Data structures and algorithms

1) Consider a vector V[1..N] which contains N positive integer entries.

a. [25 points] Use pseudo-code to describe an algorithm j=FindLargest(V) which returns L as the *index* of the largest entry of vector V (without modifying its contents) with complexity O(N).

b. [25 points] Use pseudo-code to describe an algorithm S=Sort(V) which returns a sorted vector S[1..N] and uses FindLargest(V) as a function to assist in the sorting of the entries in V. You can use temporary variables and vectors. What is the complexity of your sort algorithm?

c. [50 points] The use of heap data structures can lead to faster sorting. Consider a heap data structure implemented as a binary tree.

   i. What is the property that must be satisfied for every node "i" (except the root node) of the binary tree for it to be a heap?

   ii. Consider the implementation of FindLargest using a heap: p=FindLargest(H) returns a *pointer* to the heap node which stores the largest element of the heap (without modifying its contents). Can this algorithm achieve lower complexity than the algorithm in part a)?

   iii. Describe how the Sort() algorithm would need to be modified to work with a heap data structure and the p=FindLargest(H) function. What is the complexity of the modified Sort() algorithm?

Operating Systems

2) This question considers a typical UNIX-like multi-user, multi-task operating systems which employs virtual memory, processes, and preemptive CPU schedulers to support time-sharing of resources among multiple tasks. In answering the five parts of this question, state your assumptions if necessary.

i) [20 points] Describe two major constituents of the context of a running process.

ii) [20 points] Consider a process “fork” event. Describe when and how the different constituents of the context of the parent process are copied to the child process.

iii) [20 points] Describe one hardware event and one software event that can trigger the execution of the O/S kernel scheduler and a context switch.

iv) [20 points] Why does an O/S kernel flush the contents of the translation look-aside buffer when a process context switch is performed?

v) [20 points] Why is it typically faster for the O/S kernel to switch among threads of a process than it is to switch process contexts?