Minimum Spaning Tree Report

Name : Remon Hanna Wadie Youssef.

Number : 23.

Subject : Data Structures 2 assignment (MST).

1) Input:

There are two ways to input the graph:

First : press(1) to read the graph from the (input.txt) file.

Second : press(2) to enter the information of the graph in the console.

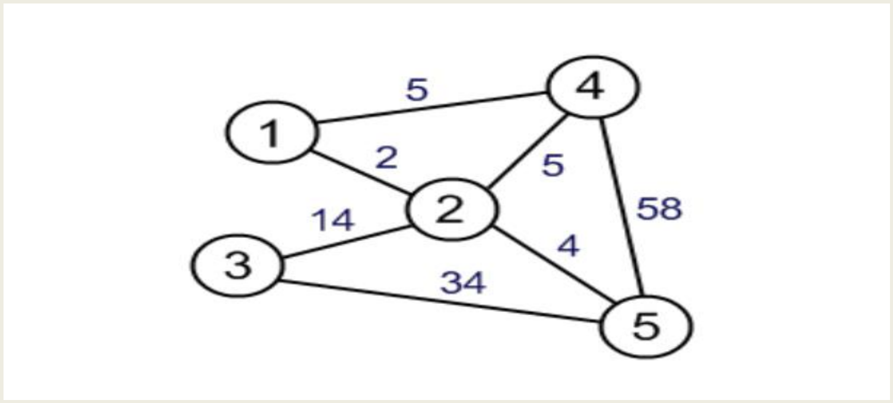
The input format like that:

```
5
7
1,2,2
1,4,5
2,3,14
2,4,5
2,5,4
3,5,34
4,5,58
```

Line 1: number of vertices.

Line 2: number of edges (n).

Line 3 … 3 + n: i,j,w represent edge between vertices i,j weighted by w.



2) Algorithms:

A) Prim Algorithm:

Type of the graph	Complexity
Adjacency Matrix	O(V^2)
Adjacency List	O(V logE)

B) Kruskal Algorithm:

Type of the graph	Complexity
Adjacency Matrix	O($E \log E$)
Adjacency List	O(E log E)

---

2) Data Sturcture used:

Graph representation:

1) Adjacency Matrix:

   2D Array.

2) Adjacency List:

   Array of arrayList.

3) Disjoint set data structure:

   To check there is a cycle in the graph when adding new edge.

4) Test Cases:

Test 1:

Input File:

```
7
12
1,2,2
```

```
1,4,5
2,3,14
2,4,5
2,5,4
3,5,34
4,5,58
6,2,22
6,4,20
7,5,18
7,1,2
7,2,9
```

Output:

```
Adjacency Matrix Representation:
Minimum Cost = 47
Time of execution = 403955.0 NS.
Time of execution = 0.403955 MS.
Edges from MST (prim Algorithm):
Edge(0) 1<--->2  weight = 2
Edge(1) 1<--->7  weight = 2
Edge(2) 2<--->5  weight = 4
Edge(3) 1<--->4  weight = 5
Edge(4) 2<--->3  weight = 14
Edge(5) 4<--->6  weight = 20
---------------------------------------------------
Edges from MST (Kruskal Algorithm):
Minimum Cost (Kruskal) = 47
Time of execution = 2479969.0 NS.
Time of execution = 2.479969 MS.
Edge(0) 1<--->2  weight = 2
Edge(1) 7<--->1  weight = 2
Edge(2) 2<--->5  weight = 4
Edge(3) 2<--->4  weight = 5
Edge(4) 2<--->3  weight = 14
Edge(5) 6<--->4  weight = 20
```

```
=================================================
Adjacency List Representation:
Minimum Cost = 47
Time of execution = 278334.0 NS.
Time of execution = 0.27833399999999997 MS.
Edges from MST (prim Algorithm):
Edge(0) 1<--->2  weight = 2
Edge(1) 1<--->7  weight = 2
Edge(2) 2<--->5  weight = 4
Edge(3) 2<--->4  weight = 5
Edge(4) 2<--->3  weight = 14
Edge(5) 4<--->6  weight = 20
-------------------------------------------------
Minimum Cost (Kruskal) = 47
Time of execution = 247135.0 NS.
Time of execution = 0.247135 MS.
Edges from MST (Kruskal Algorithm):
Edge(0) 1<--->2  weight = 2
Edge(1) 7<--->1  weight = 2
Edge(2) 2<--->5  weight = 4
Edge(3) 4<--->2  weight = 5
Edge(4) 2<--->3  weight = 14
Edge(5) 6<--->4  weight = 20
```

	Adjacency Matrix Time	Ajacency List Time
Prim	0.403955 MS.	0.27833399999999997 MS.
Kruskal	2.479969 MS.	0.247135 MS.

Note:

1) Adjacency Matrix take time greater than adjacency list in both algorithms prim and kruskal.

Test 2:

Input File:

```
13
18
2,1,3
2,3,8
3,1,9
3,4,15
4,5,12
6,5,18
1,6,2
11,6,40
9,6,17
8,7,20
5,7,2
8,9,50
9,11,35
10,12,40
11,13,18
12,1,4
13,1,3
8,1,5
```

```
Output:

Adjacency Matrix Representation:
Minimum Cost = 129
Time of execution = 471685.0 NS.
Edges from MST (prim Algorithm):
Edge(0) 2<--->1  weight = 3
Edge(1) 1<--->6  weight = 2
Edge(2) 1<--->13  weight = 3
Edge(3) 1<--->12  weight = 4
Edge(4) 1<--->8  weight = 5
Edge(5) 2<--->3  weight = 8
Edge(6) 3<--->4  weight = 15
Edge(7) 4<--->5  weight = 12
Edge(8) 5<--->7  weight = 2
Edge(9) 6<--->9  weight = 17
Edge(10) 13<--->11  weight = 18
Edge(11) 12<--->10  weight = 40
--------------------------------------------------
Edges from MST (Kruskal Algorithm):
Minimum Cost (Kruskal) = 129
Time of execution = 2433551.0 NS.
Edge(0) 1<--->6  weight = 2
Edge(1) 5<--->7  weight = 2
Edge(2) 13<--->1  weight = 3
Edge(3) 2<--->1  weight = 3
Edge(4) 12<--->1  weight = 4
Edge(5) 8<--->1  weight = 5
Edge(6) 2<--->3  weight = 8
Edge(7) 4<--->5  weight = 12
Edge(8) 3<--->4  weight = 15
Edge(9) 9<--->6  weight = 17
Edge(10) 11<--->13  weight = 18
Edge(11) 10<--->12  weight = 40
==================================================
Adjacency List Representation:
Minimum Cost = 129
Time of execution = 313636.0 NS.
Edges from MST (prim Algorithm):
Edge(0) 2<--->1  weight = 3
Edge(1) 1<--->6  weight = 2
Edge(2) 1<--->13  weight = 3
Edge(3) 1<--->12  weight = 4
Edge(4) 1<--->8  weight = 5
Edge(5) 2<--->3  weight = 8
Edge(6) 3<--->4  weight = 15
Edge(7) 4<--->5  weight = 12
Edge(8) 5<--->7  weight = 2
Edge(9) 6<--->9  weight = 17
Edge(10) 13<--->11  weight = 18
Edge(11) 12<--->10  weight = 40
--------------------------------------------------
```

```
Minimum Cost (Kruskal) = 129
Time of execution = 338677.0 NS.
Edges from MST (Kruskal Algorithm):
Edge(0) 1<--->6  weight = 2
Edge(1) 5<--->7  weight = 2
Edge(2) 13<--->1  weight = 3
Edge(3) 1<--->2  weight = 3
Edge(4) 1<--->12  weight = 4
Edge(5) 8<--->1  weight = 5
Edge(6) 3<--->2  weight = 8
Edge(7) 4<--->5  weight = 12
Edge(8) 4<--->3  weight = 15
Edge(9) 6<--->9  weight = 17
Edge(10) 13<--->11  weight = 18
Edge(11) 10<--->12  weight = 40
```

	Adjacency Matrix Time	Ajacency List Time
Prim	0.24590399999999998 MS.	0.146967 MS.
Kruskal	1.1818959999999998 MS.	0.150662 MS.

Note:

1) Adjacency Matrix take time greater than adjacency list in both algorithms prim and kruskal.
2) Prim algorithm in the adjacency matrix representation is faster than kruskal.

Test 3:

Input File:

```
10
17
1,2,3
1,6,2
6,7,7
6,5,1
5,4,11
9,4,4
9,3,18
3,4,8
2,3,17
9,10,9
9,8,12
9,5,10
2,4,16
7,8,15
7,5,6
5,8,5
8,10,13
```

	Adjacency Matrix Time	Ajacency List Time
Prim	0.243441 MS.	0.151483 MS.
Kruskal	1.3855149999999998 MS.	0.167493 MS.

Note:

1) Adjacency Matrix take time greater than adjacency list in both algorithms prim and kruskal.
2) Prim algorithm in the adjacency matrix representation is faster than kruskal.

Output:

```
Adjacency Matrix Representation:
Minimum Cost = 48
Time of execution = 403950.0 NS.
Edges from MST (prim Algorithm):
Edge(0) 1<--->6  weight = 2
Edge(1) 6<--->5  weight = 1
Edge(2) 1<--->2  weight = 3
Edge(3) 5<--->8  weight = 5
Edge(4) 5<--->7  weight = 6
Edge(5) 5<--->9  weight = 10
Edge(6) 9<--->4  weight = 4
Edge(7) 4<--->3  weight = 8
Edge(8) 9<--->10  weight = 9
---------------------------------------------------
Edges from MST (Kruskal Algorithm):
Minimum Cost (Kruskal) = 48
Time of execution = 2380183.0 NS.
Edge(0) 6<--->5  weight = 1
Edge(1) 1<--->6  weight = 2
Edge(2) 1<--->2  weight = 3
Edge(3) 9<--->4  weight = 4
Edge(4) 5<--->8  weight = 5
Edge(5) 7<--->5  weight = 6
Edge(6) 3<--->4  weight = 8
Edge(7) 9<--->10  weight = 9
Edge(8) 9<--->5  weight = 10
===================================================
Adjacency List Representation:
Minimum Cost = 48
Time of execution = 310352.0 NS.
Edges from MST (prim Algorithm):
Edge(0) 1<--->6  weight = 2
Edge(1) 6<--->5  weight = 1
Edge(2) 1<--->2  weight = 3
Edge(3) 5<--->8  weight = 5
Edge(4) 5<--->7  weight = 6
Edge(5) 5<--->9  weight = 10
Edge(6) 9<--->4  weight = 4
Edge(7) 4<--->3  weight = 8
Edge(8) 9<--->10  weight = 9
--------------------------------------------------
Minimum Cost (Kruskal) = 48
Time of execution = 337856.0 NS.
Edges from MST (Kruskal Algorithm):
Edge(0) 6<--->5  weight = 1
Edge(1) 6<--->1  weight = 2
Edge(2) 2<--->1  weight = 3
Edge(3) 9<--->4  weight = 4
Edge(4) 8<--->5  weight = 5
Edge(5) 7<--->5  weight = 6
Edge(6) 4<--->3  weight = 8
Edge(7) 9<--->10  weight = 9
Edge(8) 9<--->5  weight = 10
```

Test 4:

Input File:

```
5
7
1,2,2
1,4,5
2,3,14
2,4,5
2,5,4
3,5,34
4,5,58
```

	Adjacency Matrix Time	Ajacency List Time
Prim	0.19048299999999999 MS.	0.09934699999999999 MS.
Kruskal	.018097 MS.	0.09524099999999999 MS.

Note:

1) djacency Matrix take time greater than adjacency list in both algorithms prim and kruskal.
2) Prim algorithm in the adjacency matrix representation is faster than kruskal.

Output:

```
Adjacency Matrix Representation:
Minimum Cost = 25
Time of execution = 348530.0 NS.
Edges from MST (prim Algorithm):
Edge(0) 1<--->2  weight = 2
Edge(1) 2<--->5  weight = 4
Edge(2) 1<--->4  weight = 5
Edge(3) 2<--->3  weight = 14
--------------------------------------------------
Edges from MST (Kruskal Algorithm):
Minimum Cost (Kruskal) = 25
Time of execution = 2247176.0 NS.
Edge(0) 1<--->2  weight = 2
Edge(1) 2<--->5  weight = 4
Edge(2) 2<--->4  weight = 5
Edge(3) 2<--->3  weight = 14
==================================================
Adjacency List Representation:
Minimum Cost = 25
Time of execution = 193765.0 NS.
Edges from MST (prim Algorithm):
Edge(0) 1<--->2  weight = 2
Edge(1) 2<--->5  weight = 4
Edge(2) 1<--->4  weight = 5
Edge(3) 2<--->3  weight = 14
--------------------------------------------------
Minimum Cost (Kruskal) = 25
Time of execution = 192533.0 NS.
Edges from MST (Kruskal Algorithm):
```

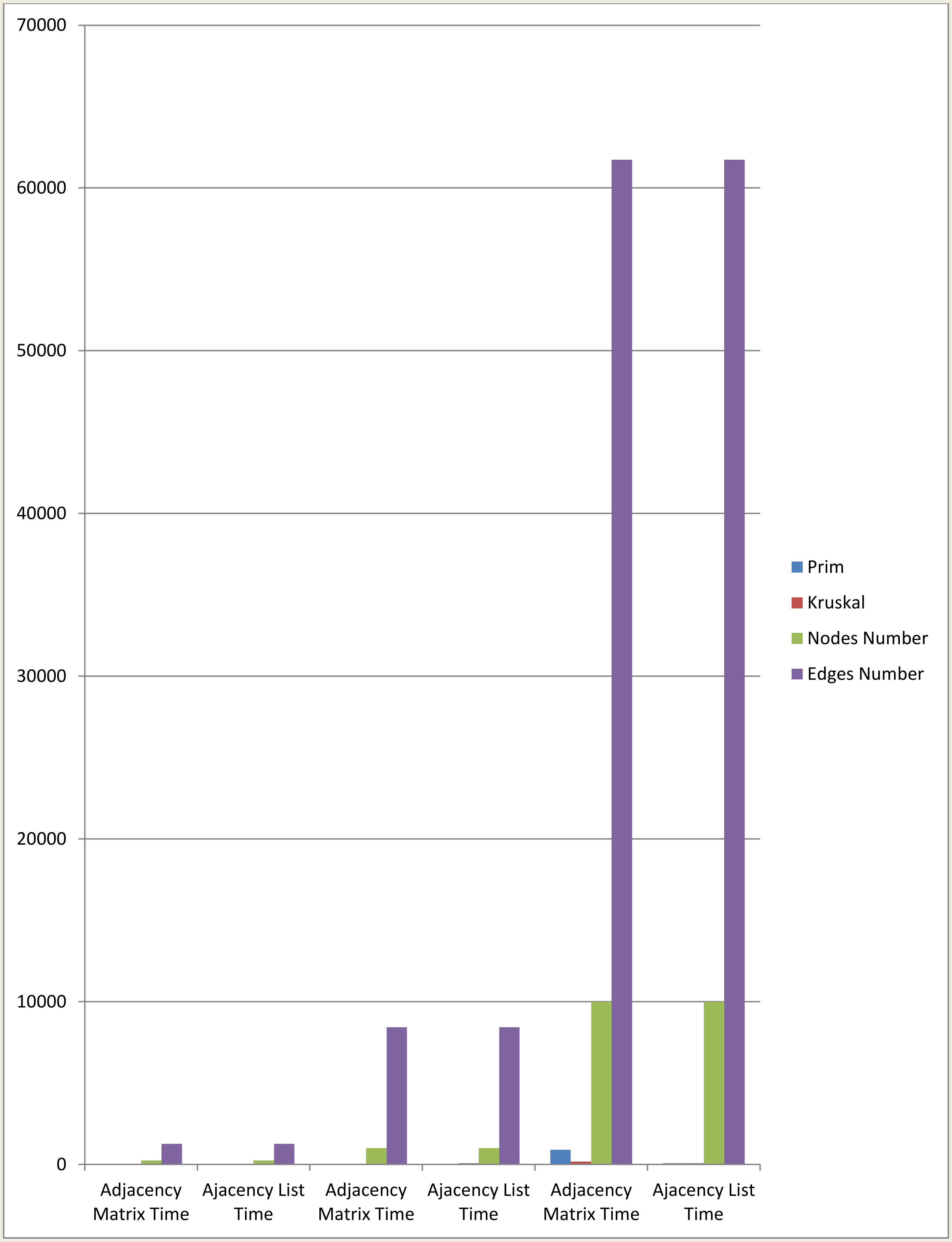
```
Edge(0) 1<--->2  weight = 2
Edge(1) 2<--->5  weight = 4
Edge(2) 4<--->2  weight = 5
Edge(3) 2<--->3  weight = 14
```

Test [5]:

	Adjacency Matrix Time	Ajacency List Time	Nodes Number	Edges Number
Prim	19.600011 MS.	6.2190129999999995 MS.	250	1273
Kruskal	9.547533 MS.	6.423864 MS.	250	1273
Prim	25.738152 MS.	15.599053999999999 MS.	1000	8433
Kruskal	23.120245999999998 MS.	74.94261999999999 MS.	1000	8433
Prim	898.848359 MS.	69.595967 MS.	10000	61731
Kruskal	176.433762 MS.	82.107067 MS.	10000	61731

From All this tests we can conclude that:

1) djacency Matrix take time greater than adjacency list in both algorithms prim and kruskal.
2) Prim algorithm in the adjacency matrix representation is slower than prim in the adjacency list representation.
3) Kruskal algorithm in the adjacency matrix representation is faster than kruskal in the adjacency list representation.



70000
60000
50000
40000
30000
20000
10000
0
Adjacency Matrix Time
Ajacency List Time
Adjacency Matrix Time
Ajacency List Time
Adjacency Matrix Time
Ajacency List Time
Prim
Kruskal
Nodes Number
Edges Number