BLG 335E - Analysis of Algorithms I, Fall 2017

Project 2

Total worth: 7.5% of your grade

Handed out: 19 Oct 2017, Thursday

Due: 09 Nov 2017 Thursday - 23:59

Overview

Assume that you are working in the data mining department of a civil registry and you need to process big data that includes statistical information. Before applying data mining techniques to the data, you need to apply some preprocessing operations. You are provided a data set (*population_by_zip2010.csv*) that includes the information about the residences in 2010 in the US. In this file, every single line (except the first one, since it is the header) belongs to a distinct residence. The following information is provided in the data set:

- Population: The population for that segment
- Minimum age (can be null): Minimum age in the range
- Maximum age (can be null): Maximum age in the range
- Gender (can be null)
- Zip code: Zip code of that residence
- Geo_id: Geo code of that residence

The department wants to have this data sorted in this manner: The data should be sorted considering the population. However, if two residence has the same number of habitants, geo_id should be considered. If both parts match, the behavior of your algorithm should decide which residence should appear first.

Code (50 points)

You are required to implement Quicksort algorithm by considering the constraints described above for this problem. Your program should be executed from the command line as follows with the following parameters:

./yourExecutableName N

N: number of the residences to be processed (You can just take the first N entries from the file)

After the execution, a message including the elapsed time should be printed out, and an output file that includes the sorted data with the same format of the input file should be produced.

Report (50 points)

a. (10 points) Give the asymptotic upper bound on the running time for Quicksort. If there is a recurrence for the algorithm, give and solve it.

b. (10 points) Run your program for **N** as {10, 100, 1000, 10000, 100000, 500000, 1000000}, and calculate the average time of execution (for example run 10 times for *N = 10* and take the average, 10 times for *N = 100* and take the average, and so on…) for each value of **N**. After calculating average execution times, you are required to prepare a two-lined plot in order to visualize the run time complexity of Quicksort for different values of **N.** Comment on the results by considering the asymptotic bound that you have found in ***(a)***.
Note: 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. (20 points) What is the worst case for Quicksort? Construct a data set in order to simulate this case, and execute your program for different values of N as {10, 100, 1000, 10000, 100000, 500000, 1000000}. Report the average execution times, and introduce a plot like you have done in part ***(b)***. Introduce a solution in order to overcome the worst case of Quicksort (You do not need to implement that solution).

d. (10 points) Is Quicksort stable? Explain your answer by illustrating your answer with a small fraction from the data set. You may modify the values if necessary.

Submission

You should be aware that the Ninova system clock may not be synchronized with your computer, watch, or cell phone. Do not e-mail the teaching assistant or the instructors your submission after the Ninova submission has closed. If you have submitted to Ninova once and want to make any changes to your report, you should do it before the Ninova submission system closes. **Your changes will not be accepted by e-mail**. Connectivity problems to the Internet or to Ninova in the last few minutes are not valid excuses for being unable to submit. **You should not risk leaving your submission to the last few minutes**. After uploading to Ninova, check to make sure that your submission appears there.

Policy: You may discuss the problem addressed by the project at an abstract level with your classmates, but you should not share or copy code from your classmates or from the Internet. **You should submit your own, individual project.** Plagiarism and any other forms of cheating will have serious consequences, including failing the course.

Submission Instructions: Please submit your assignment through Ninova. Include both your report (as a PDF file) and your code (including header files) in the archive file you submit.

All your code must be written in C++, and must compile and run on the common ITU Linux Server (accessible through SSH) using g++. If your code requires non-standard compiling, please state compilation instructions in your report.

When you write your code, try to follow an object-oriented methodology with well-chosen variable, method, and class names and comments where necessary**. Your code must compile without any errors; otherwise, you may get a grade of zero on the coding part of the assignment.**

If a question is not clear, please let the teaching assistant know by email (daltan@itu.edu.tr).