LE-2
Due Date: Feb 16th 2024
Algorithmic Complexity

Description:
The lab exercise LE2 is a hands-on component for the concepts covered in class lectures on analysis of algorithms. LE2 is intended to give us the opportunity to visualize growth rate and confirm the algorithmic complexity of functional code snippets provided with the LE. There are two expectations associated with LE2:

1. **Examine the code for each working algorithm** (coded in C++) to develop a theoretical understanding of the algorithm growth rate and corresponding Big-O runtime complexity. You are NOT required to solve mathematical expressions; simply a high level assessment based on study of critical code blocks will be sufficient. Associated with each algorithm are multiple-choice questions in Canvas Quiz that present an array of possibilities. Pick the correct one for each algorithm.
2. **Run the code file and for each algorithm, analyze the respective plots of execution time** for the general trend representing the growth rate and verify the correctness of your submitted answers in (1) above. A basic plotting code is also provided, you may change the values and experiment. The characterization of the growth trends may also be done by creating your own growth rate functions for the most common time complexities and comparing the output of the algorithms with them.

Starter Code:
You are provided with the le2_algo.cpp file containing functional C++ code for five algorithms. You will need to execute this file as per the directions given to verify the complexities assessed using the theoretical framework. The code file le2_algo.cpp and the corresponding Makefile is provided in the Canvas quiz named LE2.

How to compile and run le2_algo.cpp:
Once you have downloaded both the le2_algo.cpp and the Makefile from Canvas:

- Navigate to the directory in which you have downloaded le2_algo.cpp and the Makefile
- Run the command **make** on the terminal (which will compile le2_algo.cpp and create the executable le2_algo)
- Run **./le2_algo** to obtain the resultant graphs of each of the 5 algorithms (algo1, algo2, algo3, algo4, algo5)

[Advisory]:

- **The unit of measurement for each plot is described in the respective header of the algorithm's plot**
    - **Plots for algo1, algo2, algo4, and algo5 are measured in microseconds**
    - **The plot for algo3 is measured in nanoseconds**

Task:
You will be graded for correctness of the case-matching question uploaded on Canvas quiz linked to LE2. In addition, you are expected to submit a PDF document describing your approach in (1) above. This should particularly include the analysis of the code snippets to estimate the expected growth rate (before running the code) and which parts of the code would suggest/contribute to that. Include a snapshot of the graphs (multiple snapshots of the same graph is acceptable in case a single snapshot is not enough to capture the whole graph) collected as a result of running the code for the respective algorithms. Specifically,

1. Complete the quiz LE2
2. Upload the PDF document in the quiz containing short explanations for the algorithmic complexities along with the graphs.
3. The quiz will have two attempts in case you are not able to finalize your answers in the first attempt. Please make sure to refer to the question, and answer the questions within the two attempts.