Министерство образования и науки Российской Федерации
САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Факультет программной инженерии и компьютерной техники
Направление подготовки 09.03.04 Программная инженерия
Дисциплина «**Алгоритмы и структуры данных**»

ОТЧЁТ
по лабораторным работам
Неделя 1

Студент **Сараев В.В. группы P3218**
Преподаватель**:**
Романов Алексей Андреевич
Волчек Дмитрий Геннадьевич

Санкт-Петербург
2018 г.

Заданния:

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Задача «a+b»

В данной задаче требуется вычислить сумму двух заданных чисел.

Формат входного файла

Входной файл состоит из одной строки, которая содержит два целых числа a и b. Для этих чисел выполняются условия $-10^9 \le a \le 10^9$, $-10^9 \le b \le 10^9$.

Формат выходного файла

В выходной файл выведите единственное целое число — результат сложения $a + b$.

Листинг

Файл Lab1_1.py

```
#открываем файл для чтения
input_file = open('input.txt', 'r')
#считываем строку
params = input_file.read().split(" ")
#суммируем
sum = int(params[0]) + int(params[1])
output_file = open('output.txt','w')
#записываем файл
output_file.write(str(sum))
```

Бенчмарк:

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.078	17764352	25	11
1	OK	0.078	17764352	7	2
2	OK	0.078	17735680	8	3
3	OK	0.062	17711104	5	1
4	OK	0.078	17719296	5	1
5	OK	0.062	17743872	6	1
6	OK	0.062	17739776	9	4
7	OK	0.078	17702912	23	10
8	OK	0.062	17752064	25	11
9	OK	0.062	17752064	24	1
10	OK	0.062	17694720	24	1
11	OK	0.078	17739776	14	10
12	OK	0.062	17739776	23	10
13	OK	0.078	17694720	23	11
14	OK	0.062	17735680	20	9
15	OK	0.062	17760256	23	11
16	OK	0.078	17707008	20	9
17	OK	0.062	17739776	22	10
18	OK	0.062	17752064	23	11
19	OK	0.078	17715200	22	10
20	OK	0.062	17743872	22	10
21	OK	0.062	17747968	22	10

Задача «a+b^2»

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

Формат входного файла

Входной файл состоит из одной строки, которая содержит два целых числа a и b. Для этих чисел выполняются условия $-10^9 \le a \le 10^9$, $-10^9 \le b \le 10^9$.

Формат выходного файла

В выходной файл выведите единственное целое число — результат вычисления выражения $a + b^2$.

Листинг:

Файл Lab1_2.py

```
#открываем файл для чтения
input_file = open('input.txt', 'r')
#считываем строку
params = input_file.read().split(" ")
#производим мат.операцию
sum = int(params[0]) + int(params[1])**2
output_file = open('output.txt','w')
#записываем в файл
output_file.write(str(sum))
```

Бенчмарк:

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.093	17825792	25	19
1	OK	0.062	17772544	7	3
2	OK	0.093	17760256	8	3
3	OK	0.078	17711104	5	1
4	OK	0.078	17772544	5	1
5	OK	0.062	17772544	6	1
6	OK	0.078	17666048	6	1
7	OK	0.062	17813504	23	19
8	OK	0.062	17813504	25	18
9	OK	0.078	17756160	24	18
10	OK	0.062	17825792	24	19
11	OK	0.062	17788928	23	18
12	OK	0.093	17760256	23	18
13	OK	0.078	17793024	20	15
14	OK	0.062	17788928	23	18
15	OK	0.093	17752064	20	18
16	OK	0.093	17797120	22	18
17	OK	0.062	17809408	23	18
18	OK	0.093	17739776	22	17
19	OK	0.062	17788928	22	17
20	OK	0.062	17788928	22	18

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.

Листинг:

Файл Lab1_3.py

```
#открываем файл
input_file = open('input.txt', "r")
#считываем строку
for number, line in enumerate(input_file.readlines()):
if number == 0:
       lenght = int(line)
       continue
elements = [int(num) for num in line.split(' ')]

position = []
#производим сортировку вставками
for i in range(lenght):
    j = i - 1
    key = elements[i]
    while elements[j] > key and j >= 0:
       elements[j + 1] = elements[j]
       j -= 1
   elements[j + 1] = key
   position.append(j+2)
  #записываем в файл
  output_file = open("output.txt", "w")
  output_file.write(' '.join(str(pos) for pos in position) + '\n')
  output_file.write(' '.join(str(els) for els in elements))
```

Бенчмарк:

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		1.187	10371072	10415	14296
1	OK	0.031	8921088	25	40
2	OK	0.046	9007104	7	5
3	OK	0.031	8986624	12	12
4	OK	0.031	9015296	8	8
5	OK	0.046	8908800	10	12
6	OK	0.031	8953856	29	31
7	OK	0.031	9023488	10	12
8	OK	0.031	8876032	10	12
9	OK	0.046	8945664	10	12
10	OK	0.031	9011200	10	12
11	OK	0.031	8908800	10	12
12	OK	0.046	8970240	57	63
13	OK	0.031	9105408	56	62
14	OK	0.046	8908800	57	63
15	OK	0.062	8941568	77	87
16	OK	0.031	9003008	76	86
17	OK	0.031	8896512	77	87
18	OK	0.031	8941568	112	127
19	OK	0.046	8953856	111	127
20	OK	0.046	9015296	110	125
21	OK	0.062	9039872	949	1190
22	OK	0.046	8986624	960	1219
23	OK	0.062	9060352	957	1134
24	OK	0.093	9052160	1490	1888
25	OK	0.062	8978432	1486	1944
26	OK	0.078	9039872	1481	1761
27	OK	0.203	9285632	3723	4888
28	OK	0.203	9326592	3729	5047
29	OK	0.218	9248768	3727	4437
30	OK	0.765	9945088	8456	11338
31	OK	0.671	9981952	8471	11609
32	OK	0.812	9576448	8415	10035
33	OK	1.140	9981952	10415	14035
34	OK	0.984	10371072	10410	14296
35	OK	1.187	9654272	10393	12386

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 ($3 \le n \le 9999$, 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 10^6.

Формат выходного файла

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

Листинг:
Файл Lab1_4.py

```
#открываем файл
input_file = open('input.txt', 'r')
#считываем строку
for number, line in enumerate(input_file.readlines()):
if number == 0:
  lenght = int(line)
  continue
elements = [float(num) for num in line.split(' ')]


#копируем массив
temp_el = elements[:]
#производим сортировку массива (пузырьком)
for i in range(int(lenght)-1):
      for j in range(int(lenght)-1):
            if temp_el[j] < temp_el[j+1]:
                  tmp = temp_el[j]
                  temp_el[j] = temp_el[j+1]
                  temp_el[j+1] = tmp
#находим средний элемент в отсортированном массиве
for i in range(int(lenght)):
      for j in range(int(lenght)):
            if elements[j] == temp_el[int((lenght)/2 - 0.5)]:
                  med_pos = j
                  continue
max = elements.index(max(elements))+1
med = med_pos+1
min = elements.index(min(elements))+1
output_file = open("output.txt", "w")
output_file.write(str(min) + ' ' + str(med) + ' ' + str(max))
```

Бенчмарк:

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		1.593	21299200	98892	14
1	OK	0.078	18051072	30	5
2	OK	0.062	18038784	33	5
3	OK	0.093	19095552	1065	8
4	OK	0.078	19476480	3732	10
5	OK	0.140	19914752	14975	13
6	OK	0.140	20017152	14998	11
7	OK	0.250	19955712	28749	14
8	OK	0.312	20353024	34791	12
9	OK	0.328	20463616	38037	13
10	OK	0.312	20451328	38074	14
11	OK	0.343	20557824	39288	13
12	OK	0.453	20721664	48638	13
13	OK	0.515	20365312	50722	12
14	OK	0.531	20570112	52757	14
15	OK	0.625	20426752	58008	13
16	OK	0.781	20434944	66504	14
17	OK	0.875	20762624	71786	14
18	OK	0.890	20844544	72346	14
19	OK	0.921	20836352	73304	13
20	OK	0.984	21299200	76139	14
21	OK	1.171	20885504	83944	14
22	OK	1.203	20754432	85179	13
23	OK	1.218	20987904	86522	12
24	OK	1.312	21004288	89202	13
25	OK	1.593	21008384	98892	14

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 ($1 \leq n \leq 5000$) — число элементов в массиве. Во второй строке находятся n целых чисел, по модулю не превосходящих 10^9. 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 $(1 \leq X, Y \leq n)$. 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.

Листинг
Файл Lab1_5.py

```
#открываем файл
input_file = open('input.txt', 'r')
#считываем строку
for number, line in enumerate(input_file.readlines()):
  if number == 0:
     lenght = int(line)
     continue
  elements = [int(num) for num in line.split(' ')]
output_file = open('output.txt','w')
#сортируем массив
for i in range(int(lenght)-1):
      min_el = min(elements[i:])
      position = elements[i:].index(min_el) + i
      if min_el < elements[i]:
```

```
            elements[i], elements[position] = min_el, elements[i]
            #записываем строку в файл
            output_file.write(f'Swap elements at indices {i+1} and {
                                                        position+1}.\n')
output_file.write('No more swaps needed.\n')
string = ''
for i in range(int(lenght)):
      string += str(elements[i]) + ' '
output_file.write(string)
```