Introduction to algorithms

Please write the solutions followed by each problems with blue color font.
Due date 5/17/2020.
Email for submission: 1535164365@qq.com (TA: Jerry)

1. Solve the following recurrence relation and give a Θ bound for each them.
(1) $T(n) = 2T(n/3) + 1$
(2) $T(n) = 4T(n/2) + n^2$
(3) $T(n) = 8T(n/3) + n^3$
(4) $T(n) = T(n-1) + 2$
(5) $T(n) = 2T(n/3) + nlogn$

2. Suppose you are choosing between the following three algorithms:
Algorithm A solves problems by dividing them into five subproblems of half the size, recursively solving each subproblem, and then combining the solutions in linear time.
Algorithm B solves problems of size n by recursively solving two subproblems of size n - 1 and then combining the solutions in constant time.
Algorithm C solves problems of size n by dividing them into nine subproblems of size n/3, recursively solving each subproblem, and then combining the solutions in O(n^2) time.

What are the running times of each these algorithms(in big-O) notation), and which would you choose?

3. A *k-way* merge operation. Suppose you have *k* sorted arrays, each with *n* elements, and you want to combine them into a single sorted array of *kn* elements.
(1) Here is one strategy: merge the first two arrays, then merge in the third, and merge in the fourth, and so on. What is the time complexity of this algorithm in terms of *k* and *n* ?
(2) Give a more efficient solution to this problem using divide and conquer, and what is the complexity ?

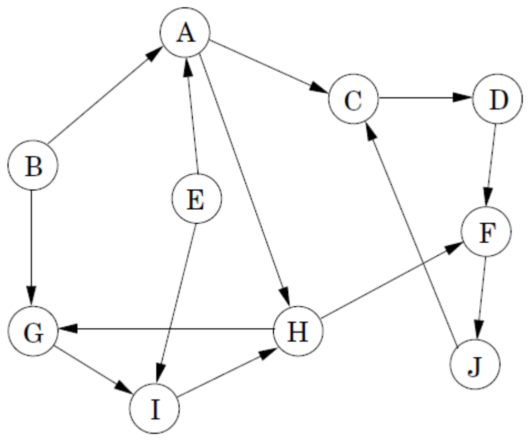
4. Given a sorted array of distinct integers $A[1, \cdots, n]$, you want to find out whether there is an index *i* for which $A[i] = i$. Given a divide and conquer algorithm that runs in time *O(*log*n)*.

5. Perform depth-first search of the following graphs whenever there is a choice of vertices, pick the one that is alphabetically first.
(1) Give the pre and post number of each vertex. Classify each edges as a tree edge, forward edge, back edege or cross edge.

Introduction to algorithms

(2) When doing DFS on G^R, what are the strongly connected components found ?

(3) Draw the "metagraph"



6. Consider the following knapsack problem instance in dynamic programming:

(1) Show the tables for knapsack with repetition and for the knapsack without repetition.

(2) Write the algorithm of knapsack without repetition using dynamic programming.

Item	Weight	Value
1	6	\$30
2	3	\$14
3	4	\$16
4	2	\$9

7. Palindromic Subsequence. A subsequence is *palindromic* if it is the same whether read left to right or right to left. For instance, the sequence A,C,G,T,G,T,C,A,A,A,A,T,C,G has many palindromic subsequences, including A,C,G,C,A and A,A,A,A (on the other hand, the subsequence A,C,T is *not* palindromic).

(1) Devise an algorithm in a pseudocode that takes a sequence $x[1,\cdots,n]$ and returns the length of the longest palindromic subsequence using a 2-dimensional table. Its running time should be $O(n^2)$.

(2) Show its running time is $O(n^2)$.