Name: Atakan Akar
No: 22203140
Section: 1
CS 202 HW4

Question 1)

Adjacency matrix:

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
V1	0.0	5.0	10.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0
V2	5.0	0.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0	12.0
V3	10.0	5.0	0.0	7.0	8.0	9.0	10.0	11.0	12.0	13.0
V4	5.0	6.0	7.0	0.0	9.0	10.0	11.0	12.0	13.0	14.0
V5	6.0	7.0	8.0	9.0	0.0	11.0	12.0	13.0	14.0	15.0
V6	7.0	8.0	9.0	10.0	11.0	0.0	13.0	14.0	15.0	16.0
V7	8.0	9.0	10.0	11.0	12.0	13.0	0.0	15.0	16.0	17.0
V8	9.0	10.0	11.0	12.0	13.0	14.0	15.0	0.0	17.0	18.0
V9	10.0	11.0	12.0	13.0	14.0	15.0	16.0	17.0	0.0	19.0
V10	11.0	12.0	13.0	14.0	15.0	16.0	17.0	18.0	19.0	0.0

To determine the minimum spanning tree (MST), we must identify and add the edges with the smallest weights to our tree while ensuring that no cycles is created. The problem specifies that there are 10 vertices, numbered 1 to 10, and the graph is complete. Since the goal is to find the MST, we focus on minimizing the total weight. Using Kruskal's algorithm is good in this scenario as it starts by selecting the arc with the smallest weight. Kruskal's algorithm is a greedy algorithm that builds the Minimum Spanning Tree (MST) by starting with the edge of the smallest weight and incrementally adding the next smallest edge, provided it does not form a cycle, until the minimum spanning tree is created.

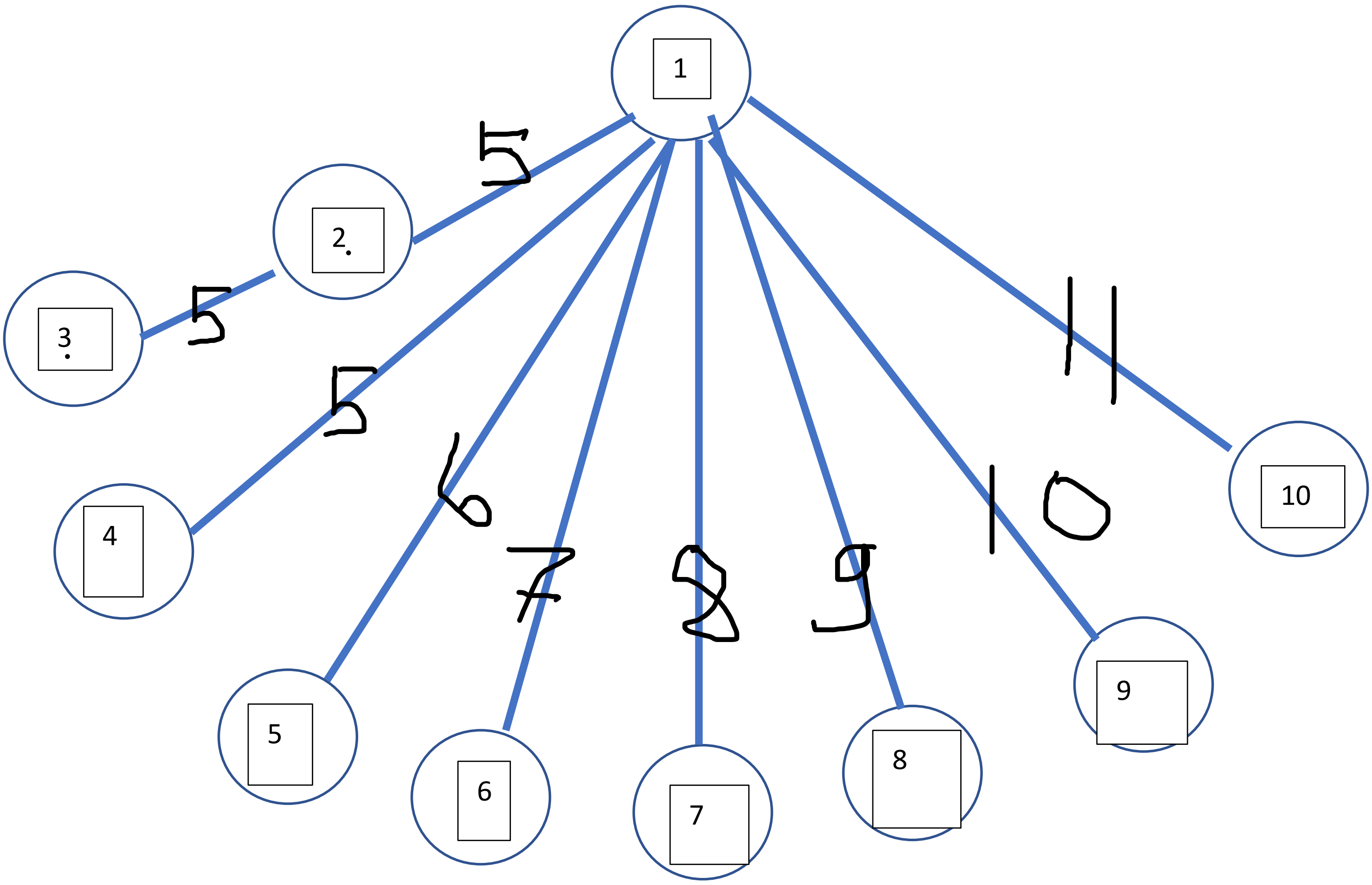
Begin with the arc (1, 2). Process:

(1, 2) > 5 (1, 4) > 5 (2, 3) > 5 (1, 5) > 6 (1, 6) > 7 (1, 7) > 8

(1, 8) > 9 (1, 9) > 10 (1, 10) > 11

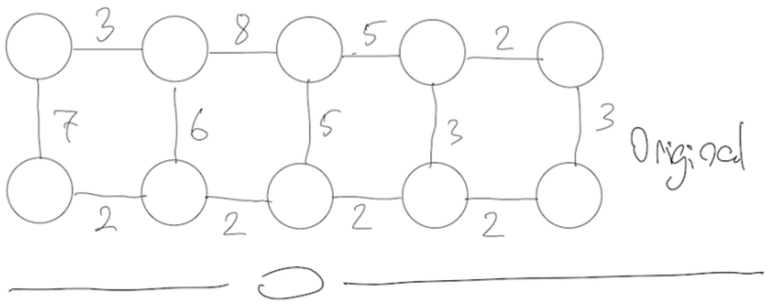
5+5+5+6+7+8+9+10+11= 66

There are E – 1 arcs so 10 -1 9. And also it does not contaion cycles.

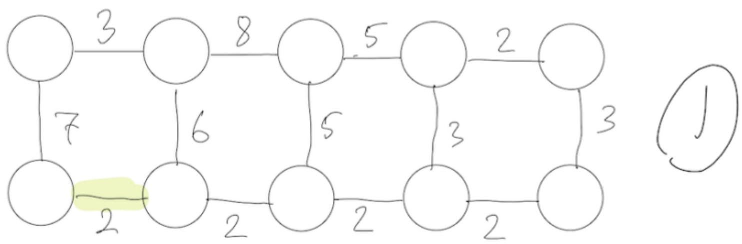

1
2.
3.
4
5
6
7
8
9
10
5
5
5
6
7
8
9
10
11

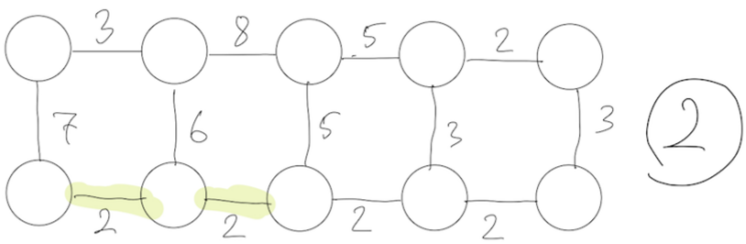
Question 2)

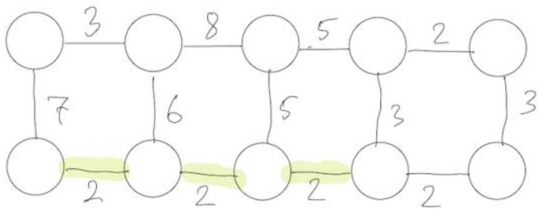
The problem requires minimizing the network cost. We must find the minimum cost spanning tree. Since the graph is relatively dense, with a total of 10 nodes and 13 edges, Kruskal’s algorithm will be more suitable than Prim’s algorithm. Kruskal’s algorithm has a complexity of O(ElogV), because it requires sorting all edges and uses a Union-Find data structure to handle cycle detection and also get benefit from a heap. In contrast, Prim’s algorithm, when implemented with an adjacency matrix, has a complexity of O(V^2), where V is the number of vertices. This makes Kruskal’s algorithm more efficient for this problem due to its edge-centric approach. Kruskal's algorithm requires that all of the graph's edges to be sorted. Once the edges have been sorted and processed, the minimum weight edge among the remaining edges is added to the tree. However, it is not inserted if that edge creates a cycle in the tree because it distorts the MST feature. By ensuring no cycles are created when adding edges, Kruskal's algorithm preserves the properties of a tree while efficiently building the minimum spanning tree.



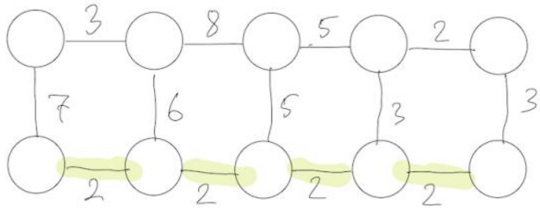
Choose min arc since there are so many decide one



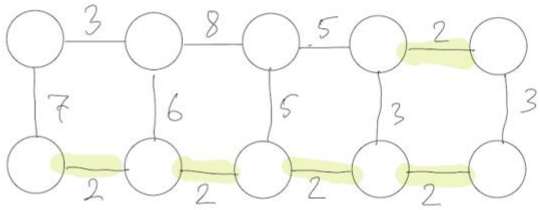




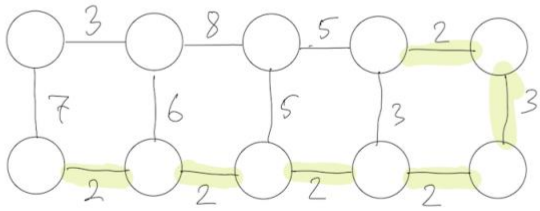
3



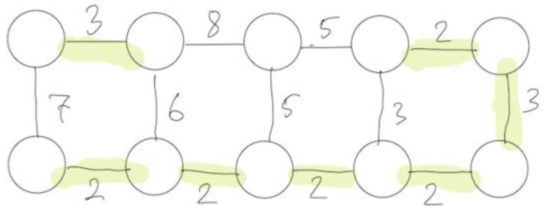
4



5

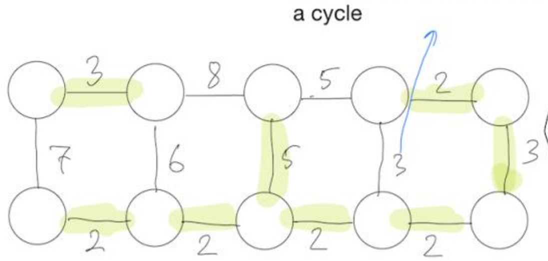


6

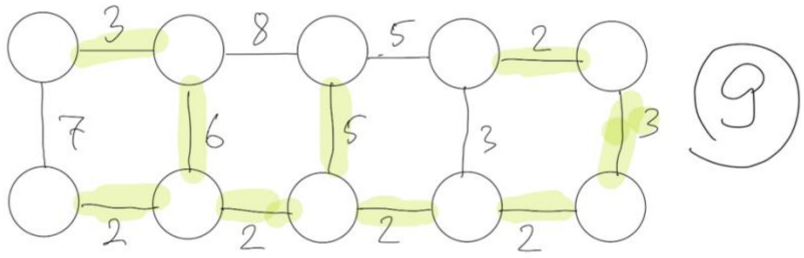


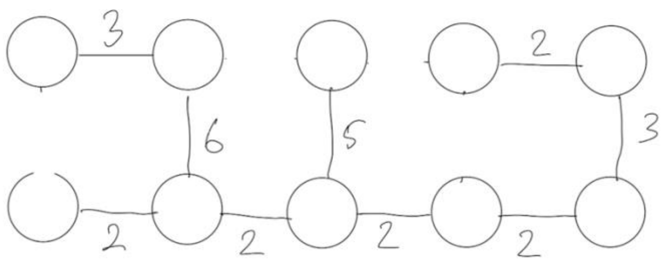
7

Cant choosen since forms
a cycle



8





The minimum spanning tree might also be created with different shape since there are multiple arcs with same weight however arcs with weight 2 must be in the tree according to Kruskal's algorithm and therefore one of the arcs with weight 3 must be excluded. Moreover, one of the arc with 5 must also be excluded since they would form a cycle.

2+2+2+2+2+3+3+5+6 = 27 MST Total Weight.