Санкт-Петербургский Национальный Исследовательский Университет
Информационных Технологий, Механики и Оптики

Лабораторная работа №1

по дисциплине

«Алгоритмы и структуры данных»

Выполнил: Студент группы P3217

Сергачев Данила Дмитриевич

Преподаватели:

Романов Алексей Андреевич и

Волчек Дмитрий Геннадьевич

Санкт-Петербург

2018 г

Задание №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 $-10^9 \le a \le 10^9$, $-10^9 \le b \le 10^9$.

Формат выходного файла

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

Решение

```
package
week1;

        import mooc.*;

        public class task1 {
            public static void main(String[] args) {
                try (EdxIO io = EdxIO.create()) {
                    io.println(io.nextInt() + io.nextInt());
                }
            }
        }
```

Результат работы

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.140	21635072	25	13
1	OK	0.125	21569536	7	4
2	OK	0.125	21585920	8	5
3	OK	0.109	21585920	5	3
4	OK	0.125	21520384	5	3
5	OK	0.125	21635072	6	3
6	OK	0.109	21598208	9	6
7	OK	0.109	21610496	23	12
8	OK	0.109	21581824	25	13
9	OK	0.109	21549056	24	3
10	OK	0.109	21610496	24	3
11	OK	0.109	21524480	14	12
12	OK	0.125	21581824	23	12
13	OK	0.140	21594112	23	13
14	OK	0.109	21557248	20	11
15	OK	0.109	21536768	23	13
16	OK	0.140	21565440	20	11
17	OK	0.109	21569536	22	12
18	OK	0.109	21573632	23	13
19	OK	0.125	21630976	22	12
20	OK	0.125	21581824	22	12
21	OK	0.125	21549056	22	12

Задание №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 $a + b^2$.

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 $a + b^2$.

Решение

```
package week1;
import mooc.*;

public class task2 {
   public static void main(String[] args){
       try (EdxIO io = EdxIO.create()) {
           long nextA = io.nextLong();
           long nextB = io.nextLong();

           io.println((long)(nextA + nextB * nextB ));
       }
   }
}
```

Результат работы

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.140	21630976	25	21
1	OK	0.125	21594112	7	5
2	OK	0.109	21561344	8	5
3	OK	0.109	21561344	5	3
4	OK	0.109	21532672	5	3
5	OK	0.093	21610496	6	3
6	OK	0.125	21614592	6	3
7	OK	0.125	21577728	23	21
8	OK	0.125	21553152	25	20
9	OK	0.109	21557248	24	20
10	OK	0.109	21630976	24	21
11	OK	0.125	21594112	23	20
12	OK	0.140	21532672	23	20
13	OK	0.109	21594112	20	17
14	OK	0.125	21565440	23	20
15	OK	0.125	21553152	20	20
16	OK	0.125	21614592	22	20
17	OK	0.125	21565440	23	20
18	OK	0.125	21573632	22	19
19	OK	0.125	21614592	22	19
20	OK	0.109	21553152	22	20

Задание №3

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.

Решение

```
package week1;
import mooc.*;

public class task3 {
   public static void main(String[] args){
       try (EdxIO io = EdxIO.create()) {

            int count = io.nextInt();
            long[] array = new long[count];

            for(int i = 0; i < count; i++){
                array[i] = io.nextLong();
            }

            sort(array, count, io);
            for (long item : array) {
                io.print(String.format("%d ", item));
            }
        }
    }

    private static void sort(long[] array, int count, EdxIO io){
```

```
        io.print(String.format("%d ", 1));

        for(int i = 1; i< count; i++){
            long key = array[i];
            int j = i-1;
            while(j >= 0 && array[j] > key){
                array[j+1] = array[j];
                j--;
            }
            array[j+1] = key;

            io.print(String.format("%d ", j+2));
        }
        io.println("\n");
    }
                                                                    }
```

Результат работы

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.250	28160000	10415	14300
1	OK	0.125	21725184	25	44
2	OK	0.109	21741568	7	9
3	OK	0.125	21782528	12	16
4	OK	0.125	21782528	8	12
5	OK	0.109	21733376	10	16
6	OK	0.109	21721088	29	35
7	OK	0.156	21819392	10	16
8	OK	0.125	21745664	10	16
9	OK	0.125	21762048	10	16
10	OK	0.125	21811200	10	16
11	OK	0.109	21692416	10	16
12	OK	0.109	21757952	57	67
13	OK	0.109	21794816	56	66
14	OK	0.109	21733376	57	67
15	OK	0.109	21790720	77	91
16	OK	0.109	21782528	76	90
17	OK	0.125	21704704	77	91
18	OK	0.109	21741568	112	131
19	OK	0.109	21811200	111	131
20	OK	0.109	21766144	110	129
21	OK	0.125	21983232	949	1194
22	OK	0.156	22036480	960	1223
23	OK	0.140	21954560	957	1138
24	OK	0.156	22462464	1490	1892
25	OK	0.171	22462464	1486	1948
26	OK	0.171	23007232	1481	1765
27	OK	0.171	23945216	3723	4892
28	OK	0.171	23646208	3729	5051
29	OK	0.156	24047616	3727	4441
30	OK	0.171	27230208	8456	11342
31	OK	0.218	27090944	8471	11613
32	OK	0.203	27119616	8415	10039
33	OK	0.203	28028928	10415	14039
34	OK	0.218	27230208	10410	14300
35	OK	0.250	28160000	10393	12390

Задание №4

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.

Решение

```
package week1;
import mooc.*;

public class task4 {
    public static void main(String[] args){
        try (EdxIO io = EdxIO.create()) {

            int count = io.nextInt();

            Pair[] pairs = new Pair[count];

            for(int i = 0; i < count; i++){
                pairs[i] = new Pair(io.nextDoublePrecise(), i);
            }

            sort(pairs, count, io);

            io.print(String.format("%d ", pairs[0].Key+1));
            io.print(String.format("%d ", pairs[count/2].Key+1));
            io.print(String.format("%d ", pairs[count-1].Key+1));
        }
```

```
    }

    private static void sort(Pair[] pairs, int count, EdxIO io){
        for(int i = 1; i< count; i++){
           Pair item = pairs[i];
           int j = i-1;

           while(j >= 0 && pairs[j].Value > item.Value){
               pairs[j+1] = pairs[j];
               j--;
            }

            pairs[j+1] = item;
        }
    }

    private static class Pair{
        public double Value;
        public int Key;

        public Pair(double value, int key){
            Value = value;
            Key = key;
        }
    }
}
```

Результат работы

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.328	27557888	98892	15
1	OK	0.140	21884928	30	6
2	OK	0.140	21856256	33	6
3	OK	0.171	21831680	1065	9
4	OK	0.140	21970944	3732	11
5	OK	0.156	24449024	14975	14
6	OK	0.156	24039424	14998	12
7	OK	0.187	26275840	28749	15
8	OK	0.156	26271744	34791	13
9	OK	0.218	26132480	38037	14
10	OK	0.234	26324992	38074	15
11	OK	0.234	26337280	39288	14
12	OK	0.250	26796032	48638	14
13	OK	0.265	26656768	50722	13
14	OK	0.250	26451968	52757	15
15	OK	0.296	26828800	58008	14
16	OK	0.296	27017216	66504	15
17	OK	0.281	26767360	71786	15
18	OK	0.281	27164672	72346	15
19	OK	0.281	26968064	73304	14
20	OK	0.281	27074560	76139	15
21	OK	0.281	27324416	83944	15
22	OK	0.296	27115520	85179	14
23	OK	0.281	27066368	86522	13
24	OK	0.281	27410432	89202	14
25	OK	0.328	27557888	98892	15

Задание №5

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.

Решение

```
Package week1;
import mooc.*;

public class task5 {

    private static long[] array;
    public static void main(String[] args){
        try (EdxIO io = EdxIO.create()) {

            int count = io.nextInt();
            array = new long[count];
```

```
            for(int i = 0; i < count; i++){
                array[i] = io.nextLong();
            }

            qSort(io);

            for (long item : array) {
                io.print(String.format("%d ", item));
            }
        }
    }

    private static void qSort(EdxIO io){
        int startIndex = 0;
        int endIndex = array.length - 1;
        doSort(startIndex, endIndex, io);

        displayEndOfSwap(io);
    }

    private static void doSort(int startIndex, int endIndex, EdxIO io){
       if(startIndex >= endIndex){
           return;
       }

       int start = startIndex, end = endIndex;
       // Выбор индекса опорного элемента
       int middleIndex = startIndex + (endIndex - startIndex) / 2;
       while(start < end){
           while((array[start] <= array[middleIndex]) && start < middleIndex) { start++;
}

           while((array[middleIndex] <= array[end]) && end > middleIndex) { end--; }

           if(start < end){
               swap(start, end, io);
               // Задание новых границ
               if(start == middleIndex){
                   middleIndex = end;
               }
               else if(end == middleIndex){
                   middleIndex = start;
               }
           }
       }

       doSort(startIndex, middleIndex, io);
```

```
        doSort(middleIndex+1, endIndex, io);
    }

    private static void swap(int firstItemIndex, int secodItemIndex, EdxIO io){
        long tmp = array[firstItemIndex];
        array[firstItemIndex] = array[secodItemIndex];
        array[secodItemIndex] = tmp;

        displaySwapPair(firstItemIndex+1, secodItemIndex+1, io);
    }

    private static void displaySwapPair(int firstItemIndex, int secodItemIndex, EdxIO
io){
        io.println(String.format("Swap elements at indices %d and %d.", firstItemIndex,
secodItemIndex));
    }

    private static void displayEndOfSwap(EdxIO io){
        io.println("No more swaps needed.");
    }
                                                                                    }
```

Результат работы

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.593	72077312	51993	714743
1	OK	0.125	21798912	14	173
2	OK	0.109	21757952	7	26
3	OK	0.093	21676032	12	31
4	OK	0.125	21753856	8	62
5	OK	0.125	21688320	10	29
6	OK	0.109	21757952	10	29
7	OK	0.125	21741568	29	48
8	OK	0.156	21716992	10	64
9	OK	0.109	21798912	10	64
10	OK	0.187	21729280	10	99
11	OK	0.109	21708800	10	64
12	OK	0.125	21794816	10	99
13	OK	0.109	21770240	50	174
14	OK	0.109	21696512	56	180
15	OK	0.109	21790720	57	76
16	OK	0.125	21733376	55	144
17	OK	0.109	21725184	75	304
18	OK	0.109	21762048	76	95
19	OK	0.109	21749760	78	202
20	OK	0.125	21696512	108	267
21	OK	0.109	21762048	107	125
22	OK	0.125	21811200	108	302
23	OK	0.140	23162880	948	6085
24	OK	0.203	21929984	947	965
25	OK	0.187	21991424	948	2622
26	OK	0.250	28446720	3720	32944
27	OK	0.140	23216128	3735	3752
28	OK	0.171	24887296	3722	10612
29	OK	0.265	30822400	8463	88221
30	OK	0.156	24289280	8441	8458
31	OK	0.218	27303936	8434	24177
32	OK	0.437	43634688	22822	279054
33	OK	0.218	28442624	22825	22841
34	OK	0.281	30494720	22877	66845
35	OK	0.593	72077312	51987	714743
36	OK	0.296	31305728	51940	51956
37	OK	0.359	36577280	51993	153402