**Министерство образования и науки Российской Федерации**

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,   
МЕХАНИКИ И ОПТИКИ**

Факультет программной инженерии и компьютерной техники

Кафедра информатики и прикладной математики   
Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Алгоритмы и структуры данных»

**ОТЧЁТ**

по лабораторной работе №3  
неделя третья

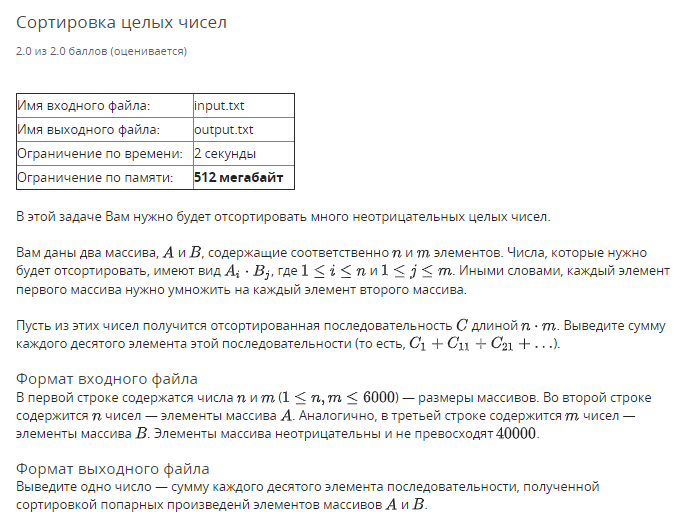
Выполнил:  
Айгузин Иван Олегович   
P3218

Преподаватели:

Романов Алексей Андреевич  
Волчек Дмитрий Геннадьевич

Санкт-Петербург

2018



using System;

using System.IO;

using System.Linq;

namespace Task01 {

public sealed class Program {

private const int Base = 256;

private static StreamReader \_in;

private static StreamWriter \_out;

private static void Main() {

\_in = new StreamReader("input.txt");

\_out = new StreamWriter("output.txt");

Run();

\_in?.Dispose();

\_out?.Dispose();

}

private static void Run() {

var (n, m, arrayA, arrayB) = GetInitValues();

var array = new int[n \* m];

var max = 0;

var cursor = 0;

for (var i = 0; i < arrayA.Length; i++) {

var a = arrayA[i];

for (var j = 0; j < arrayB.Length; j++) {

var c = a \* arrayB[j];

array[cursor++] = c;

if (c > max) {

max = c;

}

}

}

var output = new int[array.Length];

var count = new int[Base];

for (var pow = 0; 1L << pow <= max; pow += 8) {

Array.Clear(count, 0, Base);

for (var i = 0; i < array.Length; i++) {

var t = array[i];

count[(t >> pow) & 255]++;

}

for (var i = 1; i < Base; i++) {

count[i] += count[i - 1];

}

for (var i = array.Length - 1; i >= 0; i--) {

var a = array[i];

var index = (a >> pow) & 255;

output[--count[index]] = a;

}

for (var i = 0; i < array.Length; i++) {

array[i] = output[i];

}

}

long sum = 0;

for (var i = 0; i < array.Length; i += 10) {

sum += array[i];

}

\_out?.WriteLine(sum);

}

private static (int, int, int[], int[]) GetInitValues() {

var (n, m) = ReadFirstLine();

var arrayA = ReadIntList();

var arrayB = ReadIntList();

return (n, m, arrayA, arrayB);

}

private static (int N, int K) ReadFirstLine() {

var p = ReadIntList();

return (p[0], p[1]);

}

private static int[] ReadIntList() {

return \_in.ReadLine()

.Split(' ')

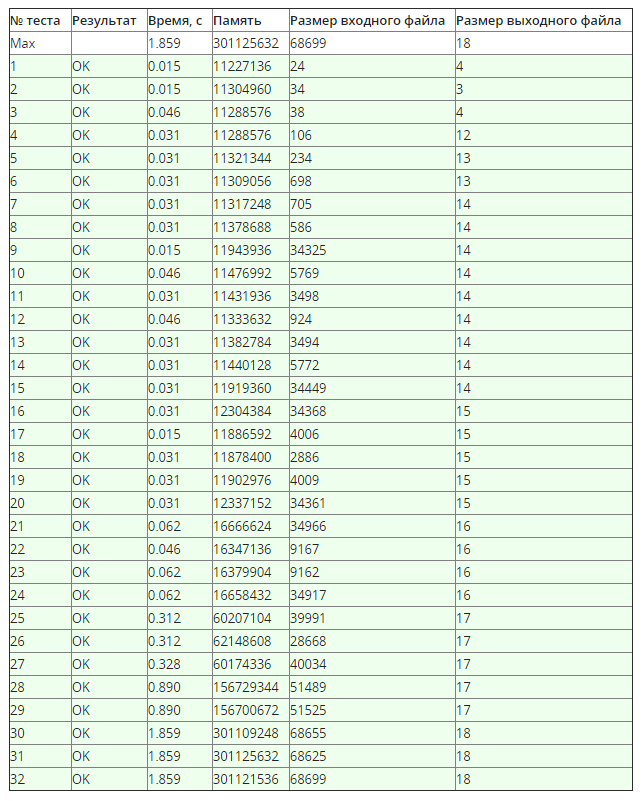
.Select(int.Parse)

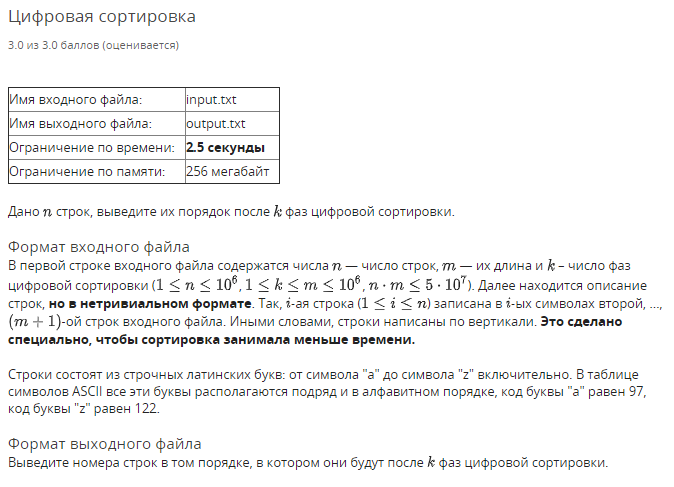
.ToArray();

}

}

}





using System;

using System.IO;

using System.Linq;

namespace Task02 {

public sealed class Program {

private const int Base = 26;

private static StreamReader In;

private static StreamWriter Out;

private static readonly int NewLineLength = Environment.NewLine.Length;

private static void Main(string[] args) {

if (!args.Contains("console1")) {

SetupIO();

}

Run();

DisposeIO();

}

private static void Run() {

var (n, len, k, buffer) = GetInitValues();

// Radix sort

var indexer = Enumerable.Range(0, n).ToArray();

var output = new int[n];

var count = new int[Base];

for (var phase = 1; phase <= k; phase++) {

Array.Clear(count, 0, Base);

var offset = (len - phase) \* n;

for (var i = 0; i < n; i++) {

count[buffer[offset + i]]++;

}

for (var i = 1; i < Base; i++) {

count[i] += count[i - 1];

}

for (var i = n - 1; i >= 0; i--) {

var a = buffer[offset + indexer[i]];

output[--count[a]] = indexer[i];

}

for (var i = 0; i < n; i++) {

indexer[i] = output[i];

}

}

//

for (var i = 0; i < indexer.Length; i++) {

Out.Write($"{indexer[i] + 1} ");

}

Out.WriteLine();

}

private static (int n, int m, int k, byte[] buffer) GetInitValues() {

// var random = new Random(1000);

// var bytes = Enumerable.Range(0, 200)

// .Select(x => (byte) random.Next(0, 25))

// .ToArray();

//

// return (20, 10, 10, bytes);

var firstLine = In.ReadLine();

var p = firstLine

.Split(' ')

.Select(int.Parse)

.ToList();

var (n, m, k) = (p[0], p[1], p[2]);

var stream = In.BaseStream;

stream.Seek(firstLine.Length + NewLineLength, SeekOrigin.Begin);

var buffer = ReadStrings(n, m, k);

return (n, m, k, buffer);

}

private static byte[] ReadStrings(int n, int m, int k) {

var stream = In.BaseStream;

var bufferLength = n \* m;

var buffer = new byte[bufferLength];

for (var i = 0; i < m; i++) {

stream.Read(buffer, i \* n, n);

stream.Seek(NewLineLength, SeekOrigin.Current);

}

for (var i = 0; i < bufferLength; i++) {

buffer[i] -= 0x61; // 'a'

}

return buffer;

}

private static void SetupIO() {

In = new StreamReader("input.txt");

Out = new StreamWriter("output.txt");

}

private static void DisposeIO() {

In?.Dispose();

Out?.Dispose();

}

}

}

