Ibraheem

42896

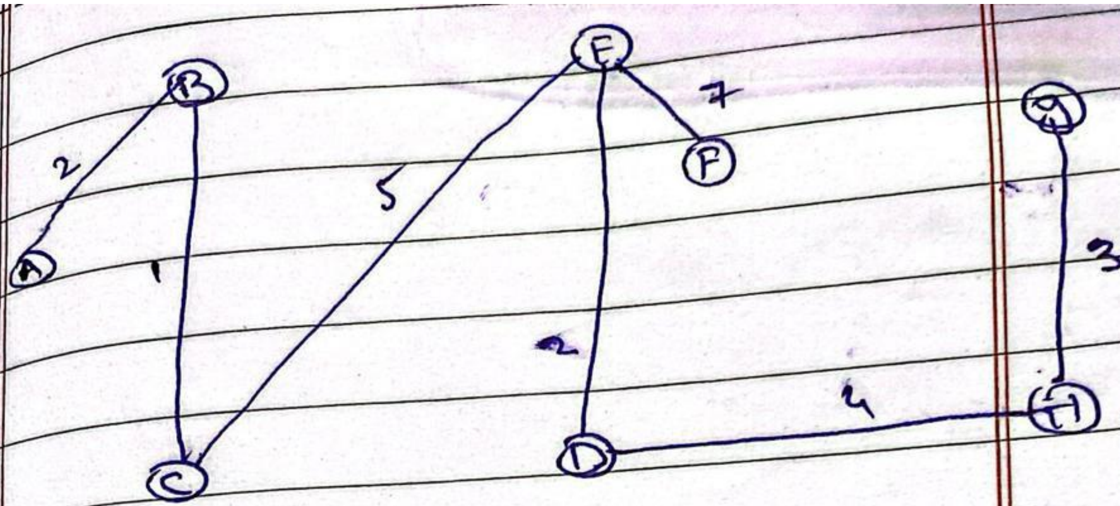
BSCS 5

Assignment 3

DAA

Problem 1

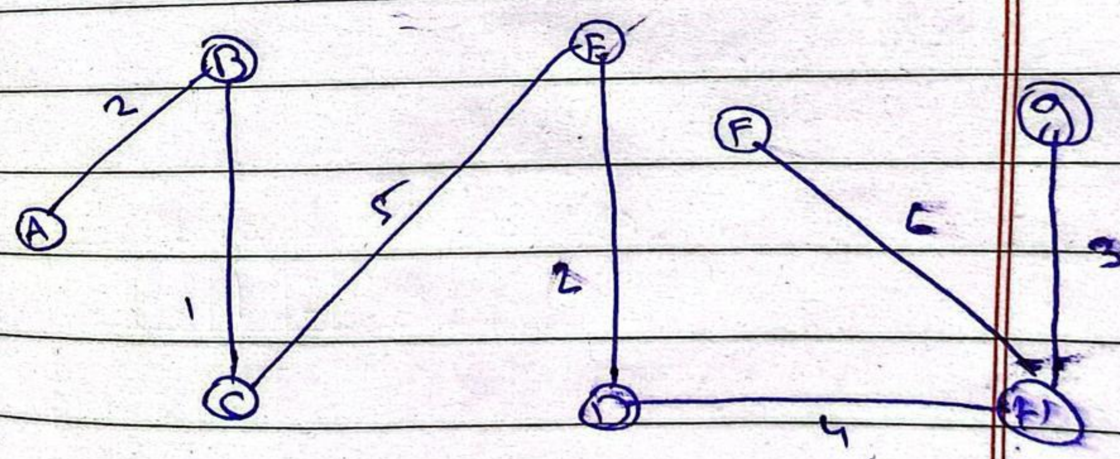
Kruskal algorithm



$1+2+5+2+7+4+3 = 24$

(b)

prims



$1+2+5+2+4+3+6 = 23$

Problem 2

(a):

Steps of Dijkstra's Algorithm, Source = A

Graph Edges:

- A → B (5), A → C (11)
- B → C (2), B → E (3)
- C → E (4), C → D (6)
- E → D (1), E → F (7)
- D → F (8)

Part b :

Can Dijkstra's algorithm handle negative weights?

- **No.** Dijkstra's algorithm assumes that once a node's shortest distance is finalized, it won't change. Negative weights **violate** this.
- Example: If a shorter path with a negative weight is found **after** visiting a node, Dijkstra will **miss** it.

Which algorithm to use instead?

- Use Bellman-Ford Algorithm. It works correctly with negative weights (as long as there are no negative cycles).

Problem 3 :

Given two strings `A` and `B`, find the **longest contiguous substring** that appears in both.

Example:

- `A = "abcdefgyu"`
- `B = "bcdtyu"`

Output: "`bcd`" (length = 3)

Dynamic Programming Approach :

If A[i-1] == B[j-1], then:

dp[i][j] = dp[i-1][j-1] + 1

Else:

dp[i][j] = 0

problem 4 :

Given two sequences `S` and `T`, the **Longest Common Subsequence** (LCS) problem is to find the length of the longest subsequence present in **both** sequences.

- **Subsequence**: A sequence derived from another by deleting some or no elements, without changing the order of the remaining elements.
- For example, LCS of "`abcdefg`" and "`bdf`" is "`bdf`" (length = 3).

Dynamic Programming Solution :

If S[i-1] == T[j-1]:

dp[i][j] = dp[i-1][j-1] + 1

Else:

dp[i][j] = max(dp[i-1][j], dp[i][j-1])