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

Дисциплина «**Алгоритмы и структуры данных**»

ОТЧЁТ

по лабораторным работам за 1 неделю

Студент **Пастухов К.А. группы P3218**

Санкт-Петербург
2018 г.

Задача «a+b»

В данной задаче требуется вычислить сумму двух заданных чисел.

Выполнение:

```
def sum():
    with open('input.txt', "r") as file:
        a = file.readline()
        b = a.split(" ")
        sum = int(b[0]) + int(b[1])
    with open("output.txt", "w") as res:
        res.write(str(sum))


if __name__ == "__main__":
    sum()
```

Задача «a+b^2»

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.046	9003008	25	11
1	OK	0.015	8978432	7	2
2	OK	0.031	8896512	8	3
3	OK	0.031	8908800	5	1
4	OK	0.031	8974336	5	1
5	OK	0.046	8925184	6	1
6	OK	0.031	8892416	9	4
7	OK	0.031	8962048	23	10
8	OK	0.031	8900608	25	11
9	OK	0.031	8863744	24	1
10	OK	0.031	8974336	24	1
11	OK	0.031	8921088	14	10
12	OK	0.031	8884224	23	10
13	OK	0.046	9003008	23	11
14	OK	0.031	8859648	20	9
15	OK	0.031	8904704	23	11
16	OK	0.031	8986624	20	9
17	OK	0.031	8916992	22	10
18	OK	0.046	8900608	23	11
19	OK	0.031	8957952	22	10
20	OK	0.046	8929280	22	10
21	OK	0.031	8855552	22	10

Задача «a+b^2»

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

Выполнение:

```
from decimal import Decimal


def sum():
    with open('input.txt', "r") as file:
        a = file.readline()
        b = a.split(" ")
        dec_a = Decimal(b[0])
        dec_b = Decimal(b[1]).__pow__(2)
        sum = dec_a + dec_b
    with open("output.txt", "w") as res:
        res.write(str(int(sum)))


if __name__ == "__main__":
    sum()
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.093	9977856	25	19
1	OK	0.078	9961472	7	3
2	OK	0.046	9854976	8	3
3	OK	0.062	9867264	5	1
4	OK	0.046	9977856	5	1
5	OK	0.046	9809920	6	1
6	OK	0.062	9879552	6	1
7	OK	0.046	9949184	23	19
8	OK	0.046	9859072	25	18
9	OK	0.062	9846784	24	18
10	OK	0.046	9953280	24	19
11	OK	0.031	9875456	23	18
12	OK	0.046	9904128	23	18
13	OK	0.046	9908224	20	15
14	OK	0.046	9895936	23	18
15	OK	0.046	9809920	20	18
16	OK	0.046	9928704	22	18
17	OK	0.046	9867264	23	18
18	OK	0.062	9822208	22	17
19	OK	0.093	9932800	22	17
20	OK	0.046	9871360	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.

Выполнение:

```
def insertion(len, data):
    index = []
    for i in range(len):
        j = i - 1
        key = data[i]
        while data[j] > key and j >= 0:
            data[j + 1] = data[j]
            j -= 1
        data[j + 1] = key
        index.append(j+2)
    with open("output.txt", "w") as res:
        res.write(' '.join(str(i) for i in index) + '\n')
        res.write(' '.join(str(d) for d in data))


def sum():
    with open('input.txt', "r") as file:
        for number, line in enumerate(file.readlines()):
            if number == 0:
                len = int(line)
                continue
            data = [int(num) for num in line.split(' ')]
    return len, data


if __name__ == "__main__":
    len, data = sum()
    insertion(len, data)
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.187	9252864	10415	14296
1	OK	0.031	8978432	25	40
2	OK	0.031	9056256	7	5
3	OK	0.031	8986624	12	12
4	OK	0.031	9068544	8	8
5	OK	0.015	8990720	10	12
6	OK	0.015	8986624	29	31
7	OK	0.031	9076736	10	12
8	OK	0.031	8990720	10	12
9	OK	0.031	8994816	10	12
10	OK	0.031	9056256	10	12
11	OK	0.031	8990720	10	12
12	OK	0.031	8966144	57	63
13	OK	0.015	9027584	56	62
14	OK	0.031	9007104	57	63
15	OK	0.046	8949760	77	87
16	OK	0.031	9072640	76	86
17	OK	0.031	9015296	77	87
18	OK	0.031	8966144	112	127
19	OK	0.031	9068544	111	127
20	OK	0.031	9035776	110	125
21	OK	0.031	8966144	949	1190
22	OK	0.031	9093120	960	1219
23	OK	0.062	9031680	957	1134
24	OK	0.031	8986624	1490	1888
25	OK	0.046	9089024	1486	1944
26	OK	0.031	9023488	1481	1761
27	OK	0.046	8990720	3723	4888
28	OK	0.078	9117696	3729	5047
29	OK	0.062	9027584	3727	4437
30	OK	0.078	9043968	8456	11338
31	OK	0.031	9129984	8471	11609
32	OK	0.125	9097216	8415	10035
33	OK	0.109	9125888	10415	14035
34	OK	0.046	9252864	10410	14296
35	OK	0.187	9072640	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.

Выполнение:

```
def get_middle_index(index):
    return int((index - 1)/2)


def sum():
    with open('input.txt', "r") as file:
        for number, line in enumerate(file.readlines()):
            if number == 0:
                len = int(line)
                continue
            data = [[i+1, float(num)] for i, num in
enumerate(line.split(' '))]
    return len, data


if __name__ == "__main__":
    len, data = sum()
    sorted_data = sorted(data, key=lambda x: x[1])

    min = sorted_data[0][0]
    mid = sorted_data[get_middle_index(len)][0]
    max = sorted_data[-1][0]

with open("output.txt", "w") as res:
    res.write(str(min) + " " + str(mid) + " " + str(max))
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.078	11407360	98892	14
1	OK	0.031	9129984	30	5
2	OK	0.062	8941568	33	5
3	OK	0.031	8970240	1065	8
4	OK	0.031	9019392	3732	10
5	OK	0.031	9347072	14975	13
6	OK	0.046	9261056	14998	11
7	OK	0.046	9641984	28749	14
8	OK	0.046	9793536	34791	12
9	OK	0.031	9850880	38037	13
10	OK	0.046	9912320	38074	14
11	OK	0.046	9924608	39288	13
12	OK	0.031	10031104	48638	13
13	OK	0.031	10133504	50722	12
14	OK	0.078	10248192	52757	14
15	OK	0.046	10317824	58008	13
16	OK	0.031	10551296	66504	14
17	OK	0.046	10715136	71786	14
18	OK	0.046	10694656	72346	14
19	OK	0.046	10641408	73304	13
20	OK	0.046	10829824	76139	14
21	OK	0.031	10895360	83944	14
22	OK	0.062	11014144	85179	13
23	OK	0.046	10977280	86522	12
24	OK	0.046	11120640	89202	13
25	OK	0.046	11407360	98892	14

Секретарь Своп

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.

Выполнение:

```
with open('input.txt', "r") as file:
    _ = int(file.readline())
    data = [int(num) for num in file.readline().split(' ')]
with open("output.txt", "w") as res:
    for i in range(len(data)):
        minimal = min(data[i:])
        index = data[i:].index(minimal) + i
        if minimal < data[i]:
            data[i], data[index] = minimal, data[i]
            res.write(f'Swap elements at indices {i+1} and {index+1}.
\n')

    res.write("No more swaps needed.\n")
    res.write(' '.join(str(d) for d in data))
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.781	10035200	51993	255415
1	OK	0.046	8945664	14	137
2	OK	0.031	8986624	7	25
3	OK	0.031	9035776	12	30
4	OK	0.031	8884224	8	61
5	OK	0.031	8945664	10	28
6	OK	0.031	8896512	10	28
7	OK	0.046	9035776	29	47
8	OK	0.031	8941568	10	63
9	OK	0.031	8904704	10	63
10	OK	0.046	9068544	10	98
11	OK	0.031	8945664	10	63
12	OK	0.031	9023488	10	98
13	OK	0.031	8884224	50	138
14	OK	0.031	8941568	56	179
15	OK	0.031	9023488	57	75
16	OK	0.031	8945664	55	143
17	OK	0.046	8904704	75	303
18	OK	0.046	9011200	76	94
19	OK	0.031	8949760	78	201
20	OK	0.031	8925184	108	266
21	OK	0.031	9011200	107	124
22	OK	0.031	8949760	108	301
23	OK	0.046	8916992	948	4175
24	OK	0.031	9035776	947	964
25	OK	0.031	8949760	948	2621
26	OK	0.031	9113600	3720	17382
27	OK	0.062	9138176	3735	3751
28	OK	0.046	9187328	3722	10611
29	OK	0.062	9166848	8463	39802
30	OK	0.046	9261056	8441	8457
31	OK	0.046	9236480	8434	24176
32	OK	0.171	9474048	22822	111244
33	OK	0.125	9318400	22825	22840
34	OK	0.156	9396224	22877	66844
35	OK	0.781	10035200	51987	255415
36	OK	0.546	9805824	51940	51955
37	OK	0.703	9887744	51993	153401