HW_2 Data Structures

All code MUST be commented. Every question should have a “white paper” (a pdf) explaining the concept of how it works. All work should be posted to github.

1. Comment ALL of the code in the file “stack linked list.txt” and “postfix.txt”
2. Write a prefix stack and show how it works (in main).
3. Test a stack's speed (with millions of operations) as to which is faster: an array based stack or a linked list based stack.
4. Create your own stack functions to do the following:
    a. List all the elements in the stack
    b. Iterate through the stack and change one of the values based on its position
5. Use an array based stack to solve the “longest increasing subsequence problem” (https://www.techiedelight.com/longest-increasing-subsequence/)
6. Use a stack to pass in a string and reverse each word in the string. For instance, if you have the string “*When Chuck Norris falls in water, Chuck Norris doesn't get wet. Water gets Chuck Norris.*” It should now say, “*nehW kcuhC sirroN sllaf ni retaw, …*” That is, it should take in one string and be a function (or multiple functions) that use a stack (or multiple stacks) to reverse each word in the sentence. Create a text file that you input of Chuck Norris jokes and have the program reverse each of the jokes. [Here is some help if you want ot teg detracts: https://www.techiedelight.com/reverse-string-without-using-recursion/
7. Use a stack to solve a maze
8. Company X generally processes orders based on a first come, first serve basis. Sometimes a client will ask for rush processing and will be awarded a number based on the urgency. If they pay extra they will receive a 3 priority, if they have a bulk order they will receive a 2 priority, and if they know someone in the company and ask for a favor they will receive the highest priority, a 1. Design a system that accepts orders from clients and processes these orders based on no priority or somewhere on the priority scale. Show how the system will work for random orders with a random processing time.
9. An airport has four runways, each pointing in a different direction. Planes are assigned to a runway based on both the time they come in and their point of departure. 70% of planes can only go in one direction, 20% of planes can leave in one of two directions and will always choose whichever has the shorter line, and 10% of planes can leave in one of three directions and will always choose the shorter line. The “airport man” wants to have two systems displayed before him, one that shows all four departure runways and their planes and another system that shows one departure calendar that combines all four runways. Create and model this with an example running in main.