Санкт-Петербургский Национальный Исследовательский
Университет Информационных Технологий, Механики и Оптики
Мегафакультет компьютерных технологий и управления

Дисциплина
«Алгоритмы и структуры данных»
Лабораторная работа №1

Выполнил:
Студент группы P3218
Рябов Сергей Витальевич
Преподаватель:
Муромцев Дмитрий Ильич

Санкт-Петербург,
2018

1. Задача «a+b»

В данной задаче требуется вычислить сумму двух заданных чисел.

Исходный код (python):

```
file_input = open("input.txt", "r")
file_output = open("output.txt", "w")
a, b = file_input.read().split()
file_output.write(str(int(a) + int(b)))
```

Результат:

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.109	17752064	25	11
1	OK	0.078	17702912	7	2
2	OK	0.062	17739776	8	3
3	OK	0.109	17747968	5	1
4	OK	0.062	17739776	5	1
5	OK	0.062	17743872	6	1
6	OK	0.062	17678336	9	4
7	OK	0.078	17747968	23	10
8	OK	0.062	17739776	25	11
9	OK	0.062	17702912	24	1
10	OK	0.078	17747968	24	1
11	OK	0.078	17743872	14	10
12	OK	0.078	17698816	23	10
13	OK	0.078	17752064	23	11
14	OK	0.078	17752064	20	9
15	OK	0.062	17711104	23	11
16	OK	0.062	17731584	20	9
17	OK	0.062	17690624	22	10
18	OK	0.078	17739776	23	11
19	OK	0.062	17707008	22	10
20	OK	0.062	17739776	22	10
21	OK	0.078	17739776	22	10

2. Задача «a+b^2»

В данной задаче требуется вычислить значение выражения $a + b^2$.

Исходный код (python):

```
file_input = open("input.txt", "r")
file_output = open("output.txt", "w")
a, b = file_input.read().split()
file_output.write(str(int(a) + int(b) ** 2))
```

Результат:

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.093	17825792	25	19
1	OK	0.093	17764352	7	3
2	OK	0.062	17698816	8	3
3	OK	0.062	17756160	5	1
4	OK	0.062	17780736	5	1
5	OK	0.062	17723392	6	1
6	OK	0.062	17776640	6	1
7	OK	0.078	17813504	23	19
8	OK	0.062	17747968	25	18
9	OK	0.062	17805312	24	18
10	OK	0.062	17825792	24	19
11	OK	0.093	17784832	23	18
12	OK	0.062	17756160	23	18
13	OK	0.078	17801216	20	15
14	OK	0.062	17793024	23	18
15	OK	0.078	17743872	20	18
16	OK	0.078	17817600	22	18
17	OK	0.062	17813504	23	18
18	OK	0.062	17756160	22	17
19	OK	0.062	17825792	22	17
20	OK	0.078	17784832	22	18

3. Сортировка вставками

Дан массив целых чисел. Ваша задача — отсортировать его в порядке неубывания с помощью сортировки вставками.

Сортировка вставками проходится по всем элементам массива от меньших индексов к большим («слева направо») для каждого элемента определяет его место в предшествующей ему отсортированной части массива и переносит его на это место (возможно, сдвигая некоторые элементы на один индекс вправо). Чтобы проконтролировать, что Вы используете именно сортировку вставками, мы попросим Вас для каждого элемента массива, после того как он будет обработан, выводить его новый индекс.

Формат входного файла

В первой строке входного файла содержится число n ($1 \le n \le 1000$) — число элементов в массиве. Во второй строке находятся n различных целых чисел, по модулю не превосходящих 10^9.

Формат выходного файла

В первой строке выходного файла выведите n чисел. При этом i-ое число равно индексу, на который, в момент обработки его сортировкой вставками, был перемещен i-ый элемент исходного массива. Индексы нумеруются, начиная с единицы. Между любыми двумя числами должен стоять ровно один пробел.

Во второй строке выходного файла выведите отсортированный массив. Между любыми двумя числами должен стоять ровно один пробел.

Исходный код (python):

```
file_input = open("input.txt", "r")
file_output = open("output.txt", "w")

size = file_input.readline()
array = file_input.read().split()
result = [int(array[0])]
result_indicies = [1]

for i in range(1, int(size)):
    for j in range(len(result)):
        if int(array[i]) < result[j]:
            result.insert(j, int(array[i]))
            result_indicies.append(j + 1)
            break
    else:
        result.append(int(array[i]))
        result_indicies.append(len(result))

for r in result_indicies:
    file_output.write(f"{r} ")
file_output.write('\n')
for r in result:
    file_output.write(f"{r} ")
```

Результат:

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.203	19685376	10415	14298
1	OK	0.062	17883136	25	42
2	OK	0.078	17833984	7	7
3	OK	0.078	17879040	12	14
4	OK	0.062	17846272	8	10
5	OK	0.062	17874944	10	14
6	OK	0.062	17874944	29	33
7	OK	0.078	17817600	10	14
8	OK	0.062	17903616	10	14
9	OK	0.062	17858560	10	14
10	OK	0.062	17842176	10	14

4. Знакомство с жителями Сортлэнда

Владелец графства Сортлэнд, граф Бабблсортер, решил познакомиться со своими подданными. Число жителей в графстве нечетно и составляет n, где n может быть достаточно велико, поэтому граф решил ограничиться знакомством с тремя представителями народонаселения: с самым бедным жителем, с жителем, обладающим средним достатком, и с самым богатым жителем.

Согласно традициям Сортлэнда, считается, что житель обладает средним достатком, если при сортировке жителей по сумме денежных сбережений он оказывается ровно посередине. Известно, что каждый житель графства имеет уникальный идентификационный номер, значение которого расположено в границах от единицы до n. Информация о размере денежных накоплений жителей хранится в массиве M таким образом, что сумма денежных накоплений жителя, обладающего идентификационным номером i, содержится в ячейке M[i]. Помогите секретарю графа мистеру

Свопу вычислить идентификационные номера жителей, которые будут приглашены на встречу с графом.

Формат входного файла

Первая строка входного файла содержит число жителей n (3≤n≤9999, n нечетно). Вторая строка содержит описание массива M, состоящее из n положительных вещественных чисел, разделенных пробелами. Гарантируется, что все элементы массива M различны, а их значения имеют точность не более двух знаков после запятой и не превышают 106.

Формат выходного файла

В выходной файл выведите три целых положительных числа, разделенных пробелами — идентификационные номера беднейшего, среднего и самого богатого жителей Сортлэнда.

Исходный код (python):

```
file_input = open("input.txt", "r")
file_output = open("output.txt", "w")

size = file_input.readline()
array = file_input.read().split()

for ind in range(len(array)):
    array[ind] = [float(array[ind]), ind + 1]
result = [array[0]]

for i in range(1, int(size)):
    for j in range(len(result)):
        if array[i][0] < result[j][0]:
            result.insert(j, array[i])
            break
    else:
        result.append(array[i])
file_output.write(f"{result[0][1]} {result[int(size) // 2][1]} {result[int(size) -
1][1]}")
```

Результат:

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		1.156	21483520	98892	14
1	OK	0.062	17879040	30	5
2	OK	0.093	17944576	33	5
3	OK	0.093	18538496	1065	8
4	OK	0.078	19124224	3732	10
5	OK	0.109	19648512	14975	13
6	OK	0.109	19677184	14998	11
7	OK	0.156	20107264	28749	14
8	OK	0.187	20168704	34791	12
9	OK	0.218	20029440	38037	13
10	OK	0.218	19996672	38074	14

5. Секретарь Своп

Уже знакомый нам из предыдущей задачи граф Бабблсортер поручил своему секретарю, мистеру Свопу, оформлять приглашения беднейшему, богатейшему и среднему по достатку жителю своих владений. Однако кто же, в отсутствие мистера Свопа, будет заниматься самым важным делом — сортировкой массивов чисел? Видимо, это придется сделать Вам!

Дан массив, состоящий из n целых чисел. Вам необходимо его отсортировать по неубыванию. Но делать это нужно так же, как это делает мистер Своп — то есть, каждое действие должно быть взаимной перестановкой пары элементов. Вам также придется записать все, что Вы делали, в файл, чтобы мистер Своп смог проверить Вашу работу.

Формат входного файла

В первой строке входного файла содержится число n ($1 \le n \le 5000$) — число элементов в массиве. Во второй строке находятся n целых чисел, по модулю не превосходящих 109. Числа могут совпадать друг с другом.

Формат выходного файла

В первых нескольких строках выведите осуществленные Вами операции перестановки элементов. Каждая строка должна иметь следующий формат:

Swap elements at indices X and Y.

где X и Y — различные индексы массива, элементы на которых нужно переставить ($1 \le X,Y \le n$). Мистер Своп любит порядок, поэтому сделайте так, чтобы X<Y.

После того, как все нужные перестановки выведены, выведите следующую фразу:

No more swaps needed.

Во последней строке выходного файла выведите отсортированный массив, чтобы мистер Своп не переделывал работу за Вас. Между любыми двумя числами должен стоять ровно один пробел.

Исходный код (python):

```
file_input = open("input.txt", "r")
file_output = open("output.txt", "w")

size = file_input.readline()
array = [int(x) for x in file_input.read().split()]

for i in range(len(array)):
    min_element_index = i
    for j in range(i + 1, len(array)):
        if array[min_element_index] > array[j]:
            min_element_index = j
    if min_element_index != i:
        array[i], array[min_element_index] = array[min_element_index], array[i]
        file_output.write(f"Swap elements at indices {i + 1} and {min_element_index +
1}.\n")

file_output.write("No more swaps needed.\n")
for element in array:
    file_output.write(f"{element} ")
```

Результат:

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.281	21393408	51993	255416
1	OK	0.078	17948672	14	138
2	OK	0.078	17944576	7	26
3	OK	0.062	17899520	12	31
4	OK	0.062	17911808	8	62
5	OK	0.062	17981440	10	29
6	OK	0.062	17948672	10	29
7	OK	0.062	17891328	29	48
8	OK	0.062	17948672	10	64
9	OK	0.078	17960960	10	64
10	OK	0.078	17911808	10	99