# CSCE 350: Project

Late Submissions will <u>not</u> be accepted

**Task - 1:** (50 points) **Implement the Quick-Sort Algorithm using C++**

**Instructions:**

<u>Algorithm:</u>

- Pick a random pivot value
- Swap the chosen pivot with the left-most value in the list
- Sort the values using the QuickSort algorithm
  - pseudo-code can be found in the course textbook on Pg. 176-178
- Record the time taken to sort the values using QuickSort

<u>Input:</u>

- Your program should be able to read an input ASCII file that contains a list of unsorted floating-point numbers separated by a blank space

<u>Output:</u>

- Your program should produce
  - an output ASCII file that contains sorted floating-point numbers separated by a blank space
- an ASCII file that contains the execution time in milliseconds
- *See Task-2 for additional details*

<u>Compile:</u>

- Include a makefile with instructions to compile your code.
  - For Example: Both the input and output filenames can be passed as command line arguments.
  - Compile your program using the following command:
    - *yourLastname_yourFirstname_QuickSort   input.txt   output.txt*

---

**Task – 2:** (50 points) **Empirical Analysis of Algorithm using C++**

1. Study the time complexity of QuickSort using different input sizes: 10, 100, 1000 (the

number of unsorted floating-point numbers).

**Instructions:**

- Input File Generator:
  - Write C++ program to randomly generate 25 input files for each input size.
  - You can use any uniform random number generator to create an input file.
- Run your program (QuickSort.cpp from Task-1) on each input file.
- Record the below values separated by a tab(4 spaces) in an output ASCII file named *yourLastname_yourFirstname_executionTime.txt*:
  - input size and
  - the execution time
  - For example:

| *Input Size* | *Execution Time* |
|---|---|
| 10 | 4e-09 |
| 10 | 3e-09 |
| … | |
| 100 | # |
| 100 | # |
| … | |

- Compute the average running time for each input size and store these averages in an ASCII file, named *yourLastname_yourFirstname_averageExecutionTime.txt* containing the average execution times for each input size.
- For example:

| *Input Size* | *Average Execution Time* |
|---|---|
| 10 | 2e-09 |
| 100 | 4e-09 |
| 1000 | … |

- Plot:
  - Show the average execution times in a plot, where X-axis represents the input size and the Y-axis represents the time.

    - You will have a curve for QuickSort, where a point on the curve represents the average execution time for an input size.
    - Save the plot into a file named "*yourLastname_yourFirstname_plotAverageExecutionTime.jpg*"

**Instructions:**

- All code should be written in C++ for Linux.
- Program file submissions that do not compile automatically receive a grade of 0.
- Please test your code on the Departmental Linux machines prior to submission on BB.
- Code must commented appropriately for major steps.

**Submission:**

- You must submit all your generated files and plots.
- Your zip folder must contain the following files:

1. *QuickSort.cpp
2. *executionTime.txt
3. *averageExecutionTime.txt
4. *plotAverageExecutionTime.jpg
5. Makefile
6. InputFileGenerator.cpp → used to generate input ASCII files
7. A ReadMe file that includes instructions on compiling your project

   *File Naming Convention: All your files(1-4 in the above list) should have the following prefix: yourLastName_yourFirstName_

- Compress all your files into a single folder titled *"CSCE350Project_yourLastname_yourFirstname"* and submit it on Blackboard by **11:59 pm Friday, 2 December 2022**
- Accepted Compression Formats: .tar.gz/ .zip only