HOMEWORK #3:

Four flavors of sort

Version 09-11-24

Due Date: Friday, November 15th, 11:59.59pm

Description:

Write a program that sorts a sequence of numbers using **four** different methods, and displays some statistics.

Methods:

Recall from the course notes the sorting algorithms Bubble Sort, Selection Sort, Insertion Sort and Merge Sort.

Input & Output:

Your program should read input from **standard input** (keyboard), and output to **standard output** (screen). Do **not prompt** for the input.
The first line of the input will consist of a single positive integer **N**. Then **N** positive integers follow, one per line, which correspond to the input array to sort.

The first line of the output should be the number of *key comparisons* followed by the number of *array swap* operations required by the Bubble Sort algorithm to sort the input array.

The second line of the output should be the number of *key comparisons* followed by the number of *array swap* operations required by the Selection Sort algorithm to sort the input array.

The third line of the output should be the number of *key comparisons* followed by the number of *array assignment* operations required by the Insertion Sort algorithm to sort the input array.

The fourth line of the output should be the number of *key comparisons* required by the Merge Sort algorithm to sort the input array. (these comparisons occur during the "merge" part of the algorithm)
Output **must be** in the **format** shown in the sample output.

Sample Input and Output:

(Other sample Input / Output pairs will be available in the repository)

Sample 0

Input (What the user will type)	**Output** (What your program should display)
7 11 15 8 3 5 2 13	21 14 21 6 17 20 14

Sample 1

Input (What the user will type)	**Output** (What your program should display)
16 68 45 96 84 91 91 14 57 10 32 70 58 23 59 89 43	120 70 120 15 82 85 44

Sample 2

Input (What the user will type)	**Output** (What your program should display)
20 28 11 12 13 14 25 16 17 18 19 20 21 22 23 24 15 26 27 10 29	190 54 190 19 71 73 61

Sample 3

Input (What the user will type)	**Output** (What your program should display)
32 97 10 54 57 54 46 95 27 84 64 33 34 41 71 24 56 32 71 29 92 32 81 28 96	496 264 496 31 294 295 127

64 26 20 83 26 64 96 19	

Submission and Grading:

Submit your assignment by placing all code/scripts in this assignment's git repository in the course's GitLab server, [link] . (You should have a repository setup by the end of this week).

Your main file shall be called "**sort4**" regardless of extension. (e.g. if you are programming in C++, your main file should be called "`sort4.cpp`". If you are programming in Java your main file should be called "`sort4.java`"). Your main file should include your **name**. Include any other necessary files in your submission. In order to accommodate different languages, your submission should include a bash script named '`run.sh`' that **compiles** and **runs** your program with all the necessary options and commands.

For example, the following is a possible '`run.sh`' script for **C++ 11**.

```
#!/bin/bash
g++ -std=c++11 *.cpp -o sort4.ex
./sort4.ex
```

An example '`run.sh`' script for **Python3**:

```
#!/bin/bash
python3 sort4.py3
```

An example '`run.sh`' script for **Java** if the program is called '`sort4.java`' :and the main class is called '`sort4`'.

```
#!/bin/bash
javac sort4.java
java sort4
```

Your program will be evaluated and graded on the **Computer Science department's Linux machines** so your program needs to be compatible with the current system, compilers and environment. Your program will be evaluated and graded using the command:

```
./run.sh < gradinginput.txt > gradingoutput.txt
```

IMPORTANT: Remember to make your '`run.sh`' file executable as a script with the UNIX command:

```
chmod +x run.sh
```

Also, if you develop '`run.sh`' on windows, make sure is in UNIX format with the command:

```
dos2unix run.sh
```