ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Факультет программной инженерии и компьютерной техники

ЛАБОРАТОРНАЯ РАБОТА № 1

ПО ДИСЦИПЛИНЕ «Алгоритмы и структуры данных»

Выполнил: Мещеряков Владимир Алевтинович

Группа: P3218

Преподаватель: Романов Алексей Андреевич

Волчек Дмитрий Геннадьевич

Санкт-Петербург

2019 г.

Week 1

1) Задача «a+b»

1.0 из 1.0 балла (оценивается)

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

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 $-109 \le a \le 109$, $-109 \le b \le 109$.

Формат выходного файла

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

Примеры

input.txt	output.txt
23 11	34
-100 1	-99

Исходный код:

```
static void Man21()
        {
            string path = "input.txt";
            string writePath = "output.txt";
            string[] split;
            using (StreamReader sr = new StreamReader(path, System.Text.Encoding.Default))
            {
                string line;
                line = sr.ReadLine();
                split = line.Split(' ');

            }
```

```
            using (StreamWriter sw = new StreamWriter(writePath, false,
System.Text.Encoding.Default))
            {
                sw.WriteLine(Convert.ToInt32(split[0]) + Convert.ToInt32(split[1]));
            }
}
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.046	10211328	25	13
1	OK	0.015	10125312	7	4
2	OK	0.031	10092544	8	5
3	OK	0.031	10149888	5	3
4	OK	0.031	10088448	5	3
5	OK	0.031	10092544	6	3
6	OK	0.031	10113024	9	6
7	OK	0.031	10088448	23	12
8	OK	0.046	10108928	25	13
9	OK	0.031	10153984	24	3
10	OK	0.031	10125312	24	3
11	OK	0.031	10076160	14	12
12	OK	0.031	10211328	23	12
13	OK	0.015	10088448	23	13
14	OK	0.015	10092544	20	11
15	OK	0.015	10121216	23	13
16	OK	0.031	10137600	20	11
17	OK	0.015	10108928	22	12
18	OK	0.031	10108928	23	13
19	OK	0.031	10096640	22	12
20	OK	0.015	10145792	22	12
21	OK	0.031	10100736	22	12

2)Задача «a+b^2»

1.0 из 1.0 балла (оценивается)

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

В данной задаче требуется вычислить значение выражения a+b2.

Формат входного файла

Входной файл состоит из одной строки, которая содержит два целых числа a и b. Для этих чисел выполняются условия $-109 \le a \le 109$, $-109 \le b \le 109$.

Формат выходного файла

В выходной файл выведите единственное целое число — результат вычисления выражения $a+b2$.

Примеры

input.txt	output.txt
23 11	144
-100 1	-99

```
static void Second()
        {
            string path = "input.txt";
            string writePath = "output.txt";
            string[] split;
            using (StreamReader sr = new StreamReader(path, System.Text.Encoding.Default))
            {
                string line;
                line = sr.ReadLine();
                split = line.Split(' ');

            }


            using (StreamWriter sw = new StreamWriter(writePath, false,
System.Text.Encoding.Default))
            {
                sw.WriteLine(Convert.ToInt64(split[0]) + Convert.ToInt64(split[1]) *
Convert.ToInt64(split[1]));
            }
}
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.078	10194944	25	21
1	OK	0.078	10149888	7	5
2	OK	0.015	10137600	8	5
3	OK	0.031	10153984	5	3
4	OK	0.015	10141696	5	3
5	OK	0.015	10141696	6	3
6	OK	0.031	10149888	6	3
7	OK	0.015	10186752	23	21
8	OK	0.031	10149888	25	20
9	OK	0.015	10149888	24	20
10	OK	0.031	10170368	24	21
11	OK	0.031	10141696	23	20
12	OK	0.031	10141696	23	20
13	OK	0.015	10190848	20	17
14	OK	0.015	10158080	23	20
15	OK	0.015	10194944	20	20
16	OK	0.031	10162176	22	20
17	OK	0.031	10117120	23	20
18	OK	0.015	10121216	22	19
19	OK	0.015	10133504	22	19
20	OK	0.031	10194944	22	20

3)Сортировка вставками

1.0 из 1.0 балла (оценивается)

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

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.

Пример

input.txt	output.txt
10 1 8 4 2 3 7 5 6 9 0	1 2 2 2 3 5 5 6 9 1 0 1 2 3 4 5 6 7 8 9

```
static void ex3()
        {
            string path = "input.txt";
            string writePath = "output.txt";
            string[] split;
            int count;
            using (StreamReader sr = new StreamReader(path, System.Text.Encoding.Default))
            {
                string line;
                count = int.Parse(sr.ReadLine());
                line = sr.ReadLine();
                split = line.Split(' ');
            }
            int[] index = new int[count];
            index[0] = 1;


            int[] myInts = Array.ConvertAll(split, int.Parse);


            for (int i = 1; i < myInts.Length; i++)
            {
                int cur = myInts[i];
                int j = i;
                while (j > 0 && cur < myInts[j - 1])
                {
                    myInts[j] = myInts[j - 1];
                    j--;
                }
                myInts[j] = cur;

                index[i] = j + 1;
            }


            using (StreamWriter sw = new StreamWriter(writePath, false,
System.Text.Encoding.Default))
            {
                string output = "";
                for (int i = 0; i < index.Length; i++)
                {
                    output = output + Convert.ToString(index[i]) + " ";
                }
                sw.WriteLine(output);
                output = "";
                for (int i = 0; i < myInts.Length; i++)
                {
                    output = output + Convert.ToString(myInts[i]) + " ";
                }
                sw.WriteLine(output);

            }
        }
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.031	11718656	10415	14300
1	OK	0.015	10489856	25	44
2	OK	0.031	10493952	7	9
3	OK	0.031	10502144	12	16
4	OK	0.031	10489856	8	12
5	OK	0.031	10518528	10	16
6	OK	0.015	10522624	29	35
7	OK	0.031	10489856	10	16
8	OK	0.031	10457088	10	16
9	OK	0.031	10457088	10	16
10	OK	0.031	10526720	10	16
11	OK	0.031	10522624	10	16
12	OK	0.015	10506240	57	67
13	OK	0.031	10518528	56	66
14	OK	0.031	10522624	57	67
15	OK	0.031	10493952	77	91
16	OK	0.031	10514432	76	90
17	OK	0.031	10502144	77	91
18	OK	0.031	10498048	112	131
19	OK	0.031	10498048	111	131
20	OK	0.015	10473472	110	129
21	OK	0.031	10641408	949	1194
22	OK	0.031	10620928	960	1223
23	OK	0.031	10653696	957	1138
24	OK	0.031	10817536	1490	1892
25	OK	0.031	10858496	1486	1948
26	OK	0.031	10784768	1481	1765
27	OK	0.031	11628544	3723	4892
28	OK	0.031	11612160	3729	5051
29	OK	0.031	11608064	3727	4441
30	OK	0.031	11694080	8456	11342
31	OK	0.015	11706368	8471	11613
32	OK	0.031	11718656	8415	10039
33	OK	0.031	11714560	10415	14039
34	OK	0.031	11702272	10410	14300
35	OK	0.031	11710464	10393	12390

4)Знакомство с жителями Сортлэнда

1.0 из 1.0 балла (оценивается)

Имя входного файла:	input.txt

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

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.

Пример

input.txt	output.txt
5 10.00 8.70 0.01 5.00 3.00	3 4 1

```
static void ex4()
 {
     string path = "input.txt";
     string writePath = "output.txt";
     string[] split;
     int count;
     using (StreamReader sr = new StreamReader(path, System.Text.Encoding.Default))
     {
```

```
            string line;
            count = int.Parse(sr.ReadLine());
            line = sr.ReadLine();
            split = line.Split(' ');
        }


        double[] myInts = Array.ConvertAll(split, double.Parse);
        double[] array = Array.ConvertAll(split, double.Parse);

        for (int i = 1; i < myInts.Length; i++)
        {
            double cur = myInts[i];
            int j = i;
            while (j > 0 && cur < myInts[j - 1])
            {
                myInts[j] = myInts[j - 1];
                j--;
            }
            myInts[j] = cur;
        }

        using (StreamWriter sw = new StreamWriter(writePath, false,
System.Text.Encoding.Default))
        {
            string output = "";

            for (int i = 0; i < array.Length; i++)
            {
                if (array[i] == myInts[0])
                    output += i + 1 + " ";

            }
            for (int i = 0; i < array.Length; i++)
            {
                if (array[i] == myInts[(count + 1) / 2 - 1])
                    output += i + 1 + " ";

            }
            for (int i = 0; i < array.Length; i++)
            {
                if (array[i] == myInts[myInts.Length - 1])
                    output += i + 1;

            }
            sw.WriteLine(output);
        }
    }
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.078	11694080	98892	16
1	OK	0.031	10493952	30	7
2	OK	0.031	10555392	33	7
3	OK	0.031	10579968	1065	10
4	OK	0.031	10612736	3732	12
5	OK	0.031	10764288	14975	15
6	OK	0.015	10743808	14998	13
7	OK	0.015	10895360	28749	16
8	OK	0.031	10944512	34791	14
9	OK	0.031	11001856	38037	15
10	OK	0.031	11010048	38074	16
11	OK	0.046	11014144	39288	15
12	OK	0.031	11104256	48638	15
13	OK	0.046	11194368	50722	14
14	OK	0.031	11186176	52757	16
15	OK	0.046	11255808	58008	15
16	OK	0.046	11321344	66504	16
17	OK	0.062	11386880	71786	16
18	OK	0.046	11370496	72346	16
19	OK	0.046	11415552	73304	15
20	OK	0.062	11407360	76139	16
21	OK	0.062	11530240	83944	16
22	OK	0.046	11558912	85179	15
23	OK	0.078	11550720	86522	14
24	OK	0.062	11608064	89202	15
25	OK	0.062	11694080	98892	16

5)Секретарь Своп

1.0 из 1.0 балла (оценивается)

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

Ограничение по памяти:	256 мегабайт

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 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 ($1 \le X, Y \le n$). 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.

Пример

input.txt	output.txt
5 3 1 4 2 2	Swap elements at indices 1 and 2. Swap elements at indices 2 and 4. Swap elements at indices 3 and 5. No more swaps needed. 1 2 2 3 4

```
static void ex5()
        {
            string path = "input.txt";
            string writePath = "output.txt";
            string[] split;
            int count;
            using (StreamReader sr = new StreamReader(path, System.Text.Encoding.Default))
            {
                string line;
                count = int.Parse(sr.ReadLine());
                line = sr.ReadLine();
                split = line.Split(' ');
            }

            int[] myInts = new int[count];
            for (int i = 0; i < split.Length; i++)
            {
                myInts[i] = int.Parse(split[i]);

            }


            using (StreamWriter sw = new StreamWriter(writePath, false,
System.Text.Encoding.Default))
            {
                quicksort(myInts, 0, myInts.Length - 1,sw);
                string output = "";
                sw.WriteLine("No more swaps needed.");
                for (int i = 0; i < myInts.Length; i++)
                {
                    output += myInts[i] + " ";
                }
                sw.WriteLine(output);
            }

            int partition(int[] array, int start, int end, StreamWriter str)
            {
                    int temp;//swap helper
                    int marker = start;//divides left and right subarrays
                    for (int i = start; i <= end; i++)
                    {
                        if (array[i] < array[end]) //array[end] is pivot
                        {
                            temp = array[marker]; // swap
                            if (marker != i)
                                str.WriteLine("Swap elements at indices " + (marker+1) + " and " +
(i+1) + ".");
                            array[marker] = array[i];
                            array[i] = temp;
                            marker += 1;
                        }
                    }
                    //put pivot(array[end]) between left and right subarrays
                    temp = array[marker];
                    array[marker] = array[end];
                    array[end] = temp;
                    if (marker != end)
                    str.WriteLine("Swap elements at indices " + (marker+1) + " and " + (end+1) +
".");
                    return marker;
            }

            void quicksort(int[] array, int start, int end, StreamWriter str)
            {
                if (start >= end)
                {
                    return;
                }
                int pivot = partition(array, start, end, str);
```

```
        quicksort(array, start, pivot - 1, str);
        quicksort(array, pivot + 1, end, str);
    }
}
```

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.156	15458304	51993	1270653
1	OK	0.031	10178560	14	175
2	OK	0.015	10174464	7	28
3	OK	0.015	10182656	12	33
4	OK	0.031	10162176	8	64
5	OK	0.015	10149888	10	66
6	OK	0.031	10186752	10	31
7	OK	0.031	10235904	29	50
8	OK	0.031	10211328	10	66
9	OK	0.015	10170368	10	66
10	OK	0.031	10162176	10	101
11	OK	0.031	10219520	10	66
12	OK	0.031	10186752	10	101
13	OK	0.031	10244096	50	176
14	OK	0.015	10194944	56	182
15	OK	0.031	10153984	57	78
16	OK	0.031	10227712	55	146
17	OK	0.031	10178560	75	376
18	OK	0.031	10170368	76	97
19	OK	0.031	10276864	78	204
20	OK	0.031	10158080	108	479
21	OK	0.031	10199040	107	127
22	OK	0.015	10158080	108	304
23	OK	0.031	10407936	948	11513
24	OK	0.031	10301440	947	967
25	OK	0.015	10313728	948	2624
26	OK	0.031	11378688	3720	68936
27	OK	0.015	11345920	3735	3754
28	OK	0.031	11276288	3722	10614
29	OK	0.031	11395072	8463	167272
30	OK	0.031	11505664	8441	8460
31	OK	0.031	11419648	8434	24179
32	OK	0.062	11530240	22822	506184
33	OK	0.046	11657216	22825	22843
34	OK	0.046	11554816	22877	66847
35	OK	0.109	15204352	51987	1270653
36	OK	0.125	15458304	51940	51958
37	OK	0.156	15454208	51993	153404