Exercise 1 - Complexity

What is the (asymptotic) running time of each of the following pseudo-codes or functions? Justify your answers.

a)

```
for i = 1 to n * n do
     for j = 1 to i do
        sequence of statements
     end for
end for
```

b)

```
for (i = 0; i < N; i++) {
      for (j = 0; j < N; j++) {
            sequence of statements
      }
}
for (k = 0; k < N; k++) {
      sequence of statements
}
```

c)

$15n + 2^n + 0.5n^3$

DELIVERY: pdf file with the answers.

Exercise 2 – Search algorithms

Write a program in C to solve the following problem. Given a static array with the following numbers:

```
int arr[10]={2,6,-1,4,-7,8,3,-3,2,-6};
```

Find all negative values, count them and replace them with their absolute value. You can use the `abs()` function:

`int abs(int x)` returns the absolute value of int x.

The example output of the program

```
The input array is: 2,6,-1,4,-7,8,3,-3,2,-6
The output array is: 2,6,1,4,7,8,3,3,2,6
There are 4 negative elements.
```

DELIVERY: source file .c with the implemented program.

Exercise 3 – Sort algorithms

We will work with 2 different algorithms: bubble sort and merge sort. We divide our program into different source files to manage better the implementation of the given problem. Having different source files for different parts of the program provides us with modularity and better files' management.

We provide you with the following files:

`bubble.h` - declaration of all functions (prototypes) related to the bubble sort algorithms

`bubble.c` - implementation of all functions related to the bubble sort algorithms

`merge.h` - declaration of all functions (prototypes) related to the merge sort algorithms

`merge.c` - implementation of all functions related to the merge sort algorithms

`sort.c` - main program

1. Analyze the provided code for two sort algorithms. Analyze which are the dependencies between these files.

Question 1: Briefly describe dependencies between source files.

2. Proceed with the compilation process: there are several ways to compile this program. We propose you two ways that you can do on the Terminal tab of VSC:
   - Step by step: first, compile all source files one by one (.c); second, link all the generated intermediate object files (.o) creating app.exe file. You can do it, for example, in the following way:

     ```
     gcc -c bubble.c

     gcc -c merge.c

     gcc -c sort.c

     gcc -o app.exe *.o
     ```

   - All in one: compile and link all files into one app.exe file. You can do it, for example, in the following way:

     ```
     gcc -o app.exe bubble.c merge.c sort.c
     ```

     or even like this:

     ```
     gcc -o app.exe *.c
     ```

Choose the way you prefer 😊

3. Analyze how to execute the program, i.e., which arguments it requires.

Question 2: How would you execute the program for sorting 10 numbers using bubble sort algorithm?

4. As you've seen, we left the bubble sort functionality without any implementation. Search for the correct place to implement the bubble sort algorithm. Implement your solution for this algorithm (we've seen it during the theoretical class).

5. Analyze the merge sort functionality. As you remember, this algorithm follows "divide and merge" approach.

Question 3: Which lines of the code provide the “divide” part of the algorithm? Which lines of the code provide the “merge” part of the algorithm?

6. Analyze the functions that we provide you for measuring time (in sort.c file). Use the function that provides the result in microseconds and measure the execution time of the bubble sorting function and the merge sorting function.

Question 4: Which is the return value (time unit) for each of the 3 provided functions for measuring time. How will you measure the execution time in your code (provide the corresponding line/lines)?

7. Execute different program configurations and measure execution time. Warning: to obtain clearer program output, you can comment printing arrays.

Question 5: Which is the execution time for the following cases:

Sort	100 numbers	1000 numbers	10000 numbers	100000 numbers	1000000 numbers
Bubble sort					
Merge sort					

DELIVERY: pdf file with the answers to the 5 presented questions.