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: Сортировка слиянием

Студент гр. 1304 _____ Поршнев Р.А.

Преподаватель _____ Глазунов С.А.

Санкт-Петербург

2022

Цель работы.

Написать программу, которая сортирует матрицы по возрастанию суммы чисел на главной диагонали с использованием алгоритма сортировки 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 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.

В Main происходит считывание количества квадратных матриц. Затем для каждой матрицы считывается её размер. Далее происходит построчное считывание текущей матрицы. Каждая строка матрицы обрабатывается следующим образом:

1) Разбивается по пробелу;

2) Каждый элемент, получившийся в результате разбиения, приводится к типу int с помощью функции map;
3) Полученный объект преобразуется в список и записывается в переменную row.

Каждая преобразованная строка матрицы row добавляется в текущую матрицу.

В список data добавляется индекс текущей матрицы и сумма элементов по диагонали. Затем в список matrix_data добавляется список с данными матрицы data.

После завершения цикла переменной numb_matrix, в которой будет хранится ответ, присваивается результат работы функции ans.

Затем происходит вывод номеров матриц с помощью цикла for.

Функция ans принимает список matrix_data, в котором содержатся данные о каждой матрице.

Внутри данной функции matrix_data присваивается результат функции merge_sort. Далее в ans записывается окончательный порядок матриц. После записи ответа в ans у данной строки отсекается начальный пробел, которой появился из-за особенности реализации записи ответа. Затем в numb_matrix записывается итоговое расположение матриц и возвращается список, в котором содержится ответ на изначальную задачу.

В функции merge_sort реализована модернизированная сортировка слиянием. В качестве аргументов она принимает список с данными о каждой матрице и список, в котором будет хранится ответ на задачу. Функция возвращает отсортированный список с данными.

Модернизация основана на том, что в ходе сортировки в новый массив записываются не только сравниваемые значения сумм элементов матриц по диагонали, но и соответствующие матрицам индексы.

Суть сортировки заключается в том, что функция рекурсивно делит список на 2 до тех пор, пока его длина не станет равна 1, а затем скрепляет 2 части списка по одному элементу, сравнивая элементы списков между собой.

Функция sum принимает матрицу. С помощью цикла for она считает сумму элементов по диагонали. По завершению цикла функция возвращает сумму.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	3 2 1 2 1 31 3 1 1 1 1 11 1 1 1 -1 5 1 2 0 1 -1 1 2 0 1 -1 1 2 0 1 -1 1 2 0 1 -1 1 2 0 1 -1	2 1 2 1 0 2 1 0	Ответ правильный, проверяется случай при убывании сумм элементов матриц по диагонали
2.	6 2 0 0 0 50 3 5 1 2 -10 15 9 0 0 0 2 2 0 -10 -1 1 -1000 2 -2000 10 0 999 1 1111	2 1 2 1 0 4 5 4 3 5 4 3 2 1 0 5 4 3 2 1 0 5	Ответ правильный, проверяется случай при неотсортированном массиве

3.	5 2 -1 1 -1 1 1 0 3 0 0 0 0 0 0 0 0 0 2 5 0 0 -5 1 0	0 1 3 4 2 3 4 0 1 2 3 4 0 1 2 3 4	Ответ правильный, проверяется случай равенства сумм элементов матриц по диагонали
4.	1 1 5	0	Ответ правильный, проверяется случай сортировки одного элемента
5.	6 2 -1 2 10 2 3 1 2 3 4 5 6 0 0 -4 2 0 0 10 5 3 1 -100 2 0 0 10 5 5 5 2 10 0 -1 -3 2 -1 0 0 10	1 2 0 1 2 4 5 3 4 5 0 1 2 3 4 5 0 1 2 3 4 5	Ответ правильный, проверяется случай отсортированного массива

Вывод.

В ходе выполнения лабораторной работы была изучена сортировка слиянием.

Разработана программа, сортирующая матрицы по возрастанию суммы элементов на главной диагонали и выводящая порядковые номера тех матриц, которые участвуют в слиянии на очередной итерации алгоритма.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
def merge_sort(arr, numb_matrix):
    if len(arr) == 1 or len(arr) == 0:
        return arr
    L = merge_sort(arr[:len(arr) // 2], numb_matrix)
    R = merge_sort(arr[len(arr) // 2:], numb_matrix)
    n = m = k = 0
    C = [[0, 0]] * (len(L) + len(R))
    while n < len(L) and m < len(R):
        data = []
        if L[n][1] <= R[m][1]:
            data.append(L[n][0])
            data.append(L[n][1])
            C[k] = data
            n += 1
        else:
            data.append(R[m][0])
            data.append(R[m][1])
            C[k] = data
            m += 1
        k += 1
    while n < len(L):
        data = []
        data.append(L[n][0])
        data.append(L[n][1])
        C[k] = data
        n += 1
        k += 1
    while m < len(R):
        data = []
        data.append(R[m][0])
        data.append(R[m][1])
        C[k] = data
        m += 1
        k += 1
    ans = ''
    for i in range(len(arr)):
        ans += ' ' + str(C[i][0])
        arr[i] = C[i]
    ans = ans[1:]
    numb_matrix.append(ans)
    return arr
def sum(matrix):
    sum = 0
    for i in range(len(matrix)):
        sum += matrix[i][i]
    return sum
def ans(matrix_data):
    numb_matrix = []
    matrix_data = merge_sort(matrix_data, numb_matrix)
    ans = ''
    for i in range(len(matrix_data)):
        ans += ' ' + str(matrix_data[i][0])
```

```
        ans = ans[1:]
        numb_matrix.append(ans)
        return numb_matrix
    if __name__ == '__main__':
        n = int(input())
        matrix_data = []
        for i in range(n):
            matrix = []
            size_matrix = int(input())
            for j in range(size_matrix):
                row = list(map(int, input().split()))
                matrix.append(row)
            data = []
            data.append(i)
            data.append(sum(matrix))
            matrix_data.append(data)
        numb_matrix = ans(matrix_data)
        for i in range(len(numb_matrix)):
      print(numb_matrix[i])
```

Название файла: tests.py

```
    from main import merge_sort

    def test_1():
        matrix_data = [[0, 32], [1, 11], [2, 3]]
        numb_matrix = []
        merge_sort(matrix_data, numb_matrix)
        ans = ''
        for i in range(len(matrix_data)):
            ans += ' ' + str(matrix_data[i][0])
        ans = ans[1:]
        numb_matrix.append(ans)
        assert numb_matrix == ['2 1', '2 1 0', '2 1 0']

    def test_2():
        matrix_data = [[0, 100], [1, 20], [2, 1], [3, -1000], [4, -
1001], [5, 1111]]
        numb_matrix = []
        merge_sort(matrix_data, numb_matrix)
        ans = ''
        for i in range(len(matrix_data)):
            ans += ' ' + str(matrix_data[i][0])
        ans = ans[1:]
        numb_matrix.append(ans)
        assert numb_matrix == ['2 1', '2 1 0', '4 5', '4 3 5', '4 3 2
1 0 5', '4 3 2 1 0 5']

    def test_3():
        matrix_data = [[0, 0], [1, 0], [2, 0], [3, 0], [4, 0]]
        numb_matrix = []
        merge_sort(matrix_data, numb_matrix)
        ans = ''
        for i in range(len(matrix_data)):
            ans += ' ' + str(matrix_data[i][0])
        ans = ans[1:]
        numb_matrix.append(ans)
```

```
        assert numb_matrix == ['0 1', '3 4', '2 3 4', '0 1 2 3 4', '0
1 2 3 4']

    def test_4():
        matrix_data = [[0, 1]]
        numb_matrix = []
        merge_sort(matrix_data, numb_matrix)
        ans = ''
        for i in range(len(matrix_data)):
            ans += ' ' + str(matrix_data[i][0])
        ans = ans[1:]
        numb_matrix.append(ans)
        assert numb_matrix == ['0']

    def test_5():
        matrix_data = [[0, 10], [1, 11], [2, 15], [3, 25], [4, 50], [5,
100]]
        numb_matrix = []
        merge_sort(matrix_data, numb_matrix)
        ans = ''
        for i in range(len(matrix_data)):
            ans += ' ' + str(matrix_data[i][0])
        ans = ans[1:]
        numb_matrix.append(ans)
    assert numb_matrix == ['1 2', '0 1 2', '4 5', '3 4 5', '0 1 2 3 4
5', '0 1 2 3 4 5']
```