ALGORITHMICS - Bachelor of Software Engineering, 2020

LAB GUIDE. SESSION 1.2

GOALS:

- **Examples of execution time measurements in iterative algorithms**
- **Implementation constant of two algorithms**

1. Some iterative models

`Loop1.java`, `Loop2.java`, `Loop3.java` and `Unkown.java` classes are provided to you. They are iterative models on which we must work to determine their time complexity.

Execute the programs with the Eclipse IDE and check the times for different sizes of the problem to conclude whether the times obtained are the expected considering the complexity of each program.

Carry out an analysis of the code of each file to raise its analytical complexity. Check the times for different sizes of the problem to conclude whether the obtained times are the expected ones of the complexity of each program.

2. Comparison of two algorithms

To compare the execution time of two algorithms (with each other) we will calculate the **division ratio** of the execution times that these algorithms take for the same size of the problem.

When the size of the problem grows and the algorithms to be compared have different complexity:

- If the ratio tends to **0** ➔ the one associated with the numerator is the least complex (that is, the best).
- If the ratio tends to **infinity** ➔ the one associated with the numerator is the most complex (that is, the worst).

When the size of the problem grows, and the algorithms have the same complexity, the ratio tends to a **constant**, called **implementation constant**. That constant is the one that tell us which one of the two algorithms with the same complexity is better:

- If the implementation constant < 1 ➔ It is better the one in the numerator.
- If the implementation constant > 1 ➔ It is better the one in the denominator.
- If the implementation constant = 1 ➔ The algorithms are exactly equal.

TO DO:

A. Work to be done

- An Eclipse `session1_2` **package** in your course project. The content of the package should be:
    - All the files that were given with the instructions for this session together with the implementation of `Loop4.java` and `Loop5.java`.
- A **PDF document** using the course template. The activities of the document should be the following:
    - **Activity 1. Two algorithms with the same complexity**
        - We are going to compare **loop2** and **loop3** algorithms and obtain the value for the division **loop2/loop3**. To that, you should fill in the following table (remember to include always in all columns the units of time and the CPU and RAM of the machine where the measurement was made). In addition, briefly explain if the results make sense from the point of view of the complexities of the algorithms.

N	*loop2(t)*	*loop3(t)*	*loop2(t)/loop3(t)*
8	…..	…..	…..
16	…..	…..	…..
32	…..	…..	…..
64	…..	…..	…..
128	…..	…..	…..
256	…..	…..	…..
512	…..	…..	…..
1024	…..	…..	…..
2048	…..	…..	…..
4096	…..	…..	…..
……	…..	…..	…..

    - **Activity 2. Two algorithms with different complexity**
        - We are going to compare **loop1** and **loop2** algorithms and obtain the value for the division **loop1/loop2**. To that, you should fill in the following table (remember to include always in all columns the units of time and the CPU and RAM of the machine where the measurement was made). In addition, briefly explain if the results make sense from the point of view of the complexities of the algorithms.

N	*loop1(t)*	*loop2(t)*	*loop1(t)/loop2(t)*
8	…..	…..	…..
16	…..	…..	…..
32	…..	…..	…..
64	…..	…..	…..
128	…..	…..	…..

256	…..	…..	…..
512	…..	…..	…..
1024	…..	…..	…..
2048	…..	…..	…..
4096	…..	…..	…..
……	…..	…..	…..

- **Activity 3. Complexity of other algorithms**
  - We are going to **create** and compare two new algorithms, **loop4** (it should have a $O(n^4)$ complexity) and **loop5** (it should have a $O(n^3 \log n)$ complexity) algorithms and obtain the value for the division ***loop4/loop5***. To that, you should fill in the following table (remember to include always in all columns the units of time and the CPU and RAM of the machine where the measurement was made). In addition, briefly explain if the results make sense from the point of view of the complexities of the algorithms.

N	*loop4(t)*	*loop5(t)*	*loop4(t)/loop5(t)*
8	…..	…..	…..
16	…..	…..	…..
32	…..	…..	…..
64	…..	…..	…..
128	…..	…..	…..
256	…..	…..	…..
512	…..	…..	…..
1024	…..	…..	…..
2048	…..	…..	…..
4096	…..	…..	…..
……	…..	…..	…..

- **Activity 4. Study of `Unknown.java`**
  - You should create another table with execution times for different sizes of the problem for the method contained in `Unknown.java` together with a brief explanation of its complexity. Does it make sense according to the theoretical complexity? Use the formula explained in class to calculate the expected execution time when the size of the problem changes and compare it with the time you took empirically. Do it twice.

B. Delivery method

You should **commit and push** your project with your new `session1_2` package in your Github repository with the following content inside it:

- All the requested source files.
- The requested PDF document called `session1_2.pdf` with the corresponding activities.

Important:

- Make sure your Github course project is up to date after the delivery and that the last commit is strictly before the deadline. If not, the mark for this session will be 0 without any exception.
- Your mark will be eventually included in the `Marks.xlsx` file in your repository. Please, do not modify anything in that file to avoid conflicts between versions.

Deadlines:

- Group L.I-01 (Thursday): February 19, 2020 at 11:55pm.
- Group L.I-02 (Wednesday): February 18, 2020 at 11:55pm.
- Group L.I-03 (Tuesday): February 17, 2020 at 11:55pm.
- Group L.I-04 (Monday): February 23, 2020 at 11:55pm.