United International **University**

Department of Computer Science and Engineering

Course: CSI 227 Algorithms | Trimester: Spring 2019

Final Exam | Marks: 100 | Time: 2 hours

There are FOUR questions. Answer ALL questions.

1. a) For a graph $G = (V, E)$, write down algorithms that **7 × 2**

   i) finds in-degrees and out-degrees of all vertices of G, when G is *directed* and represented by the *Adjacency Lists*;

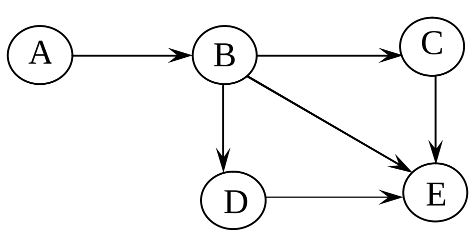
   ii) finds the maximum degree Δ and the minimum degree δ of G, when G is *undirected* and represented by the *Adjacency Matrix.*

   b) For a disconnected graph $G = (V, E)$, write an algorithm that finds the number of components of G, when G is *undirected* and represented by the *Adjacency Lists*. Analyze the running time of your algorithm. **7 + 4**

2. a) Dijkstra's algorithm for the shortest path problem is shown below. What would be the time-complexity of the algorithm (show line by line analysis) for an input graph G if *adjacency list* is used for graph representation and *an array* is used for storing `d[]` values. **10**

```
Dijkstra(G)
    Q = V[G];
    for each u ∈ Q
        d[v] = ∞;
    d[s] = 0; S = ∅;
    while (Q ≠ ∅)
        u = ExtractMin(Q);
        S = S U {u};
        for each v ∈ u->Adj[]
            if(v ∈ Q and d[v] > d[u]+w(u,v))
                d[v] = d[u]+w(u,v);
```

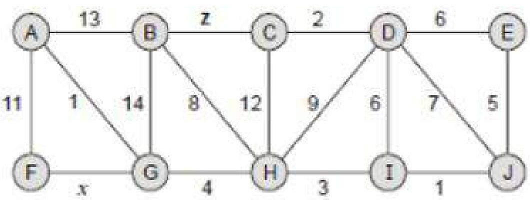
   b) Consider the graph shown below. Assuming A as the source vertex, assign weights to each of the edges such that the Dijkstra's algorithm always fails in this graph. Draw the wrong shortest-path tree of the Dijkstra's algorithm, and also draw the correct shortest-path tree. **8**



   c) Write an algorithm that can always find the shortest paths (if exist) in a graph with positive and negative edge weights. Analyze the running time of your algorithm. **7**

3. a) Write the Kruskal's algorithm that finds a minimum spanning tree of a graph. Analyze the running time of the algorithm. **10**

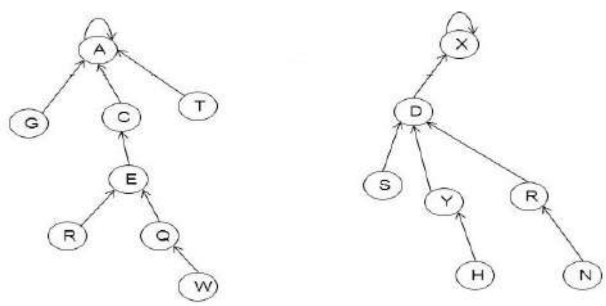
b) Draw a minimum spanning tree of the graph G shown below such that the minimum spanning tree always consists the edges with weights x and z. **6**



c) Briefly explain how probe sequences are generated by Linear Probing, Quadratic Probing, and Double Hashing techniques of open addressing. **9**

4. a) Write algorithms for UNION( ) and MAKE-SET( ) operations of ***Disjoint-Set Forest*** data structure assuming *union-by-rank* and the *path-compression* heuristics. **5**

b) What would the resultant forest be after calling UNION(W, Y) on the disjoint-sets forest of the following figure? You must use the *union-by-rank* and the *path-compression* heuristics. **5**



c) Consider an open-addressing hash table as shown below. The table already contains four data items. Assume that collisions are handled by the hash function **5**

$$h(k, i) = (\boldsymbol{h'}(\boldsymbol{k}) + i\,\boldsymbol{h_2}(\boldsymbol{k})) \bmod 13, \text{ where } \boldsymbol{h'}(\boldsymbol{k}) = (2k + 7) \bmod 13 \text{ and } \boldsymbol{h_2}(\boldsymbol{k}) = (k + 5) \bmod 13.$$

By showing calculations, redraw the table after (i) insert 90; (ii) insert 83.

0	1	2	3	4	5	6	7	8	9	10	11	12
	70			44	12				51			

d) Using modulo q = 13, find out the *valid matches* and *spurious hits* that the Rabin-Karp algorithm encounters in the text T = 17426564255 when looking for the pattern P = 265. **5**

e) What do we mean when we say a problem is in *P*, *NP*, *NP-Complete*? Show the relationship among these three classes of problems. **5**