Assignment 2

Deadline: Sunday, October 4, 23:59

Instruction

This assignment is backed by Chapter 14. You have the maximum control of the way you program, but:

- Your code should be well-written in terms of the design (careful considerations and bug-freeness) and efficiency (few duplicates).
- The program must be adequately commented and follow a coding style. See the "Introductory video" on Canvas to find references for basic rules of writing comments and programming style.

Auto-testing with CodeGrade

- We are experimenting with an auto-testing tool, called "CodeGrade", to grade this second set of assignments.
- This tool will be used to support the task of the Teaching Assistants (TAs) that will grade your assignments. We will not use auto-testing to provide immediate feedback. You will only see the results of the auto-testing once you receive your grade. You therefore have to thoroughly test your code yourself.
- The auto-testing tool will help the TAs to evaluate the 'solution correctness' of your code, that is, whether your code work according to the specification in the assignment.
- To make auto-testing feasible you should follow strictly the guidelines for preparing your code, as follows.

How it works

Let us first explain how these types of tests work in CodeGrade. Auto-testing runs tests on student work to help teachers grade assignments and to give students more feedback. Input/Output tests simply compare the output of the students code given some input to the expected output. If the output and expected output match, the test case is passed. Input is supplied via program arguments. The input can for example be the name of a .txt file that is stored on CodeGrade. The output is everything that is printed while running the students code. The expected output should preferably be short.

An example

We created a test assignment on CodeGrade to show how this works. The test is that students write a program “Question1.java” that accepts two input arguments. The first input argument is a filename. The corresponding file contains a single integer. The second input argument is an integer. Question1.java should read the file with the integer, add the other integer and print out the result. Here is an example of a working implementation:

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;
```

```java
public class Question1 {
    public static void main(String[] args) throws FileNotFoundException {
        // TODO Auto-generated method stub
        String testFileName = args[0];
        String testString = args[1];

        Scanner s = new Scanner(new File(testFileName));
        int firstInteger = Integer.parseInt(s.nextLine());
        s.close();

        int secondInteger = Integer.parseInt(testString);

        int output = firstInteger+secondInteger;
        System.out.println(output);
    }
}
```

To test this code, we upload a number of .txt files containing integers to CodeGrade. Then, we design tests that call the student's code with certain inputs, and compare the student's output to the expected output.

Guidelines for preparing your code for CodeGrade

- Each question below specifies the name of the Main Class of your program. A Main Class in Java contains the main method. In the example above, the name of the Main Class is "Question1".
- Each question below specifies the input and output formats of your program. Your program should strictly follow these formats.
- Input should be passed using program arguments. For example, for Question 1 below, the program should take two input arguments. The first argument is the name of the file that contains the list with countries (which not necessarily needs to be called “countries.txt”) and the second argument is the name of a specific country.
- You should only print the output as specified in the assignment, and nothing else.
- It is not necessary to program everything in the main method/class. You can program additional classes and methods. Indeed, as 'design' is one of those evaluation criteria, your program in general should have several methods and possibly several classes.

Question 1

We first do a basic sorting and searching exercise. In the file "countries.txt", you find a shuffled list of the names of 196 countries/regions. First, implement a selection sort algorithm to sort the list in A-Z order and write a binary search algorithm to search for a specific country/region.

The name of your Main Class should be Question1

Input format

A total of 2 input arguments. The first argument is the name of the file that contains the list with countries (each line in this file gives the name of a country/region) and the second argument is the name of a specific country/region. We have uploaded in Canvas a file called "countries.txt" that can be used as the first argument of your program.

Output format

Your program should print on the screen a total of 1 line. This line prints the position k in the sorted list of the country/region used as an input for the binary search algorithm. If the country/region used in the search is found, the position k should lie in the range [1, ..., 196], where 1 means found in the first position in the sorted list, 2 means found in the second position in the sorted list and so on. Otherwise, if the country/region used in the search is not found, -1 should be printed.

Question 2

The CPUs these days have multiple cores and we might benefit from multi-threading.

In this question, you are given a file "words.txt", in which there are 466,544 strings, each takes a separate line. Note that they are not really all words because you have things like "&c" or "1080". Also, you can assume that there are no duplicates.

First, write a usual merge sort algorithm to sort the strings, alphabetically; i.e.,

```
if (a.compareTo(b) > 0) {
    String temp = b;
    b = a;
    a = temp;
}
```

Then, write a multi-thread binary search to search for a specific string (let us call this the "needle"). The multi-thread search must be implemented as the following.

1. Split the sorted list into `n` parts
2. Have `n` threads, each search on one of the `n` parts for the needle.
3. Report the index if you find the needle. Here the index should be with respect to the full list, not a single part.

In the end, compare the multi-thread search with a single thread one, to see if there are any improvements (in terms of time).

Tips 1: You might want to read the Java tutorial on Threads and Synchronization.

Tips 2: If you write array-based algorithms, you may not see improvements at all. If you use arraylist-based ones, you may see dramatic improvements.

Tips 3: You can use the binary search algorithm you wrote for Question 1.

The name of your Main Class should be Question2

Input format

A total of 3 input arguments. The first argument is the name of the file that contains one string per line, the second argument is the name of a specific string (the "needle") and the third argument is the number of threads `n`. We have uploaded in Canvas a file called "words.txt" that can be used as the first argument of your program.

Output format

Your program should print on the screen a total of 2 lines. The first line prints the position k in the sorted list of the "needle" used as an input for the binary search algorithm. If the "needle" used in the search is found, the position k should lie in the range [1, ..., 466,544], where 1 means found in the first position in the sorted list, 2 found in the second position in the sorted list and so on. Otherwise, if the "needle" used in the search is not found, -1 should be printed. The second line

prints the total time in milliseconds (rounded to the nearest integer) that your program takes to run.

Question 3

In this exercise, you are given an array A of n distinct integers and are requested to rearrange the elements of array A such that to obtain a new array, namely B. This new array B has exactly the same elements from that of A and represents a permutation of the elements of A. Array B is a special permutation of the elements of A once it is a rearrangement of A that minimizes the sum of the absolute values of differences between its adjacent elements among all permutations of elements of A. In other words, array B is such that the sum of |B[i] - B[i-1]| for all 0 < i < n is minimal.

Of course, array B can be obtained from array A by a sequence of swaps. Your task is to write a program that returns the **minimum number of swaps** that should be performed in order to rearrange the elements of array A such that to obtain array B.

For example, suppose A = [5, 11, 9, 1]. One candidate to be array B (as defined above), can be obtained by performing the following swaps:

```
Swap      Result
      [5, 11, 9, 1]
1 5   [1, 11, 9, 5]
5 11  [1, 5, 9, 11]
```

That is, it took 2 swaps to transform A into B.

The name of your Main Class should be Question3

Input format

A total of 1 input argument. This argument is the name of a file that contains two lines. The first line contains a single integer, `n`, the number of elements in A. The second line contains `n` space-separated integers. We have uploaded in Canvas a file called "swaps.txt" that can be used as an example of an argument for your program. To test your program, we recommend that you create several other files following the file format specified for this question.

Output format

Your program should print on the screen a total of 1 line. This single line prints the minimum number of swaps needed to make array B.

Question 4

Given an array A of n integers and an integer k < n, write a program to select exactly k elements from array A such that the difference between the value of the maximum element selected and the value of the minimum element selected is minimum among all possible selections of k elements. The time complexity of your solution should be O(n * log n). **Important: the time complexity is part of the specification of this question. Therefore, a solution with worst time complexity than O(n * log n) will not be awarded full points for the 'solution correctness' criteria**.

Some examples of inputs and outputs for this question are shown below:

```
Input: A = [7, 3, 2, 4, 9, 12, 56], k = 3
Output: 2

The minimum difference is 2. If we select 2, 3 and 4, we obtain the minimum
difference between maximum (4) and minimum (2), which is 2, once 4 - 2 = 2.

Input: A = [3, 4, 1, 9, 56, 7, 9, 12], k = 5
Output: 6

The minimum difference is 6. If we select 3, 4, 7, 9 and 9, we obtain the
minimum
difference between maximum (9) and minimum (3), which is 6, once 9 - 3 = 7.
```

The name of your Main Class should be Question4

Input format

A total of 1 input argument. This argument is the name of a file that contains three lines. The first line contains a single integer, `n`, the number of elements in A. The second line contains a single integer single integer, `k`, the number of elements to be selected from A. The third line contains `n` space-separated integers. We have uploaded in Canvas a file called "mindiff.txt" that can be used as an example of an argument for your program. To test your program, we recommend that you create several other files following the file format specified for this question.

Output format

Your program should print on the screen a total of 1 line. This single line prints the minimum difference between maximum and minimum elements among all possible selections of k elements.

Question 5

Given an Array A of n integers, we say that a pair of elements A[i] and A[j] is a **bad pair** if A[i] > A[j] and i < j. In a sorted array, the number of bad pairs is 0. The number of bad pairs gives an indication of the computational effort to sort an array. Your task is to build a program to compute the number of bad pairs of a given array of n integers. The time complexity of your solution should be O(n * log n). **Important: the time complexity is part of the specification of this question. Therefore, a solution with worst time complexity than O(n * log n) will not be awarded full points for the 'solution correctness' criteria**. A *naïve* approach to this question is to explicitly compare each element with all the others and keep a count of the number of bad pairs found so far. This *naïve* approach is not a correct solution to this question, as its time complexity is significantly higher than the specified O(n * log n) time complexity.

Some examples of inputs and outputs for this question are shown below:

```
Input: A[] = [12, 5, 3, 1]
Output: 6

Given array has six bad pairs:
(12,5), (5,3),(12,3), (12,1), (5,1), (3,1).


Input: A[] = [8, 3, 5]
Output: 2

Given array has two bad pairs:
(8, 3), (8, 5)
```

The name of your Main Class should be Question5

Input format

A total of 1 input argument. This argument is the name of a file that contains two lines. The first line contains a single integer, `n`, the number of elements in A. The second line contains `n` space-separated integers. We have uploaded in Canvas a file called "badpairs.txt" that can be used as an example of an argument for your program. To test your program, we recommend that you create several other files following the file format specified for this question.

Output format

Your program should print on the screen a total of 1 line. This single line prints the number of bad pairs of array A.