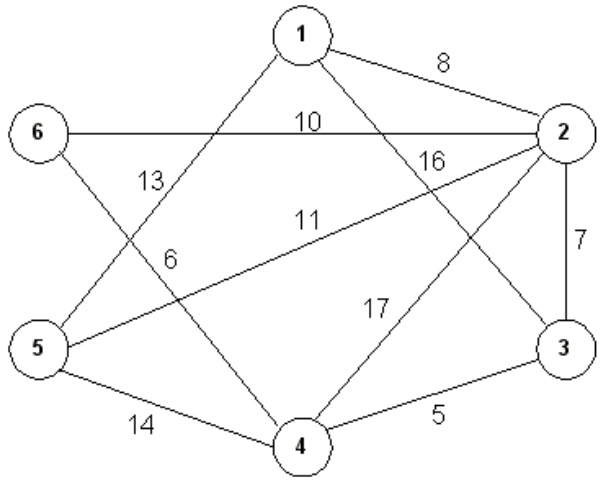
Example Diagrams for Dijkstra's Algorithm

1. An undirected, weighted graph with 6 vertices and 10 edges: (Dijkstra1 in the driver)

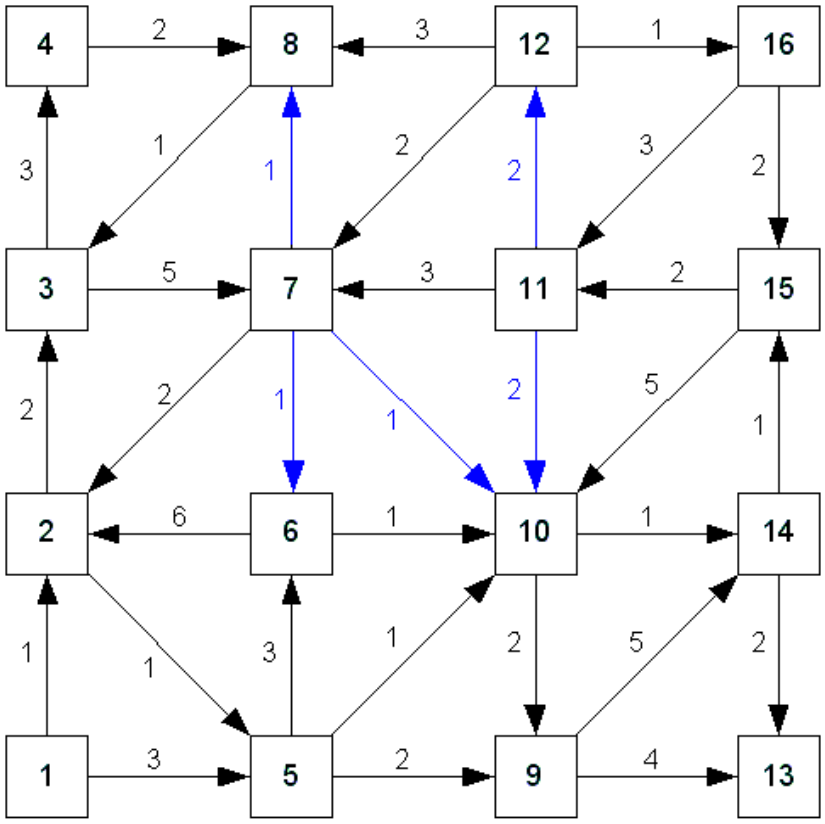



```
ALGraph g(6);

g.AddUEdge(1, 2, 8);
g.AddUEdge(1, 3, 16);
g.AddUEdge(1, 5, 13);
g.AddUEdge(2, 3, 7);
g.AddUEdge(2, 4, 17);
g.AddUEdge(2, 5, 11);
g.AddUEdge(2, 6, 10);
g.AddUEdge(3, 4, 5);
g.AddUEdge(4, 5, 14);
g.AddUEdge(4, 6, 6);
```

Output:

```
Adjacency list:
-------------------------------
ID: [ 1] --   8 --> [ 2] --  13 --> [ 5] --  16 --> [ 3]
ID: [ 2] --   7 --> [ 3] --   8 --> [ 1] --  10 --> [ 6] --  11 --> [ 5] --  17 -->
[ 4]
ID: [ 3] --   5 --> [ 4] --   7 --> [ 2] --  16 --> [ 1]
ID: [ 4] --   5 --> [ 3] --   6 --> [ 6] --  14 --> [ 5] --  17 --> [ 2]
ID: [ 5] --  11 --> [ 2] --  13 --> [ 1] --  14 --> [ 4]
ID: [ 6] --   6 --> [ 4] --  10 --> [ 2]

Cost to reach all nodes from node 1:
-----------------------------------
Node:  1:  Cost:   0  Path:  1
Node:  2:  Cost:   8  Path:  1  2
Node:  3:  Cost:  15  Path:  1  2  3
Node:  4:  Cost:  20  Path:  1  2  3  4
Node:  5:  Cost:  13  Path:  1  5
Node:  6:  Cost:  18  Path:  1  2  6
```

2. A directed, weighted graph with 16 vertices and 33 edges: (Dijkstra4 in the driver).
Pay attention to the order of the edges in the adjacency list when there are multiple edges with the same weight.




```
ALGraph g(16);

g.AddDEdge(1, 2, 1);
g.AddDEdge(1, 5, 3);
g.AddDEdge(2, 3, 2);
g.AddDEdge(2, 5, 1);
g.AddDEdge(3, 4, 3);
g.AddDEdge(3, 7, 5);
g.AddDEdge(4, 8, 2);
g.AddDEdge(5, 6, 3);
g.AddDEdge(5, 9, 2);
g.AddDEdge(5, 10, 1);
g.AddDEdge(6, 2, 6);
g.AddDEdge(6, 10, 1);
g.AddDEdge(7, 2, 2);
g.AddDEdge(7, 8, 1);
g.AddDEdge(7, 6, 1);
g.AddDEdge(7, 10, 1);
g.AddDEdge(8, 3, 1);
g.AddDEdge(9, 13, 4);
g.AddDEdge(9, 14, 5);
g.AddDEdge(10, 9, 2);
g.AddDEdge(10, 14, 1);
g.AddDEdge(11, 7, 3);
g.AddDEdge(11, 10, 2);
g.AddDEdge(11, 12, 2);
g.AddDEdge(12, 7, 2);
g.AddDEdge(12, 8, 3);
g.AddDEdge(12, 16, 1);
g.AddDEdge(14, 13, 2);
g.AddDEdge(14, 15, 1);
g.AddDEdge(15, 10, 5);
g.AddDEdge(15, 11, 2);
g.AddDEdge(16, 11, 3);
g.AddDEdge(16, 15, 2);
```

Output:

```
Adjacency list:
--------------------------------
ID: [ 1] --   1 --> [ 2] --   3 --> [ 5]
ID: [ 2] --   1 --> [ 5] --   2 --> [ 3]
ID: [ 3] --   3 --> [ 4] --   5 --> [ 7]
ID: [ 4] --   2 --> [ 8]
ID: [ 5] --   1 --> [10] --   2 --> [ 9] --   3 --> [ 6]
ID: [ 6] --   1 --> [10] --   6 --> [ 2]
ID: [ 7] --   1 --> [ 6] --   1 --> [ 8] --   1 --> [10] --   2 --> [ 2]
ID: [ 8] --   1 --> [ 3]
ID: [ 9] --   4 --> [13] --   5 --> [14]
ID: [10] --   1 --> [14] --   2 --> [ 9]
ID: [11] --   2 --> [10] --   2 --> [12] --   3 --> [ 7]
ID: [12] --   1 --> [16] --   2 --> [ 7] --   3 --> [ 8]
ID: [13]
ID: [14] --   1 --> [15] --   2 --> [13]
ID: [15] --   2 --> [11] --   5 --> [10]
ID: [16] --   2 --> [15] --   3 --> [11]

Cost to reach all nodes from node 1:
-----------------------------------
Node:  1:  Cost:   0  Path:  1
Node:  2:  Cost:   1  Path:  1  2
```

```
Node:  3:  Cost:   3  Path:  1  2  3
Node:  4:  Cost:   6  Path:  1  2  3  4
Node:  5:  Cost:   2  Path:  1  2  5
Node:  6:  Cost:   5  Path:  1  2  5  6
Node:  7:  Cost:   8  Path:  1  2  3  7
Node:  8:  Cost:   8  Path:  1  2  3  4  8
Node:  9:  Cost:   4  Path:  1  2  5  9
Node: 10:  Cost:   3  Path:  1  2  5 10
Node: 11:  Cost:   7  Path:  1  2  5 10 14 15 11
Node: 12:  Cost:   9  Path:  1  2  5 10 14 15 11 12
Node: 13:  Cost:   6  Path:  1  2  5 10 14 13
Node: 14:  Cost:   4  Path:  1  2  5 10 14
Node: 15:  Cost:   5  Path:  1  2  5 10 14 15
Node: 16:  Cost:  10  Path:  1  2  5 10 14 15 11 12 16
```