Ref. No.: Ex/IT/PC/B/T/321/2022

B.E. INFORMATION TECHNOLOGY THIRD YEAR SECOND SEMESTER EXAM 2022

DESIGN & ANALYSIS OF ALGORITHM

Time: Three Hours **Full Marks: 100**

Different parts of the same question should be answered together

CO1 (20)

1. Answer any *one* question

A. (a) Given a min-heap, give an algorithm for finding the maximum element. Compute Complexity of Your Algorithm. (5)

(b) Find the complexity of the below recurrence (5)

$$T(n) = \begin{cases} 3T(n-1), if\ n > 0, \\ 1, \quad otherwise \end{cases}$$

(c) What is the running time of the following function? (4)

```
void Function(int n) {
        int i=1, s=1;
        while( s <= n) {
                i++;
                s= s+i;
                printf("*");
        }
}
```

(d) Consider a list of cities c1; c2,...,cn. Assume that we have a relation R such that, for any i,j, R(ci,cj) is 1 if cities ci and cj are in the same state, and 0 otherwise. If is stored as a table, how much space does it require? Justify your answer (6)

B. (a) Give an efficient algorithm for merging two binary max-heaps. Let us assume that the size of the first heap is m + n and the size of the second heap is n. Compute Complexity of your Algorithm. (10+2 =12)

(b) Find the complexity of the below recurrence: (8)

$$T(n) = \begin{cases} 2T(n-1) - 1, & if\ n > 0, \\ 1, & otherwise \end{cases}$$

CO2 (10)

2. (a) "The greedy approach always gives an optimal solution" the statement is true or false? Justify your answer with-example. (4)

(b) What is advantage of dynamic programming over divide and conquer approach? (2)

(c) Differentiate between backtracking and branch & bound algorithm strategy with examples. (4)

CO3 (30)

3. Answer *any* **three** questions

A. Given a sorted array of non-repeated integers A[1.. n], check whether there is an index i for which A[i] = i. Give a divide-and-conquer algorithm that runs in time O(logn). (10)

B. Give an algorithm to find out if an integer is a square? Compute Complexity of Your Algorithm. (8+2=10)

C.

```
int f(int n) {
    int sum = 0;
    if(n==0 || n==1) //Base Case
        return 2;
    //recursive case
    for(int i=1; i < n;i++)
        sum += 2 * f(i) * f(i-1);
    return sum;
}
```

(i) What will be output of the above code?
(ii) Compute Complexity.
(iii) Improve complexity using Dynamic Programming. (4+2+4=10)

D. Describe how backtracking is applied to solve 4- Queen Problem.
Solve following fractional Knapsack Problem using greedy approach where m=16, n=6, P=(2, 6, 8, 1, 3, 5), W=(10, 6, 5, 3, 1, 3) (3+7=10)

CO4 (15)

4. (a) What are the differences between linear search and binary search? Ho can you search an element from a dataset in constant time? (2+1=3)

(b) Derive the average case time complexity for quicksort algorithm. When do you call a sorting algorithm stable or unstable? Explain it with example. (5+3=8)

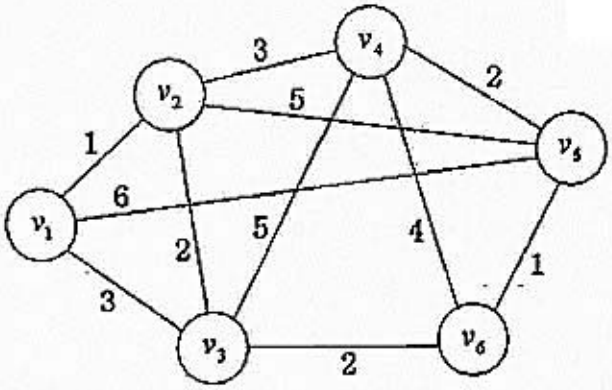
(c) The keys 12, 18, 13, 2, 3, 23, 5 and 15 are inserted into an initially empty hash table of length 10 using open addressing with hash function h(k) = k mod 10 and linear probing. What is the resultant hash table? (4)

CO5 (10)

5. Answer *any* **one** question

A. Develop an Algorithm to Detect Cycle in a Graph. Illustrate with an example. Compute Complexity. (6+2+2=10)

B. Implement Disjoint Set Concept to Compute the Minimum Spanning tree (MST) of the given graph. Derive the time complexity.. (8+2=10)



CO6 (15)

6. (a) How do you convert an optimization problem to a decision problem? Explain with an example. Define NP-Hard and NP Complete Problem. (3+2=5)

   (b) Represent complete sub graph problem (CSP) as 3-SAT problem with an example. In which class this problem belongs to and why?.(6+3=10)

CO1: Recollect notations for algorithm analysis and basic data structures and assess the performance of the associated operations (K3)

CO2: Illustrate and sketch different algorithmic paradigms to solve problems and analyze them (K3)

CO3: Apply different algorithmic techniques to solve the problems (K4)

CO4: Analyze, compare and differentiate the behavior of sorting/searching algorithms under different cases and solve the problem. (K4)

CO5: Analyze, compare and distinguish the different graph and geometric algorithms and solve problems (K4)

CO6: Describe and express the concept of NP-completeness and Approximation algorithms. (K2)