C4-R4: ADVANCED ALGORITHM

NOTE:

1.	**Answer question 1 and any FOUR from questions 2 to 7.**
2.	**Parts of the same question should be answered together and in the same sequence.**

Time: 3 Hours **Total Marks: 100**

1.

a) What are algorithms? What is the smallest value of *n* such that an algorithm whose running time is $100n^2$ runs faster than an algorithm whose running time is 2^n on the same machine?

b) Draw a recursion tree for the recurrence relation
T(n) = c if n= 1 and
= T(n/2)+cn if n > 1
Show that T(n) = O(n lgn)

c) The fractional problem can be solved by greedy technique while the *0-1* problem cannot be solved with a greedy approach. Why? Justify your answer.

d) Is P = NP? Justify your answer.

e) What does dynamic programming have in common with divide-and-conquer, and what is the principal difference between the two techniques?

f) Give a counter example to the conjecture that if there is a path from *u* to *v* in a directed graph *G*, then any depth-first search must result in $d[v] \leq f[u]$.

g) Show the comparisons the naive string matcher makes for the pattern P=0001 in the text T=000010001010001.

(7x4)

2.

a) Is Kruskal's algorithm greedy? Why?

b) Develop an algorithm and a recursive function to calculate GCD of two integers a & b.

c) How can the optimal solution to the 0-1 knapsack problem be found with Dynamic Programming?

(3+8+7)

3.

a) Write short note on Approximation algorithms.

b) A sequence of *n* operations is performed on a data structure. The i^{th} operation costs *i* when *i* is an exact power of *2*, and *1* otherwise. Use aggregate analysis to determine the amortized cost per operation.

c) Which function is more efficient (below mentioned) and Why? What is the running time of TreeSearch() function?

A.
```
TreeSearch(x, k)
If (x = NULL or k = key[x])
    Return x;
    If (k < key[x])
        Return TreeSearch(left[x], k);
    else
        return TreeSearch(right[x], k);
```

B.
```
TreeSearch(x, k)
while (x != NULL and k !=key[x])
    if (k < key[x])
        x = left[x];
    else
        x = right[x];
return x;
```

(6+6+6)

4.

a) Apply quick sort algorithm to sort the list {E,X,A,M,P,L,E} in alphabetical order. Draw the tree of recursive calls made. Consider last element as a pivot element.

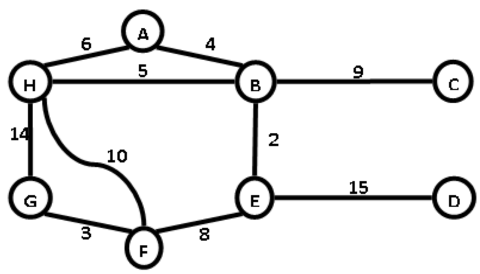
b) Find an optimal parenthesization of a matrix-chain product of A1, A2, A3 and A4, whose sequence of dimension is <5, 10, 3, 12, 5>.

(9+9)

5.

a) If P ≠ NP then prove that for any constant ρ ≥ 1, there is no polynomial-time approximation algorithm with approximation ratio ρ for the general traveling-salesman problem.

b) List out the various properties for the Shortest Path. Which edges form the minimum spanning tree (MST) of the shown below graph?



(9+9)

6.

a) How can Longest Common Subsequence problem be solved using dynamic programming?

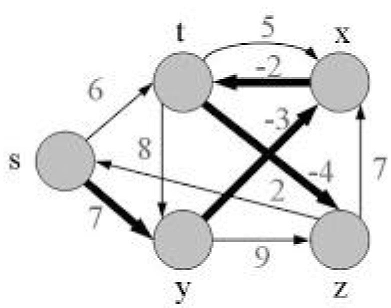
b) Construct the string matching automata for the pattern P = ababa.

c) How would you extend the Rabin-Karp method to the problem of searching a text string for an occurrence of any one of a given set of k patterns? Start by assuming that all k patterns have the same length. Then generalize your solution to allow the patterns to have different lengths.

(6+4+8)

7.

a) Describe the Bellman-Ford algorithm and execute this algorithm for the graph of 5 vertices given below:



b) What is a matching problem? Show that the matching M is maximum if and only if there is no augmenting path with respect to M.

c) Suppose that a counter begins at a number with b 1's in its binary representation, rather than at *0*. Show that the cost of performing n INCREMENT operations is $O(n)$ if $n = \Omega(b)$. (Do not assume that b is constant.)

(6+6+6)