CS2223
HW#1

DUE: Frisday, November 2

1 (2 points) Do **PROBLEM 1.11** from our text.

2. (6 points) Given $n \in \mathbb{N}^+$ and permutation $(a_1, ..., a_n)$ of $(1, ..., n)$, we want an algorithm to obtain $(a_1, ..., a_n)$ from $(1, ..., n)$. Your algorithm accepts as input $(1, ..., n)$ and the only data structure your algorithm can use is a *stack*. It must start and finish with an empty stack. The operations it can perform at any time are:

- S - remove the next element from the input and push it onto the stack
- X - pop the top element from the stack and print it.

Your algorithm will thus be a sequence of *n S*s and *n X*s. For example, an algorithm to obtain $(2,4,3,1)$ is *SSXSSXXX*.

a An algorithm is *admissible* if it never tries to pop from an empty stack and it finishes with an empty stack. Describe a linear time algorithm to test if a sequence of *S*s and *X*s is admissible.

b A permutation $(a_1, ..., a_n)$ is *realizable* if there is an admissible algorithm which prints it. For example, algorithm *SSXSSXXX* shows that $(2,4,3,1)$ is realizable. Show that $(3,5,7,6,8,4,9,2,10,1)$ is realizable but $(3,1,2)$ is not.

c What can you say about the order of the elements in the stack at any point in a computation?

EXTRA CREDIT: (4 points) Describe a predicate $P(a_i, a_j, a_k)$ such that the following THEOREM holds.

THEOREM: A permutation $(a_1, ..., a_n)$ is realizable if and only if there do not exist $1 \le i < j < k \le n$ such that $P(a_i, a_j, a_k)$.

3. (4 points) Adapted from Manber's *Introduction to Algorithms*) Suppose that we change the coins in the American monetary system to have five types of coins: a 15¢ piece, a 23¢ piece, a 29¢ piece, a 41¢ piece and a 67¢ piece. Write a program to find a combination of these coins (if there is one) which makes up \$18.08 = 1808¢. Submit a listing of your program along with its output.
(Later in the course we will seek an algorithm to solve a generalization of this problem, the *Knapsack Problem*.)