализируется посредством вывода каждого «слияния». Написанная программа покрыта юнит-тестами.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
import itertools as it

class Matrix:
    _data_ = []
    _id_ = it.count()

    def __init__(self, data):
        self._data_ = data
        self._id_ = next(Matrix._id_)

    def get_dSum(self):
        res = 0
        for i in range(len(self._data_)):
            res+=self._data_[i][i]
        return res

    def getID(self):
        return self._id_

    def __str__(self):
        return f'{self.getID()}'

    def __lt__(self, other):
        return self.get_dSum() < other.get_dSum()


def merge_sort(x):

    if len(x) < 2: return x

    result,mid = [],int(len(x)/2)

    y = merge_sort(x[:mid])
    z = merge_sort(x[mid:])

    while (len(y) > 0) and (len(z) > 0):
            if y[0] > z[0]: result.append(z.pop(0))
```

```
            else: result.append(y.pop(0))

    result.extend(y+z)
    print(*result)
    return result


def main():
    count = int(input())
    matrix_array = []
    for i in range(count):
        n = int(input())
        matrix_array.append(Matrix([list(map(int,
input().split())) for elem in range(n)]))
    print(*merge_sort(matrix_array))

if __name__ == '__main__':
    main()
```

ПРИЛОЖЕНИЕ Б

ИСХОДНЫЙ КОД UNIT-ТЕСТОВ

Название файла: test.py

```
from main import *
import pytest

def test_one_matrix():
    res = merge_sort([Matrix([1])])
    res = ' '.join(str(elem) for elem in res)
    assert res == '0'

def test_empty_matrix():
    res = merge_sort([Matrix([])])
    res = ' '.join(str(elem) for elem in res)
    assert res == '1'

def test_normal():
    a = Matrix([[1,1,1],[1,1,1],[1,1,1]])
    b = Matrix([[0, 1, 1], [1, 0, 1], [1, 1, 0]])
    c = Matrix([[2, 1, 1], [1, 2, 1], [1, 1, 2]])
    res = merge_sort([a,b,c])
    res = ' '.join(str(elem) for elem in res)
    assert res == '3 2 4'
```