(b) Suppose we implement a k-bit binary counter using an array of size k. Determine the amortized cost per operation of a sequence of n increment operations using aggregate method. (4)

[This question paper contains 8 printed pages.]

Your Roll No..............

Sr. No. of Question Paper : 2944 **H**

Unique Paper Code : 32341401

Name of the Paper : Design and Analysis of Algorithms

Name of the Course : **B.Sc. (H) Computer Science**

Semester : IV

Duration : 3 Hours Maximum Marks : 75

Instructions for Candidates

1. Write your Roll No. on the top immediately on receipt of this question paper.
2. Question No. **1** is compulsory.
3. Attempt any **four** of Questions Nos. **2** to **7**.

SECTION A

1. (a) Can an input that leads to the best-case behaviour for one sorting algorithm lead to the worst-case behaviour for another algorithm? Explain with a suitable example. (3)

(b) How many times will the print statement in the following code snippet be executed? (3)

```
i = 1
while i*i <= n
    print("Hello")
    i += 1
```

(c) Comment on the correctness of the following implementation of radix sort : (3)

```
for i = 1 to d
    use heap sort to sort array A on digit i
```

(d is the number of digits in each element)

(d) Discuss the time complexity of the following steps in the Merge sort algorithm : (3)

(i) Divide

(ii) Combine

(b) Give an example graph with 5 nodes that gives two different MSTs using Kruskal's Algorithm and Prim's Algorithm. (4)

6. (a) Let G=(V,E) be a graph where V is the set of vertices and E is the set of edges. BFS is performed starting at node s, which produces layers $L_1, L_2, \ldots L_m$. There is an edge e = (u, v) in E such that $u \in L_1$ and $v \in L_i$ for some i. Show that G is not a bipartite graph. (6)

(b) For what values of n are cycle graphs C_n bipartite? Justify. (4)

7. (a) For the Interval Scheduling Problem, consider the greedy strategy of selecting first the request that starts last, i.e., the request for which s(i) is as large as possible. Prove that this greedy approach always finds an optimal solution to the problem. (6)

4. (a) Solve the **Subset Sum Problem** using dynamic programming with W = 17, and 4 items with weights $w_1 = 4$, $w_2 = 2$, $w_3 = 9$, $w_4 = 6$. Also, give the recursive formula. (6)

(b) Give the condition(s) under which the **0-1 Knapsack problem** becomes equivalent to the subset sum problem. How would you modify the recursive formula for the subset sum problem to solve 0-1 knapsack problem? (4)

5. (a) Let G(V, E) be a graph where V is the set of vertices and E is the set of edges. Let c(e) be the cost of edge e ε E and T be a Minimum Spanning Tree (MST) of G, with cost C_T. Will the MST of the graph G change if the cost of every edge is modified as follows :

(i) 1 – c(e)

(ii) $c(e)^2$

Explain your answer. (6)

(e) What are the two principles of Dynamic Programming? (3)

(f) Prove that any comparison based sorting algorithm requires Ω(n log n) comparisons in the worst case. (4)

(g) Consider two vertices x and y in a directed graph G(V, E). Let G_x and G_y be the strongly connected components containing x and y respectively such that there exists a node u ε $G_x \cap G_y$. Show that G_x and G_y are identical. (4)

(h) Give an efficient algorithm to check if a given undirected graph has a cycle. Discuss the time complexity of your algorithm. (4)

(i) Does the Dijkstra's algorithm for the shortest path problem work for graphs with negative edge weights? Justify your claim. (4)

(j) Write an algorithm to sort 4 keys in 5 comparisons in the worst case. (4)

SECTION B

2. (a) Consider the following algorithm and write a recurrence relation for its time complexity : (6)

```
ter_search(sorted list x of n elements, t)
    Let p and q be the elements that divide the list into
    3 equal parts
    if t = x[p] return p
    else if t < x[p] then ter-search(first third of the list),
    else if t = x[q] return q
    else if t < x[q] then ter-search(second third of the list).
    else ter-search(last third of the list)
```

Solve the recurrence relation and find out the time complexity.

(b) Using the **Median of Medians** method, find the 5^{th} smallest element in the following array :

[48, 43, 38, 33, 28, 23, 18, 13, 8, 49, 44, 39, 34, 29, 24, 19, 14, 9, 50, 45] (4)

3. (a) What will be the time complexity of the following sorting algorithms on an almost-sorted input?

   (i) Heap Sort

   (ii) Insertion Sort (6)

(b) Is it possible for the randomized version of **Quick sort** to show its worst-case behaviour on the following list of numbers?

[1, 2, 3, 4, 5, 6, 7]

If yes, indicate a sequence of pivots that may cause such behaviour. And if not, justify. (4)