United International University (UIU)

Dept. of Computer Science and Engineering (CSE)

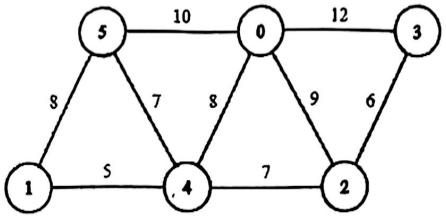
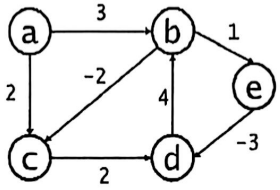
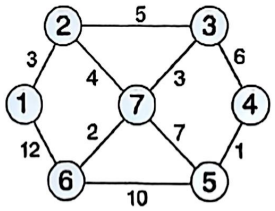
Final Exam Year: 2025 Semester: Spring

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

Marks: 40 Time: 2 Hours

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

Answer all the questions. All questions are of values indicated on the right-hand margin.

1.	**a.** Apply **Kruskal's** Algorithm to find the **Maximum Spanning Tree** (MST) of the following graph and calculate its total weight.  	[4]
	b. If an undirected graph contains **50 nodes** and **1000 edges**, which algorithm would you choose between **Prim's** and **Kruskal's** to find the minimum spanning tree (**MST**) of that graph? Justify your answer by **mentioning** the **time complexities of the algorithms**.	[2]
	c. "**Minimum spanning tree** of an undirected graph is **unique**" - Is this statement true or false? Justify your answer by constructing a graph with 4 nodes.	[1]
2.	**a.** Find the **shortest path** from 'a' to all other vertices in the following directed graph using the **appropriate** algorithm.  	[5]
	b. If a graph contains a **negative weight cycle**, why can't we find a valid shortest path from the source to some vertices? Explain with an example.	[3]
	c. Which algorithm you would prefer among **BFS**, **Dijkstra** and **Bellman-Ford** to find the shortest path in the following graph. Justify your answer with **mathematical reasoning**.  	[2]
3.	**a.** How does **path compression** heuristic **improve** the time complexity of find operation in disjoint set data structure? Explain with a proper example.	[2]

b. **Table 1** shows the **parent** array of a **Disjoint set** (Rooted tree implementation). Perform the following operations **sequentially** using **path compression** and **union-by-rank** heuristic: [2]

i. Draw the disjoint set forest. [2]
ii. What will be returned by **Find-Set(4)**, and **Find-Set(6)**? Show details. [1]
iii. Redraw the forest after **Union(6, 9)**. [1]
iv. Redraw the forest after **Union(9, 5)**.

Index	0	1	2	3	4	5	6	7	8	9	10	11
Parent	0	0	1	0	3	4	7	8	8	9	4	9

Table 1: Disjoint set (Rooted tree implementation)

4.

a. When would you choose a **Direct Address Table** over a **Hash Table** using **Chaining**? Explain with an example. [2]

b. Consider the hash function: $\mathbf{h(k, i) = (h'(k) + i^2) \bmod 13}$
where, $\mathbf{h'(k) = (2k + 1) \bmod 13}$

By proper calculation, redraw **Table 2** and illustrate the following sequence of operations, including accurate hash value calculations.

i. Insert 38 [1]
ii. Insert 51 [1]
iii. Delete 19 and replace with NIL [1]
iv. Search 51 [1]
v. If your search of 51 fails despite being present at the table, what might be the possible reason? Explain how you can modify the delete operation to prevent this from happening. [2]

Currently, the table contains the following:

0	1	2	3	4	5	6	7	8	9	10	11	12
19	32	NIL	NIL	21	NIL	NIL	42	NIL	NIL	NIL	70	NIL

Table 2: A hash table

5.

a. What is a **spurious hit** in the Rabin-Karp algorithm? Explain with an example. [2]

b. You are working on a search feature for an **e-book reader application**. One of the users wants to find whether a specific word segment exists in a book's paragraph. The paragraph contains the text: **"sunbeam".** The user wants to search for the pattern: **"eam".** You need to efficiently identify all the occurrences of the pattern within the text. [5]

The **hash code** for each character is given as follows:

character	S	U	N	B	E	A	M
code	9	5	3	2	7	1	4

The **hash function** is defined as:

$$hash(s) = (s[m-1] * d^{m-1} + s[m-2] * d^{m-2} + \ldots + s[1] * d^1 + s[0] * d^0) \bmod q$$

where, **m** = length of the string, **d** = 7, **q** = 29

Show the **indices** of both **valid hits** and **spurious hits** (if any) using the **aforementioned hash function** with detailed simulation.