Analysis of Algorithms

Programming Project 1 – Divide and Conquer

In this project, our input is an array of doubles, where each entry A[i] denotes the price of a stock (or some other asset) on the ith day. So for example, if A = [5, 2, 3, 6, 1, 3] then this means that the stock in question was worth 5 on day 0, it was worth 3 on day 2, etc. We will consider buying the stock once and then later selling it once (of course we must sell the stock after we have bought it). If we buy the stock on buyDay and we sell the stock on sellDay, then the goal is to maximize our profit A[sellDay] – A[buyDay]. In the example above, the optimal solution is for buyDay to be 1 (buying at the price A[1] = 2) and sellDay to be 3 (selling at the price A[3] = 6). This gives us a profit of 4.

Stating the parameters formally:

- We must sell on a different day than we buy, and we must sell after we buy. In other words, we must have buyDay < sellDay.
- This means that if A has <= 1 number in it, there is no feasible solution. This will affect the base case of your algorithm!
- The return value should be the optimal profit (you don't need to return the exact days you are buying and selling). So in the example above, we would return 4.
- Your output should print the following to standard output (e.g., using printf() or System.out.println(), etc.):
    - *The optimal solution for input.txt is X.*
    - Here, input.txt should be whatever the name of the given input file is, and X should be the actual optimal solution.

This problem is not hard to get a solution that works using two nested for-loops (it runs in time O(n^2)). We are wanting to improve this to a divide and conquer algorithm that runs in O(n*log n) time. **If you do not give a divide and conquer algorithm for this project, you will get a 0. The entire point of this project is to design a divide and conquer algorithm, not just to get some solution that works.**

How to code this project:

For the most part, you can code this project in any language you want provided that the language is supported on the Fox servers at UTSA. We want you to use a language that you are very comfortable with so that implementation issues do not prevent you from accomplishing the project. That said, we will be compiling and running your code on the Fox servers, so you need to pick a language that is already installed on those machines. See the PDFs on Blackboard in the Programming Projects folder for more information on how to connect to the Fox servers remotely (also covered in the Programming Project 1 overview lecture).

Since everyone is coding in their own preferred language, we are asking you to provide a bash script named *project1.sh* that will act similarly to a makefile. I covered how bash scripts work in class in the Project 1 overview lecture, and I recorded a short follow-up to this here: https://youtu.be/CaIFJWiyU_U

In short, your bash script should contain the command to compile your code, and then on a different line, it should contain the line to execute your code. There are several input files for you to test your code on, and the bash script should take a command line argument specifying which input file to run on. So, the command to execute your code should look like this:

bash project1.sh input.txt

See the YouTube video that I linked for more details on how to use command line arguments inside a bash script.

Files provided in the project:

Since you are programming in different languages, we are providing no source files for your code. We have provided a blank project1.sh file for you to fill in as well as several input files for you to test your code on. Each input file contains a single price on each line of the file. The first line should be index 0 of your array, the second line should be index 1, etc. We will test your code will all the privided input files, but we will also be testing with some different input files that will be in the same format. Some of these files may be very large (hundreds of thousands of entries).

Grading

We will grade according to the rubric provided on Blackboard. The majority of the points come from the following: did the student give a divide and conquer algorithm that runs in O(n*log n) time that returns the correct answer for all possible inputs? Proper documentation may help the grader understand your code and earn you partial credit in the event you have some mistakes in the code.

Violations of the UTSA Student Code of Conduct will be penalized harshly. In particular, be very careful about sending code to a student who asks how you accomplished a particular task. I've heard this story several times recently: "They said they just wanted to see how to perform part X of the project. I didn't think they would submit my exact code." If this happens, you will both be penalized for cheating. To protect yourself and to more properly help your fellow student, send pseudocode, and not actual compilable code.

Also we know about the online sites where people upload projects and have a third party complete the project for you. This is a particularly egregious form of cheating (it's in the best interest of your career to not tolerate this). If you use a solution from one of these sites or submit a minor modification (minor is at the disgression of the instructor) of a solution from one of these sites, you will receive a 0 and will be reported to the university for a viloation of the UTSA Student Code of Conduct.

Submitting

Zip up your project folder and submit on the dropbox on Blackboard by the due date.