|  |  |
| --- | --- |
| **Total Marks:** | **04** |
| **Obtained Marks:** |  |

**Design and Analysis of Algorithm**

**Assignment # 04**

**Submitted To: Dr. Shahzad Latif**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Student Name: Ubaid Bin Waris**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Reg. Number: 2212416**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Q1:

Implement the dynamic programming solution for finding the Longest Common Subsequence (LCS) of the following sequences:

Sequence 1: LONGEST

Sequence 2: STONE

**Solution**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Λ** | **L** | **O** | **N** | **G** | **E** | **S** | **T** |
| **Λ** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **S** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| **T** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 2 |
| **O** | 0 | 0 | **①** | 1 | 1 | 1 | 1 | 2 |
| **N** | 0 | 0 | 1 | **②** | 2 | 2 | 2 | 2 |
| **E** | 0 | 0 | 1 | 2 | 2 | **③** | 3 | 3 |

Hene the LCS is, **ONE**

**Q2:**

Maximize the Value and mass for a 20 KG capacity using knapsack algorithm.

**Item No. Mass (Kg) Value (**$**)**

Item 1 2 1

Item 2 2 2

Item 3 3 3

Item 4 4 4

Item 5 5 4

Item 6 5 5

Item 7 6 8

Item 8 7 10

**Solution**

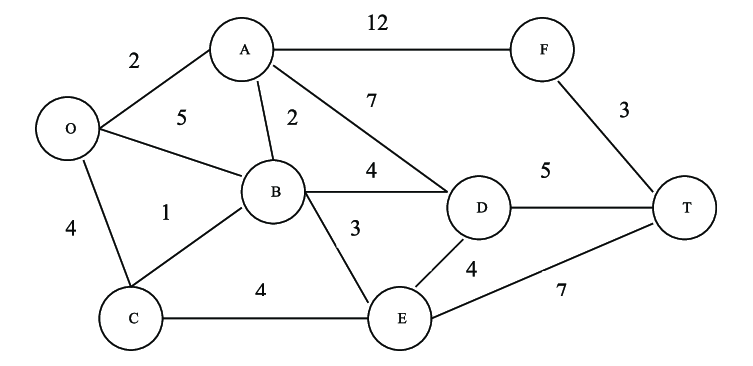
A table to find the maximum values for the kg’s and items

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Kg/items** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
| **0** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **1** | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **2** | 0 | 0 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| **3** | 0 | 0 | 2 | 3 | 3 | 5 | 5 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| **4** | 0 | 0 | 2 | 3 | 4 | 5 | 6 | 7 | 7 | 9 | 9 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| **5** | 0 | 0 | 2 | 3 | 4 | 5 | 6 | 7 | 7 | 9 | 9 | 10 | 11 | 11 | 13 | 13 | 14 | 14 | 14 | 14 | 14 |
| **6** | 0 | 0 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 12 | 13 | 14 | 15 | 16 | 16 | 18 | 18 |
| **7** | 0 | 0 | 2 | 3 | 4 | 5 | 8 | 8 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 17 | 19 | 20 | 20 | 21 |
| **8** | 0 | 0 | 2 | 3 | 4 | 5 | 8 | 10 | 10 | 12 | 13 | 14 | 15 | 18 | 17 | 20 | 21 | 22 | 23 | 24 | 25 |

Hence the maximum value is **25$** for filling 20 kg

**Q3:**

A delivery robot needs to deliver a package from the origin warehouse (O) to the target warehouse (T) in a city. The city's road network is represented as a graph, where each represents a key location, and the edges represent roads with travel costs (time in minutes). Using Dijkstra's algorithm, determine the fastest route for the delivery robot from the origin warehouse (O) to the target warehouse (T).



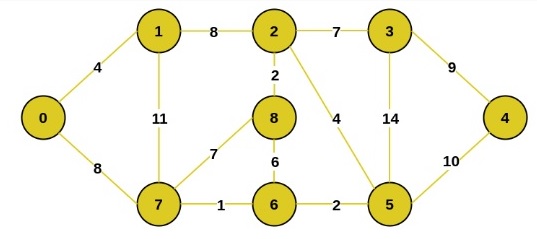
**Solution**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **A** | **B** | **C** | **D** | **E** | **F** | **T** |
| **O** | 2 (O) | 5 (O) | 4 (O) | ∞ | ∞ | ∞ | ∞ |
| **A** | 2 (O) | 4 (O, A) | 4 (O) | 9 (O, A) | ∞ | 14 (O, A) | ∞ |
| **B** | 2 (O) | 4 (O, A) | 4 (O) | 8 (O, A, B) | 7 (O, A, B) | 14 (O, A) | ∞ |
| **C** | 2 (O) | 4 (O, A) | 4 (O) | 8 (O, A, B) | 7 (O, A, B) | 14 (O, A) | ∞ |
| **E** | 2 (O) | 4 (O, A) | 4 (O) | 8 (O, A, B) | 7 (O, A, B) | 14 (O, A) | 14 (O, A, B, E) |
| **D** | 2 (O) | 4 (O, A) | 4 (O) | 8 (O, A, B) | 7 (O, A, B) | 14 (O, A) | 13 (O, A, B, D) |

Hence from O to T the Shortest Path is, **(O, A, B, D, T)** and its weight or distance is **13**.

**Q4:**

You are managing the development of a new metropolitan transportation network. To ensure efficient connectivity between major city hubs while minimizing construction costs, you need to use Prim's Algorithm to determine the minimum spanning tree (MST) of the transportation network.

******

Show step-by-step how Prim's Algorithm selects edges to form the MST, including updates to the cost and selected nodes.

**solution**

**Step 1:**

1

**Step 2:**

2

1

**Step 3:**

4

2

1

**Step 4:**

2

4

2

1

**Step 5:**

7

2

4

2

1

**Step 6:**

7

9

2

4

2

1

**Step 7:**

8

7

9

2

4

2

1

**Step 8:**

8

7

9

4

2

4

2

1

**Final Tree**

8

7

9

4

2

4

2

1

**Total Cost = 1+2+4+2+7+9+8+4 = 37**

Hence the total cost of that tree is **37.**