ITU Computer Engineering Department
BLG335E, Analysis of Algorithms I, Fall 2016
Project 1

Due Date: 21.10.2016 23:00

Problem: In this project, you are expected to find closest **K** warehouse locations among **N** warehouse locations to a given location considering their coordinates on x-y plane.

A warehouse location is represented with its x and y axis. Each warehouse has a unique identification number.

You are required to implement **merge sort, insertion sort and linear search algorithms** in order to find closest **K** warehouse locations to a given x-y coordinate among **N** locations.

A data set (warehouselocations.txt) is provided that includes unique ids, x-axis and y-axis information on each line respectively. In order to calculate the distance between two locations, you may use the euclidean distance.

Your program should be run from the command line with the following format.
./program N K algorithmType(IS,MS or LS) x-axis y-axis
Sample: ./program 100000 10 MS 1234 5678

N : Total number of locations to be sorted / searched (10, 100, 1000, 1000000). You may use the first 10, 100, 1000 locations as a subset of locations.
K : Number of closest locations to be found (1, 2, 10, N/2)
algorithmType : Method to be used to solve the problem (Insertion Sort(IS), Merge Sort(MS) or Linear Search(LS))
x-axis: X-axis information of the point that closest K location will be found
y-axis: Y-axis information of the point that closest K location will be found

After execution of your program, an output file (“output.txt”) should be created that lists K warehouses with their ids, x-axis, y-axis and the euclidean distance (to the given point (x,y)) information like the input file’s format.

In your report, you are expected to analyze and compare the running times of algorithms with respect to their computational complexity.

a. Give the asymptotic upper bound on the running time for linear search, insertion sort and merge sort and show that your implementation of these algorithms fit these values.

b. Run each search methods for each different value of **N** as {10, 100, 1000, 1000000} and **K** as {1, 2, 10, N/2} calculate the average time of execution for each value of **N**.

Hint: You can use the clock() function under ctime library for calculating time of execution for the search functions. Refer to http://www.cplusplus.com/reference/clibrary/ctime/clock for more details.

c. After calculating execution times you will prepare two line plots in Excel in order to visualize the runtime complexity of linear search, merge sort and insertion sort for different values of **N** and **K** (For example, you may prepare a separate plot in order to illustrate the runtime of the three algorithms for different **N** values for **K** = 1. Similarly, you may prepare 3 more plots for K = 2, 10 and N/2. This will result with 4 different plots). Then you are expected to interpret the results with respect to the asymptotic upper bounds you have given in ***a***. Indicate in which cases you would choose which algorithm. Why?

Additional Notes:

1. Submissions will be done through the Ninova server. You must submit all your source code in a single cpp file and a softcopy report (PDF).
2. Make sure that GNU C++ compiler (g++) compiles your project, and the application runs in Linux smoothly. You can use ITU ssh server to compile and test your application. This is important because we will evaluate your homework in Unix using g++.
3. Use comments wherever necessary in your code to explain what you did.
4. You are not allowed to use the standard template library (STL).

Note: If you have any questions, please feel free to contact Res. Asst. Çağatay Koç via e-mail (kocca@itu.edu.tr).

Academic dishonesty including but not limited to cheating, plagiarism, collaboration is unacceptable and subject to disciplinary actions.