Programming Assignment 1
CSCE 350 : Data Structures and Algorithms
Spring 2023
Due Date - March 21

Instructions:

- The solutions should be very clear and should follow the instructions below and the requirements stated for each problem.
- If you refer to any resource to get your solutions, add an acknowledgement with your solutions (details of the source, e.g., book, website, etc.).
- All the codes should be written in c or $c++$ for linux and commented appropriately for major steps/functions.
- Code that does not compile will not be graded and get a 0 automatically.
- The codes should be submitted as a single zipped file through Blackboard

Part A:

1. A network topology specifies how computers, printers and other devices are connected over a network. The problem 5 in your HW 3 introduced three common topologies of networks: the ring, the star, and the fully connected mesh. You are given a boolean matrix $A[0, \ldots, n-1, 0, \ldots, n-1]$, where $n > 3$, which is supposed to be the adjacency matrix of a graph modeling a network with one of these topologies. Your task is to determine which of these three topologies, if any, the matrix represents. Design a brute-force algorithm for this task and indicate its time efficiency class. Implement the code for checking if the topology of the given input graph is a star, ring or Algorithm using C or $C++$. (50 pts)

   Requirements:

   (a) Your code should be able to read an input ASCII file named 'input.txt' which contains the adjacency matrix

   (b) Your code will produce an output ASCII file named 'output.txt' contains the outcome of your code, which is printed text of the name of the topology

   (c) Your code should output the execution time for running the algorithm excluding time of input/output.

   (d) A script file or readme file including the instructions to compile and run the code should be submitted together with the codes

2. Implement the Quick-Sort Algorithm using C or $C++$. (50 pts)

   Requirements:

   (a) Your code should be able to read an input ASCII file named 'input.txt' which contains the unsorted float-point numbers separated by blank space

   (b) Your code will produce an output ASCII file named 'output.txt' contains the sorted float-point numbers separated by blank space

   (c) Your code should output the execution time for running Quicksort excluding time of input/output.

   (d) A script file or readme file including the instructions to compile and run the code should be submitted together with the codes

Bonus question: Empirical Analysis of Algorithm using C or C++. (20 pts)

1. Write a C or C++ code to study time complexity of QuickSort using different input size 10, 100, 1000, 10000, and 100000 (the number of unsorted float-point numbers). (10 pts)

   Requirements:

   (a) For each input size, you need to generate 100 input files randomly. You can use any random number generator to create an input file.

   (b) For each input file, run QuickSort and record the execution time for running Quicksort excluding time of input/output.

   (c) Compute the average running time for each input size

   (d) A script file or readme file including the instructions to compile and run the code should be submitted together with the codes

2. Show the average running 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 representing the average running time for an input size. You can draw the plot manually or using any software. (10 pts)