United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Final Exam Total Marks: **40** Spring 2022

Course Code: CSE 2217 Course Title: Data Structure and Algorithms II

Time: 2 hours

Any examinee found adopting unfair means will be expelled from the trimester / program as per UIU disciplinary rules.

There are **SEVEN** questions. **Answer all of them**. Show full simulation and/or solved figures wherever necessary. Figures in the right-hand margin indicate full marks.

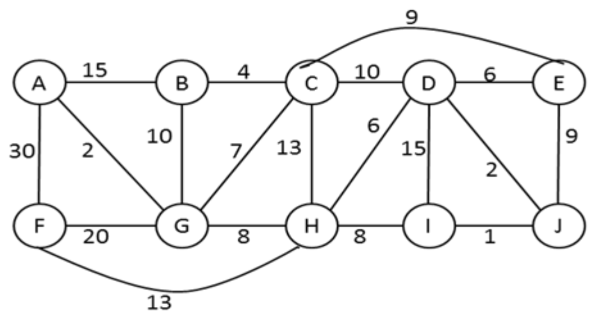


Figure 1: An undirected graph G(V,E)

1 **(a)** Find the MST for the given graph in Figure 1 using **Prim's algorithm**. Show the details of your calculation. Start from **node A**. **[3]**

(b) Given a weighted graph where multiple edges have the same weight, you are asked to apply **Kruskal's algorithm** to find MST. You have chosen the merge sort algorithm to sort the edges in ascending order. "*Depending on the sequence of the sorted edges, you may have different MSTs*". Is the statement **True** or **False**? **Justify** your answer. **[1]**

(c) Suppose you are asked to utilize a disjoint set data structure to implement **Kruskal's algorithm** to find MST from a given graph. **Explain, with an example**, how disjoint set data structure would help you to perform the following operations: **[2]**

i) joining two trees in the forest

ii) detecting a cycle

2 **(a)** Find the **Shortest path tree** for the given graph in Figure 2, where the source is vertex **y**. Show the details of your calculation. **[3]**

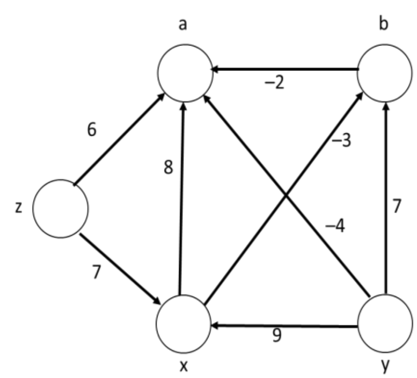


Figure 2: A directed graph G(V,E)

(b) A graph contains the vertices {A, B, C, D, E, F, G} and the shortest path from A to B is A→E→C→D→F→G→B. Is it possible to find the shortest path from **E** to **F** from given data? **Justify** your answer. **[1]**

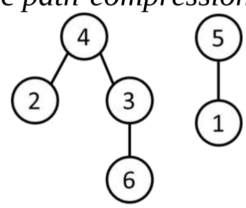
(c) What do you understand by *edge relaxation*? Let *d(v)* be the shortest path estimate from node *s* to node *v* and *δ(s,v)* be the shortest path value from node *s* to node *v*. **When does it happen** that ***d[v] = δ(s, v)*?** **[2]**

3 **(a) Show** the values for *finish time (f)* and *parent (π)* that result from running Depth-First Search (**DFS**) on the undirected graph in *Figure 1*, using vertex **A** as the source vertex. **[3]**

(b) "*Every connected directed acyclic graph (DAG) has exactly one topological ordering*" - **true** or **false**? **Design** a graph G with **exactly 4** vertices that **justifies** your reasoning. **[2]**

4 **(a) Why** do we use the heuristics: *union-by-rank* and *path-compression* in Disjoint-Set data structure? **[3]**

(b) Draw the resultant forest after calling **UNION(5, 6)** and after that **draw** the resultant forest again after calling **FIND-SET(1)** on the disjoint-sets of the following figure. You must use the *union-by-rank* and the *path-compression* heuristics. **[3]**



5 **(a) What** is *Secondary Clustering*? When might a hash table face this suffering? **[1]**

(b) Consider an open-addressing hash table as shown below. The table already contains some data items and other empty slots. Assume that collisions are handled by **Quadratic probing** using the hash function

$$h(k,i) = (h'(k) + i^2)\ mod\ 13, \quad \text{where } h'(k) = (k + 7)\ mod\ 13.$$

By showing detailed calculations, **redraw** the table after

(i) Insert 47 **[1]**
(ii) insert 64; **[1]**
(iii) delete 12 (replace with NIL); **[1]**
(iv) search 38. **[1]**

Although 38 had been present at the hash table, your search would fail here. **Explain** how one can modify the operations such that this will be prevented. **[1]**

0	1	2	3	4	5	6	7	8	9	10	11	12
70						12	38					44

Table: Open Addressing Table

6	Consider the mentioned notations Text, $t = \text{“}BATMANCATWOMAN\text{”}$ Pattern, $p = \text{“}AT\text{”}$ Modulo, $q = 13$ Hash of a string XY, $h(XY) = (X + Y)\ mod\ q$; where X and Y are the alphabetical sequence of keys [*i.e.: A =1, B=2, C=3, … Y=25, Z=26 etc*] Now answer the following questions using **Rabin-Karp** String matching algorithm: **(a) Show** the indices of *valid hit* and *spurious hit* (if any) using the aforementioned hashing function. **(b)** What is the time complexity of this algorithm? **Explain** how it improves the naïve string-matching algorithm?	 **[2+1]** **[2]**
7	**(a)** When does a problem belong to the complexity class NP? How does it differ from P class problems? **(b)** “*Being P ≠ NP drives the world economy and gives the modern Password protection system a profound advantage*.” Do you **agree** with this? Briefly **explain** your opinion.	**[2]** **[4]**