CSC 225 FALL 2015
ALGORITHMS AND DATA STRUCTURES I
ASSIGNMENT 2
UNIVERSITY OF VICTORIA

1. Solve Problems 1.3.3 on Page 161 and 1.3.13 on Page 162 of the textbook.

2. Show the various steps of Insertion-Sort, Merge-Sort and Quick-Sort on the example array, $\{5, 7, 0, 3, 4, 2, 6, 1\}$. For Quick-Sort, assume that the first element of the array is picked as pivot.

3. In any array A, an *inversion* is a pair of entries that are out of order in A. That is, an inversion is a pair (i, j) such that $i < j$ and $A[i] > A[j]$. Develop a algorithm for computing the number of inversions in a given array. The running time of your algorithm should be $O(n + k)$ where k is the number of inversions in the input array.

4. Consider an implementation of a stack using an extendible array. That is, instead of giving up with a "StackFullException" when the stack becomes full, we replace the current array S of size N with a larger one of size $f(N)$ and continue processing the push operations. Suppose that we are given two possible choices to increase the size of the array: (1) $f(N) = N + c$ (for convenience, we start with an initial array of size 0) (2) $f(N) = 2N$ (we start with an initial array of size 1). Compare the two strategies and decide which one is better.

   To analyse the two choices, assume the following cost model: A "regular" push operation costs one unit of time. A "special" push operation, when the current stack is full, costs $f(N) + N + 1$ units of time. That is, we assume a cost of $f(N)$ units to create the new array, N units of time to copy the N elements and one unit of time to copy the new element.

5. The span s_i of a stock's price on a certain day i is the maximum number of consecutive days (up to and including the current day) that the price of the stock has been less than or equal to its price on day i. You are given as input an array P of size n containing numbers such that $P[i]$ is the price of the stock on day i. Your goal is to output an array S of size n such that $S[i]$ is the span of the stock on day i.

   Observe that the naive algorithm for this problem runs in time $O(n^2)$. Show how to use a stack to design an algorithm to compute the span in $O(n)$ time. Describe your algorithm using pseudo-code and explain why its running time is $O(n)$.