Data Structure

Lab Session #13: Graphs 2

U Kang
Seoul National University

Goals

- Implement the Prim's algorithm.
  - Fill your codes in "Prim.java".
  - Use an adjacency list structure to store the edges.
- Print the sample output corresponding to the sample input.

Notice

- After implementing the code, check if your program works well
  - Check sample input and output files in the ‘testdata’ folder
  - Test your program by using them
- Please raise your hand and ask to T.A. if you have any question regarding the problems
- You need to stay for at least and hour

Build a Project

- Download the project for the lab from eTL
- Extract the project, and open it in IntelliJ
  - See the slide of 1st lab session to check how to open the project in IntelliJ

Function to Implement

- *findMST(Graph G)*
  - Find the MST using Prim's algorithm.
- *getNextVertex(Graph G, double key[], Boolean isVisited[])*
  - Find the next vertex to be included.
- *print(Graph G, int[] parent)*
  - Print the MST obtained by the Prim's algorithm.
  - Print the list of edges included in the MST and the total weight.

I/O Specification

- findMST

Input form	Output form
`Graph G`	
Description	
- **find the minimum spanning tree using Prim's algorithm.** - **Print the list of the edges included in the MST and the total weight of it.**	
Example Input	**Example Output**
`computeMST(G)`	

I/O Specification

- getNextVertex

Input form	Output form
`Graph G, double key[], Boolean isVisited[]`	**int minIndex**
Description	
- **Find the next vertex to be included.** - **Find the vertex which has minimum weight edge and not visited.**	
Example Input	**Example Output**
`getNextVertex(G,key,V)`	**3**

I/O Specification

- print

Input form	Output form
`Graph G, int[] parent`	
Description	
- **Print the minimum spanning tree.** - **Print each edges using following format:** - **"Edge: (start) to (end), weight: (weight)"** - **Print the total weight at the end: "Total weight: (total weight)"**	
Example Input	**Example Output**
`Print(G, parent)`	**Edge: 3 to 1 weight: 7.0 …**

Sample Input and Output

- Input

```
n 7
edge 0 1 10
edge 0 3 5
edge 1 2 2
edge 1 3 7
edge 1 4 12
edge 2 4 11
edge 2 5 14
edge 3 4 6
edge 3 6 9
edge 4 5 15
edge 5 6 3
edge 1 0 10
edge 3 0 5
edge 2 1 2
edge 3 1 7
edge 4 1 12
edge 4 2 11
edge 5 2 14
edge 4 3 6
edge 6 3 9
edge 5 4 15
edge 6 5 3
mst
```

- Output

```
Edge: 3 to 1, weight: 7.0
Edge: 1 to 2, weight: 2.0
Edge: 0 to 3, weight: 5.0
Edge: 3 to 4, weight: 6.0
Edge: 6 to 5, weight: 3.0
Edge: 3 to 6, weight: 9.0
Total weight: 32
```

Questions?