Университет ИТМО
Факультет программной инженерии и компьютерной техники
Алгоритмы и структуры данных

Лабораторная работа №1

Выполнил:
Ярошевич Александр P3217

Санкт-Петербург
2019

Задание №1

Сортировка слиянием

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 слиянием.

Чтобы убедиться, что Вы действительно используете сортировку слиянием, мы просим Вас, после каждого осуществленного слияния (то есть, когда соответствующий подмассив уже отсортирован!), выводить индексы граничных элементов и их значения.

Формат входного файла

В первой строке входного файла содержится число n ($1 \le n \le 105$) — число элементов в массиве. Во второй строке находятся n целых чисел, по модулю не превосходящих 109.

Формат выходного файла

Выходной файл состоит из нескольких строк.

В **последней строке** выходного файла требуется вывести отсортированный в порядке неубывания массив, данный на входе. Между любыми двумя числами должен стоять ровно один пробел.

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		1.437	29278208	1039245	4403712
1	OK	0.031	9076736	25	104
2	OK	0.031	9068544	6	1
3	OK	0.031	9072640	8	12
4	OK	0.031	9080832	8	12
5	OK	0.031	9113600	42	153
6	OK	0.031	9093120	43	153
7	OK	0.046	9129984	51	177
8	OK	0.046	9154560	45	160
9	OK	0.031	9113600	105	329
10	OK	0.031	9003008	110	342
11	OK	0.031	9080832	107	335
12	OK	0.031	9109504	461	2043
13	OK	0.031	9097216	560	2330
14	OK	0.031	9076736	388	1821
15	OK	0.062	9166848	408	1882
16	OK	0.031	9142272	1042	3775
17	OK	0.031	9121792	1043	3783
18	OK	0.031	9064448	1044	3774
19	OK	0.031	9228288	5587	25512
20	OK	0.046	9277440	6733	28936
21	OK	0.046	9170944	4737	22959
22	OK	0.031	9154560	5685	25798
23	OK	0.078	9322496	10383	39967
24	OK	0.046	9195520	10421	40059
25	OK	0.062	9216000	10420	40056
26	OK	0.140	10653696	65880	305387
27	OK	0.140	10608640	77550	340375
28	OK	0.140	10506240	57488	280212
29	OK	0.156	10469376	68090	311996
30	OK	0.156	11157504	103872	420188
31	OK	0.140	11100160	103940	420365
32	OK	0.156	11182080	103842	420121
33	OK	1.187	25296896	758839	3554250
34	OK	1.234	26923008	875802	3905101
35	OK	1.203	25014272	675241	3303453
36	OK	1.203	26112000	782803	3626112
37	OK	1.406	28241920	1038992	4403247
38	OK	1.437	29278208	1038702	4402562
39	OK	1.406	27643904	1039245	4403712

Код

```
def sort(alist, left,
right):
                                output_string = ''
                                if len(alist) > 1:
                                    mid = len(alist) // 2
                                    left_list = alist[:mid]
                                    right_list = alist[mid:]


                                    output_string += str(sort(left_list, left, left +
                            mid - 1))
                                    output_string += str(sort(right_list, left + mid,
                            right))
```

```
        index_left = 0
        index_right = 0
        position = 0

        while index_left < len(left_list) and index_right
< len(right_list):
            if left_list[index_left] <=
right_list[index_right]:
                alist[position] = left_list[index_left]
                index_left += 1
            else:
                alist[position] = right_list[index_right]
                index_right += 1
            position += 1

        while index_left < len(left_list):
            alist[position] = left_list[index_left]
            position += 1
            index_left += 1

        while index_right < len(right_list):
            alist[position] = right_list[index_right]
            position += 1
            index_right += 1

        output_string += f"{left + 1} {right + 1}
{alist[0]} {alist[-1]}\n"

    if output_string != '':
        return output_string
```

```
    else:
        return ''



inputFile = open("input.txt", "r")
length = int(inputFile.readline()) - 1
alist = [int(x) for x in inputFile.readline().split()]
inputFile.close()
outputFile = open("output.txt", "w")
res = sort(alist, 0, length)
outputFile.write(res)
outputFile.write(' '.join(map(str, alist)))
```

Задание №2

Число инверсий

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

Инверсией в последовательности чисел A называется такая ситуация, когда i<j, а Ai>Aj.

Дан массив целых чисел. Ваша задача — подсчитать число инверсий в нем.

Подсказка: чтобы сделать это быстрее, можно воспользоваться модификацией сортировки слиянием.

Формат входного файла

В первой строке входного файла содержится число n (1≤n≤105) — число элементов в массиве. Во второй строке находятся n целых чисел, по модулю не превосходящих 109.

Формат выходного файла

В выходной файл надо вывести число инверсий в массиве.

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		1.250	22269952	1039245	10
1	OK	0.015	9129984	25	2
2	OK	0.046	9101312	6	1
3	OK	0.031	9052160	8	1
4	OK	0.031	9048064	8	1
5	OK	0.046	9015296	42	1
6	OK	0.031	9117696	43	2
7	OK	0.031	9015296	51	1
8	OK	0.031	9109504	45	2
9	OK	0.031	9084928	105	2
10	OK	0.031	9048064	110	2
11	OK	0.031	9011200	107	2
12	OK	0.031	9056256	461	1
13	OK	0.015	9089024	560	4
14	OK	0.046	9076736	388	1
15	OK	0.031	9072640	408	4
16	OK	0.031	9064448	1042	4
17	OK	0.046	9052160	1043	4
18	OK	0.031	9089024	1044	4
19	OK	0.062	9113600	5587	1
20	OK	0.031	9068544	6733	6
21	OK	0.046	9056256	4737	1
22	OK	0.031	9076736	5685	6
23	OK	0.046	8994816	10383	6
24	OK	0.046	9076736	10421	6
25	OK	0.046	9097216	10420	6
26	OK	0.109	10338304	65880	1
27	OK	0.125	10080256	77550	8
28	OK	0.093	10096640	57488	1
29	OK	0.140	10092544	68090	8
30	OK	0.125	10149888	103872	8
31	OK	0.125	10133504	103940	8
32	OK	0.125	10121216	103842	8
33	OK	0.875	20877312	758839	1
34	OK	1.062	21803008	875802	10
35	OK	0.875	21454848	675241	1
36	OK	1.031	21082112	782803	10
37	OK	1.203	22269952	1038992	10
38	OK	1.234	22257664	1038702	10
39	OK	1.250	20901888	1039245	10

Код

```
def
sort(alist):
                    inversions = 0
                    if len(alist) > 1:
                        mid = len(alist) // 2
```

```
        left_list = alist[:mid]
        right_list = alist[mid:]

        inversions = sort(left_list)
        inversions += sort(right_list)

        index_left = 0
        index_right = 0
        position = 0

        while index_left < len(left_list) and index_right <
len(right_list):
            if left_list[index_left] <= right_list[index_right]:
                alist[position] = left_list[index_left]
                index_left += 1
            else:
                alist[position] = right_list[index_right]
                index_right += 1
                inversions += len(left_list) - index_left
            position += 1

        while index_left < len(left_list):
            alist[position] = left_list[index_left]
            position += 1
            index_left += 1

        while index_right < len(right_list):
            alist[position] = right_list[index_right]
            position += 1
            index_right += 1
        return inversions
```

```
    return 0
```

```
inputFile = open("input.txt", "r")
length = int(inputFile.readline()) - 1
alist = [int(x) for x in inputFile.readline().split()]
inputFile.close()
outputFile = open("output.txt", "w")
res = sort(alist)
outputFile.write(str(res))
```

Задача №3

Анти-quick sort

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

Для сортировки последовательности чисел широко используется быстрая сортировка — QuickSort. Далее приведена программа, которая сортирует массив a, используя этот алгоритм.

```
var a : array [1..N] of integer;

procedure QSort(left, right : integer);
var i, j, key, buf : integer;
begin
    key := a[(left + right) div 2];
    i := left;
    j := right;
    repeat
        while a[i] < key do
```

```
            inc(i);
        while key < a[j] do
            dec(j);
        if i <= j then begin
            buf := a[i];
            a[i] := a[j];
            a[j] := buf;
            inc(i);
            dec(j);
        end;
    until i > j;
    if left < j then QSort(left, j);
    if i < right then QSort(i, right);
end;
begin
    ...
    QSort(1, N);
end.
```

Хотя QuickSort является очень быстрой сортировкой в среднем, существуют тесты, на которых она работает очень долго. Оценивать время работы алгоритма будем числом сравнений с элементами массива (то есть, суммарным числом сравнений в первом и втором while). Требуется написать программу, генерирующую тест, на котором быстрая сортировка сделает наибольшее число таких сравнений.

Формат входного файла

В первой строке находится единственное число n ($1 \le n \le 106$).

Формат выходного файла

Вывести перестановку чисел от 1 до n, на которой быстрая сортировка выполнит максимальное число сравнений. Если таких перестановок несколько, вывести любую из них.

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		1.187	123314176	9	6888895
1	OK	0.062	8974336	3	5
2	OK	0.031	9048064	3	1
3	OK	0.031	9007104	3	3
4	OK	0.046	9007104	3	7
5	OK	0.031	8953856	3	9
6	OK	0.031	8962048	3	11
7	OK	0.031	8916992	3	13
8	OK	0.031	8957952	3	15
9	OK	0.031	8945664	3	17
10	OK	0.015	8962048	4	20
11	OK	0.031	8966144	4	35
12	OK	0.031	8953856	5	291
13	OK	0.078	9080832	6	3892
14	OK	0.046	10067968	7	48899
15	OK	0.046	10088448	7	48893
16	OK	0.156	24887296	8	756194
17	OK	0.312	36704256	8	1556238
18	OK	0.562	62779392	8	3151811
19	OK	1.187	123314176	8	6888887
20	OK	1.187	122241024	9	6888895

Код

```
inputFile = open("input.txt", "r")
length = int(inputFile.readline())
alist = [0]*length
for i in range(length):
    alist[i] = i + 1
    if i > 1:
        alist[i], alist[i//2] = alist[i//2], alist[i]

outputFile = open("output.txt", "w")
outputFile.write(' '.join(map(str, alist)))
```

Задание №4

K-ая порядковая статистика

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 из n элементов. Какие числа являются k1-ым, (k1+1)-ым, …, k2-ым в порядке неубывания в этом массиве?

Формат входного файла

В первой строке входного файла содержатся три числа: n — размер массива, а также границы интервала k1 и k2, при этом $2 \le n \le 4 \cdot 10^7$, $1 \le k1 \le k2 \le n$, $k2 - k1 < 200$. Во второй строке находятся числа A, B, C, a1, a2, по модулю не превосходящие 109. Вы должны получить элементы массива, начиная с третьего, по формуле: $a_i = A \cdot a_{i-2} + B \cdot a_{i-1} + C$. Все вычисления должны производится в 32-битном знаковом типе, переполнения должны игнорироваться.

Обращаем Ваше внимание, что массив из $4 \cdot 10^7$ 32-битных целых чисел занимает в памяти **160 мегабайт**! Будьте аккуратны!

Подсказка: эту задачу лучше всего решать модификацией быстрой сортировки. Однако сортировка массива целиком по времени, скорее всего, не пройдет, поэтому нужно подумать, как модифицировать быструю сортировку, чтобы не сортировать те части массива, которые не нужно сортировать.

Эту задачу, скорее всего, **нельзя решить ни на Python, ни на PyPy**. Мы не нашли способа сгенерировать $4 \cdot 10^7$ 32-битных целых чисел и при этом уложиться в ограничение по времени. Если у Вас тоже не получается, попробуйте другой язык программирования, например, **Cython**(расширение файла `*.pyx`).

Формат выходного файла

В первой и единственной строке выходного файла выведите k1-ое, (k1+1)-ое, …, k2-ое в порядке неубывания числа в массиве a. Числа разделяйте одним пробелом.

Код

```
using System;

using System.IO;

using System.Linq;
```

```
namespace Week2
{
    public static class Program
    {
        private static StreamReader sr;
        private static StreamWriter sw;
        private static int[] a;
        private static int n, k1, k2, A, B, C;

        public static void Quicksort(int[] elements, int left, int right)
        {
            while (true)
            {
                if (left > k2 || right < k1) return;

                int i = left, j = right;
                int pivot = elements[(left + right) / 2];

                while (i <= j)
                {
                    while (elements[i].CompareTo(pivot) < 0)
                    {
```

```
        i++;
    }

    while (elements[j].CompareTo(pivot) > 0)
    {
        j--;
    }

    if (i > j) continue;

    int tmp = elements[i];
    elements[i] = elements[j];
    elements[j] = tmp;

    i++;
    j--;
}

if (left < j)
{
    Quicksort(elements, left, j);
}
```

```
        if (i < right)
        {
            left = i;
            continue;
        }

        break;
    }
}

public static void Main(string[] args)
{
    sr = new StreamReader("input.txt");
    sw = new StreamWriter("output.txt", false);

    var inp = sr.ReadLine().Split().Select(int.Parse).ToArray();
    n = inp[0];
    k1 = inp[1] - 1;
    k2 = inp[2] - 1;

    inp = sr.ReadLine().Split().Select(int.Parse).ToArray();

    A = inp[0];
```

```
            B = inp[1];
            C = inp[2];
            a = new int[n];
            a[0] = inp[3];
            a[1] = inp[4];
            sr.Close();

            for (var i = 2; i < n; i++)
            {
                a[i] = A * a[i - 2] + B * a[i - 1] + C;
            }

            Quicksort(a, 0, a.Length - 1);
            for (int i = k1; i <= k2; i++)
            {
                sw.Write($"{a[i]} ");
            }

            sw.Close();
        }
    }
}
```

Задание №5

Сортировка пугалом

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

«Сортировка пугалом» — это давно забытая народная потешка, которую восстановили по летописям специалисты платформы «Открытое образование» специально для этого курса.

Участнику под верхнюю одежду продевают деревянную палку, так что у него оказываются растопырены руки, как у огородного пугала. Перед ним ставятся n матрёшек в ряд. Из-за палки единственное, что он может сделать — это взять в руки две матрешки на расстоянии k друг от друга (то есть i-ую и $(i+k)$-ую), развернуться и поставить их обратно в ряд, таким образом поменяв их местами.

Задача участника — расположить матрёшки по неубыванию размера. Может ли он это сделать?

Формат входного файла

В первой строчке содержатся числа n и k $(1 \le n, k \le 10^5)$ — число матрёшек и размах рук.

Во второй строчке содержится n целых чисел, которые по модулю не превосходят 10^9 — размеры матрёшек.

Формат выходного файла

Выведите «YES», если возможно отсортировать матрёшки по неубыванию размера, и «NO» в противном случае.

№ теста	Результат	Время, с	Память	Размер входного файла	Размер выходного файла
Max		0.312	25583616	1039313	3
1	OK	0.031	9027584	12	2
2	OK	0.046	8962048	16	3
3	OK	0.031	9060352	112	3
4	OK	0.031	9019392	111	2
5	OK	0.031	8986624	112	3
6	OK	0.031	8957952	112	2
7	OK	0.031	9031680	109	3
8	OK	0.062	9023488	112	2
9	OK	0.031	9015296	110	3
10	OK	0.031	9007104	111	2
11	OK	0.031	9015296	108	3
12	OK	0.031	9158656	11674	3
13	OK	0.031	9076736	11707	2
14	OK	0.031	9195520	11712	3
15	OK	0.031	9175040	11754	2
16	OK	0.031	9191424	11708	3
17	OK	0.031	9183232	11740	2
18	OK	0.031	9179136	11726	3
19	OK	0.046	9195520	11680	2
20	OK	0.031	9166848	11741	3
21	OK	0.046	10399744	128736	3
22	OK	0.046	10379264	128832	2
23	OK	0.093	10416128	128751	3
24	OK	0.062	10936320	128866	2
25	OK	0.078	10792960	128700	3
26	OK	0.046	10477568	128707	2
27	OK	0.093	10387456	128729	3
28	OK	0.062	10416128	128807	2
29	OK	0.062	10342400	128784	3
30	OK	0.218	22044672	1039313	3
31	OK	0.171	21995520	1038610	2
32	OK	0.218	21999616	1038875	3
33	OK	0.296	25264128	1038723	2
34	OK	0.312	25583616	1038749	3
35	OK	0.171	22282240	1038747	2
36	OK	0.218	22077440	1039043	3
37	OK	0.187	22233088	1039210	2
38	OK	0.265	22278144	1038967	3

```
inputFile = open("input.txt", "r")

n, k = [int(x) for x in inputFile.readline().split()]

lst = [int(x) for x in inputFile.readline().split()]
```

```
arrays = []
for i in range(k):
    temporary_array = []
    for j in range(i, n, k):
        temporary_array.append(lst[j])
    temporary_array.sort()
    arrays.append(temporary_array)
result = "YES"


for i in range(1, n):
    if arrays[i % k][i // k] < arrays[(i - 1) % k][(i - 1) // k]:
        result = "NO"
        break



file = open("output.txt", "w")
file.write(result)
```