Homework 2 (Posted 5th February, Due during or before class 12th February)

Policy for Programming Assignment: Problem 4 has a programming assignment. I would like to decouple the design from the programming part. If you are not sure of the solution for Problem 4, you can wait till 15th February, midnight to submit the program. The solution for the design part will be posted 12th February. You have 3 more days for this program. You can certainly submit your program on or before 12th, but if you want to wait please email the T.A. indicating your intention. You must email before 12th midnight (email must have a subject WAIT). All other solutions including the design part of problem 4 are due during or before class 12th February. If your name is John Smith, then name your program as JohnSmith.c and email it to yjkim78@gradient.cis.upenn.edu.

Problem 1: (Grade 2.5 + 5 pts) You have seen the implementation of a stack using an array. In the implementation shown in class, you add the first element at position 0, next element at position 1, etc. You always add and delete elements from the end of the list. Consider an array of size n. Design a stack implementation where you insert the first element at position n, the second at position n-1 etc. From which end do you add and delete elements?

Implement two stacks using one array of size n. You should be able to push and pop from both the stacks in constant time. Your stack overflows if total size of both stacks exceed n. Your algorithm should detect overflow when it happens and subsequently terminate. (You only need to give the logical steps for the implementation. No program is required.)

Problem 2: Grade 5 pts You have a character string as input to your algorithm. You need to find out whether the string is of the form a C b. Here C is the letter C, a and b are sequences of As and Bs, and a and b must be mirror images of each other, e.g., a = ABAAB, b = BAABA. For full grade you must give an O(N) algorithm where N is the number of alphabets in the input string.

Problem 3: Grade 10 pts You have two sorted lists, L_1, L_2. You know the lengths of each list, L_1 has length N_1 and L_2 has length N_2. Design algorithms to output a sorted list $L_1 \cap L_2$. You need to give two algorithms. One should have complexity $O(N_1 + N_2)$ and another should have complexity $O\left(\min(N_1, N_2)\log\left(\max(N_1, N_2)\right)\right)$. Which (if any) is faster? (There are three possibilities here, first algorithm is faster for all N_1, N_2, second algorithm is faster for all N_1, N_2, or the answer depends on the values of N_1 N_2.) Justify your answer.

Problem 4: Grade 7.5 + 10 pts This problem requires you to implement a special stack. The special stack does the usual push and pop in constant times. In addition, it must find the minimum value of elements currently in stack in constant time (find-min operation).

Write a program to implement this stack. Someone who uses this program must be able to push, pop, and find-min any number of times. Your program should ask the user whether it wants to push, pop, or find-min. The user would enter his choice, and then the value for push if he wants to push. For the other two operations, the program outputs the popped or find-min values.