Sepcifications

- There is **1 task** to be solved in **12 hours** in this take home exam.
- 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 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 **Odtuclass** or your **locale**. You can and you are encouraged to modify this file to add different test cases.
- If you want to **test** your work and see your outputs you can **compile** your work on your locale as:

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

> ./test
```

- You can test your **the0.cpp** on virtual lab environment. If you click **run**, your function will be compiled and executed with test.cpp. If you click **evaluate**, you will get a feedback for your current work and your work will be **temporarly** graded for **limited** number of inputs.
- The grade you see in lab is **not** your final grade, your code will be reevaluated with more inputs after the exam.

The system have the following limits:

- 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)
- Each task has a complexity constraint explained in respective sections.
- Solutions with longer running times will not be graded.

- If you are sure that your solution works in the expected complexity constrains 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.

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

In this exam, you are asked to complete the function definition to sort the given array **arr** with **ascending** order. Your function should also count the number of **comparison** and **swap** executed during this sorting proccess (Comparisons are only between the values to be sorted only, not your auxiliary comparisons).

You can use the following pseudocode for the base of your insertionSort implementation:

```
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
```

Constraints:

- Maximum array size is 25000.

Evaluation:

- Since this take home exam is only for testing purposes, you will not be graded with the work you have done.

Example IO:

```
1)

initial array = {9, -2, 3, 15} size=4

sorted array = {-2, 3, 9, 15}, comparison=5, swap=2

2)

initial array = {0, -5, -5, -5, 4, 1} size=6

sorted array = {-5, -5, -5, 0, 1, 4}, comparison=9, swap=4

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
```