Middle East Technical University | Department of Computer Engineering

CENG 315

Algorithms

Fall 2024-2025

Take-Home Exam 0

Due date: 20 October 2024, Sunday, 23.59

Important note: This practice exam is adapted from older exams and will **not** be graded. Some items mentioned in the Specifications and Regulations sections are therefore invalid for this exam. They are left there so that you get a chance to read them before the THE1.

1 Problem Definition

In this exam, you are asked to complete the given function definition `insertionSort` to sort the given array `arr` in ascending order. Your function should also count the number of `comparisons` and `swaps` executed during this sorting process. Note that the comparisons are only between the values to be sorted, not your auxiliary comparisons.

```
void insertionSort(int* arr, long &comparison, long & swap, int size);
```

You can use the pseudocode given in Figure 1 for the base of your implementation, but different from the pseudocode, you are expected to count all the conditions in the inner while loop as comparisons. In other words, you are also expected to count the cases where you check if the index is 0.

```
i ← 1
while i < length(A)
    x ← A[i]
    j ← i - 1
    while j >= 0 and A[j] > x
        A[j+1] ← A[j]
        j ← j - 1
    end while
    A[j+1] ← x
    i ← i + 1
end while
```

Figure 1: Pseudocode for the implementation

2 Example I/O

Example 1

```
initial array = {9, -2, 3, 15} size=4
sorted array = {-2, 3, 9, 15}, comparison=5, swap=2
```

Example 2

```
initial array = {0, -5, -5, -5, 4, 1} size=6
sorted array = {-5, -5, -5, 0, 1, 4}, comparison=9, swap=4
```

Example 3

```
initial array = {1, 5, 8, 10, 11, 17, 22} size=7
sorted array = {1, 5, 8, 10, 11, 17, 22}, comparison=6, swap=0
```

3 Specifications

- You will implement your solutions in the `the0.cpp` file.
- You are free to add other functions to `the0.cpp`.
- Do not change the first line of `the0.cpp`, which is `#include "the0.h"`
- Do not change the arguments and the return value of the function `insertionSort()` in the file `the0.cpp`.
- Do not include any other library or write include anywhere in your `the0.cpp` file (not even in comments).
- You are given a `test.cpp` file to test your work on 0DTUCLASS or your locale. You can and you are encouraged to modify this file to add different test cases.
- You can test your `the0.cpp` on the virtual lab environment. If you click run, your function will be compiled and executed with `test.cpp`. If you click evaluate, you will get feedback for your current work and your work will be temporarily graded with a limited number of inputs.
- The grade you see in VPL is not your final grade, your code will be reevaluated with more inputs after the exam.
- If you want to test your work and see your outputs on your locale you can use the following commands:

```
> g++ test.cpp the0.cpp -Wall -std=c++11 -o test
> ./test
```

4 Constraints and Limits

- Maximum array size is 25000.
- The system has the following limits to test the complexity of your solutions:
  - a maximum execution time of 1 minute
  - a 256 MB maximum memory limit
  - a stack size of 64 MB for function calls (ie. recursive solutions)
- Solutions with longer running times will not be graded.
- If you are sure that your solution works in the expected complexity constraints but your evaluation fails due to limits in the lab environment, the constant factors may be the problem.
- If your solution is correct, the time and memory limits may be adjusted to accept your solution after the lab. Please send an email if that is the case for you.

5 Regulations

- **Implementation and Submission:** The template files are available in the Virtual Programming Lab (VPL) activity called "THE0" on ODTUCLASS. At this point, you have two options:
  - You can download the template files, complete the implementation, and test it with the given sample I/O on your local machine. Then submit the same file through this activity.
  - You can directly use the editor of the VPL environment by using the auto-evaluation feature of this activity interactively. Saving the code is equivalent to submitting a file.

  Please make sure that your code runs on ODTUCLASS. There is no limitation in running your code online. The last save/submission will determine your final grade.
- **Programming Language:** You must code your program in C++. Your submission will be tested on the VPL environment in ODTUCLASS, hence you are expected to make sure your code runs successfully there.
- **Cheating: This assignment is designed to be worked on individually.** Additionally, the use of any LLMs (chatgpt, copilot, the other one that you are thinking about...) and copying code directly from the internet for implementations is strictly forbidden. Your work will be evaluated for cheating, and disciplinary action may be taken if necessary.
- **Evaluation:** Your program will be evaluated automatically using "black-box" testing, so make sure to obey the specifications. No erroneous input will be used. Therefore, you don't have to worry about invalid cases.
  Important Note: The given sample I/O's are only to ease your debugging process and NOT official. Furthermore, it is not guaranteed that they cover all the cases of required functions. As a programmer, it is your responsibility to consider such extreme cases for the functions. Your implementations will be evaluated by the official test cases to determine your final grade after the deadline.