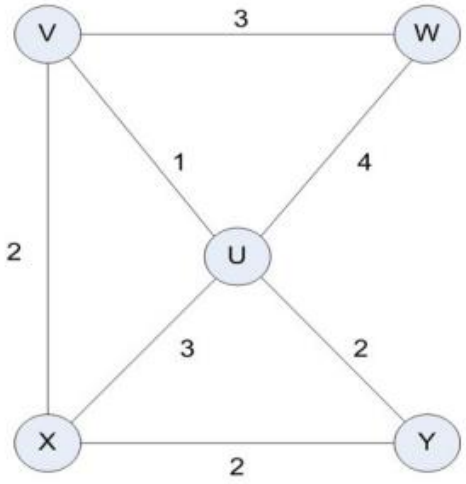
Assignment-16 – Greedy and Shortest Path

1. Carry out the steps of Dijkstra's algorithm to compute the length of the shortest path from the start vertex V.

   Write the shortest distance from V to all other Vertices.



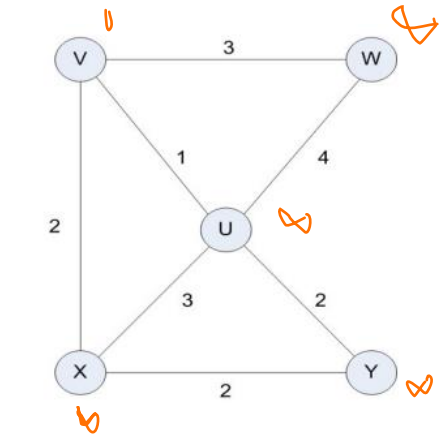
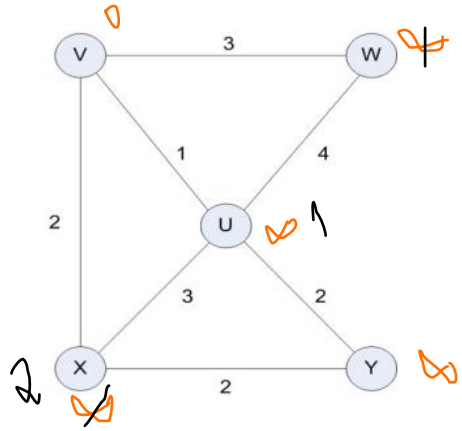
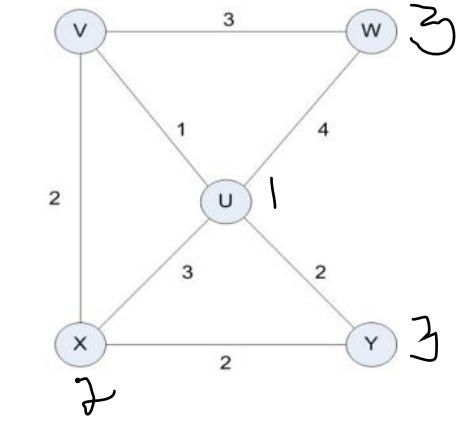
2. 0/1 Knapsack Problem

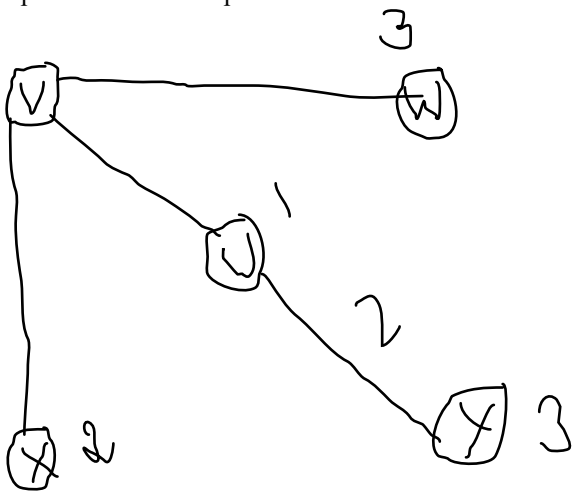
You are a traveler preparing for a journey and need to pack your backpack with limited capacity. Each item has a weight and a value, and your goal is to maximize the overall value of the items you carry while staying within the weight limit of your backpack. [Refer Slide 15]

Input

Weights: [5, 3, 2, 8, 4]
Values/Benefits: [10, 8, 5, 3, 2]
Knapsack Capacity: 10

Solution: 1 Dijkstra's algorithm to find the shortest path.

Step 1: Start from vertex V. Distance to itself is zero and distance to others vertex from v is now ∞.  	$d(z) \neg \min\{d(z), d(u)+weight(e)\}$ Initial Vertex V d(V) = 0 d(U) = ∞ d(W) = ∞ d(X) = ∞ d(Y) = ∞ V is in the Tree
Step 2: Relaxing adjacent vertex of V.  	***Adjacent of V vertex U, W and X getting relax..*** d(V) = 0 d(U) = 1 d(W) = 3 d(X) = 2 d(Y) = ∞
Step 3: Pick the vertex U since it has a minimum distance from vertex V, and keep choosing minimum and updating distance for shortest path.  	***Now V, U, W and X are in the tree.*** d(V) = 0 d(U) = 1 d(W) = 3 d(X) = 2 Relaxing vertex Y by 2. d(Y) = min{∞, d(u) + weight(y)} = min(∞, 1+ 2} = 3.

Step 4: Final Shortest path  	***Adjacent of V vertex U, W, X and Y getting relax.*** d(V) = 0 d(U) = 1 ➔A - U d(W) = 3 ➔ V - W d(X) = 2 ➔ V - X d(Y) = 3 ➔ V – U – Y

.

2. 0/1 Knapsack Problem

You are a traveler preparing for a journey and need to pack your backpack with limited capacity. Each item has a weight and a value, and your goal is to maximize the overall value of the items you carry while staying within the weight limit of your backpack. [Refer Slide 15]

Input

Weights: [5, 3, 2, 8, 4]
Values/Benefits: [10, 8, 5, 3, 2]
Knapsack Capacity: 10

Solution: 0/1 Knapsack Problem Theory, Capacity: 10

Goal: Choose items with maximum total benefit but with weight at most 10.
Object 1 (Benefit: $10, weight capacity: 5)
Object 2 (Benefit: $8, weight capacity: 3)
Object 3 (Benefit: $5, weight capacity: 2)
Total Maximum benefits = $10 + $8 + $5 = $23.

Object	1	2	3	4	5
Weights	5	3	2	8	4
Benifit	10	8	5	3	2
Selection	1	1	1	0	0