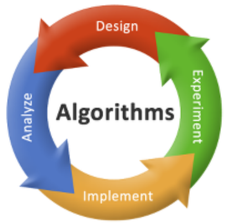
Minimum Spanning Trees

Course: Algorithms



Faculty: Dr. Rajendra Prasath

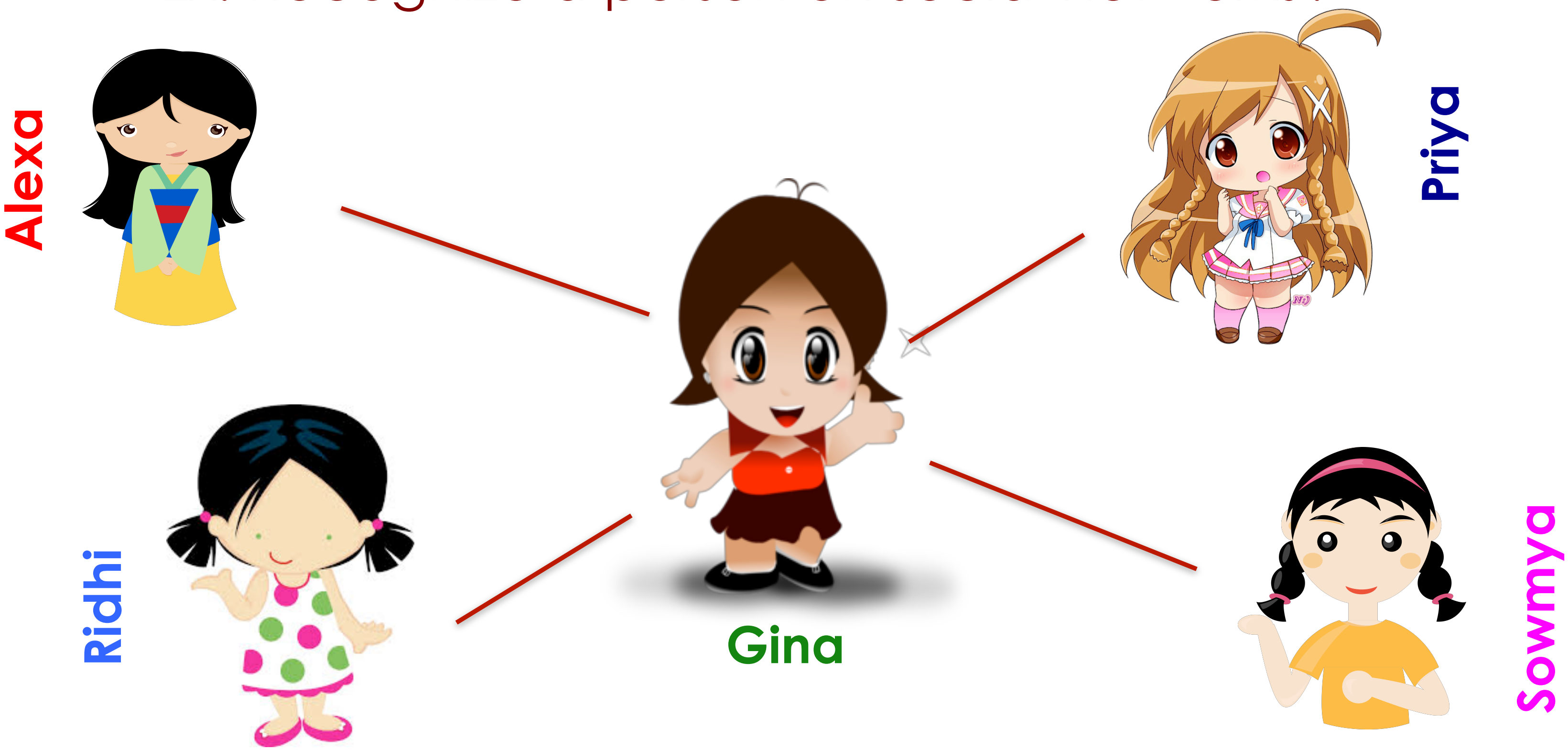
Autumn 2018

Minimum Spanning Trees – Prim and Kruskal

This lecture covers the two interesting Minimum Spanning Tree (MST) algorithms: Prim and Kruskal. We provide illustrations and the complexity analysis of these two MST algorithms

Recap: Graph Algorithms

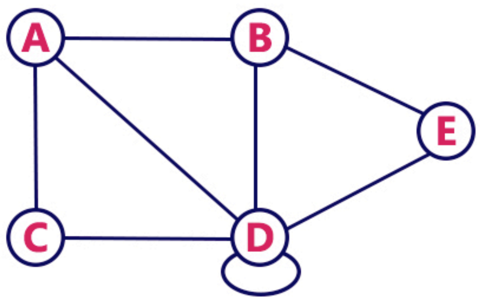
- Different Networks
  - Ex: Recognize a person on social networks?



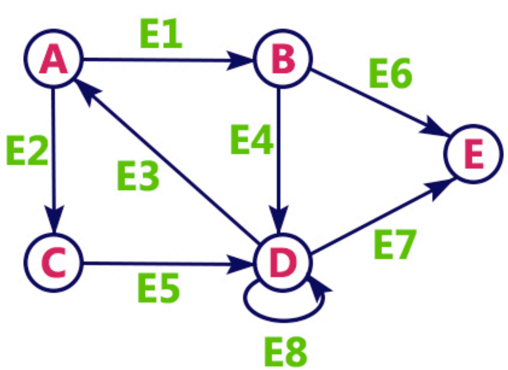
- Friend of a Friend is also a **Friend (Mutual Friend)**

Recap: Graph Representations

- Examples



	A	B	C	D	E
A	0	1	1	1	0
B	1	0	0	1	1
C	1	0	0	1	0
D	1	1	1	1	1
E	0	1	0	1	0



	E1	E2	E3	E4	E5	E6	E7	E8
A	1	1	-1	0	0	0	0	0
B	-1	0	0	1	0	1	0	0
C	0	-1	0	0	1	0	0	0
D	0	0	1	-1	-1	0	1	1
E	0	0	0	0	0	-1	-1	0

Recap: Graph Algorithms

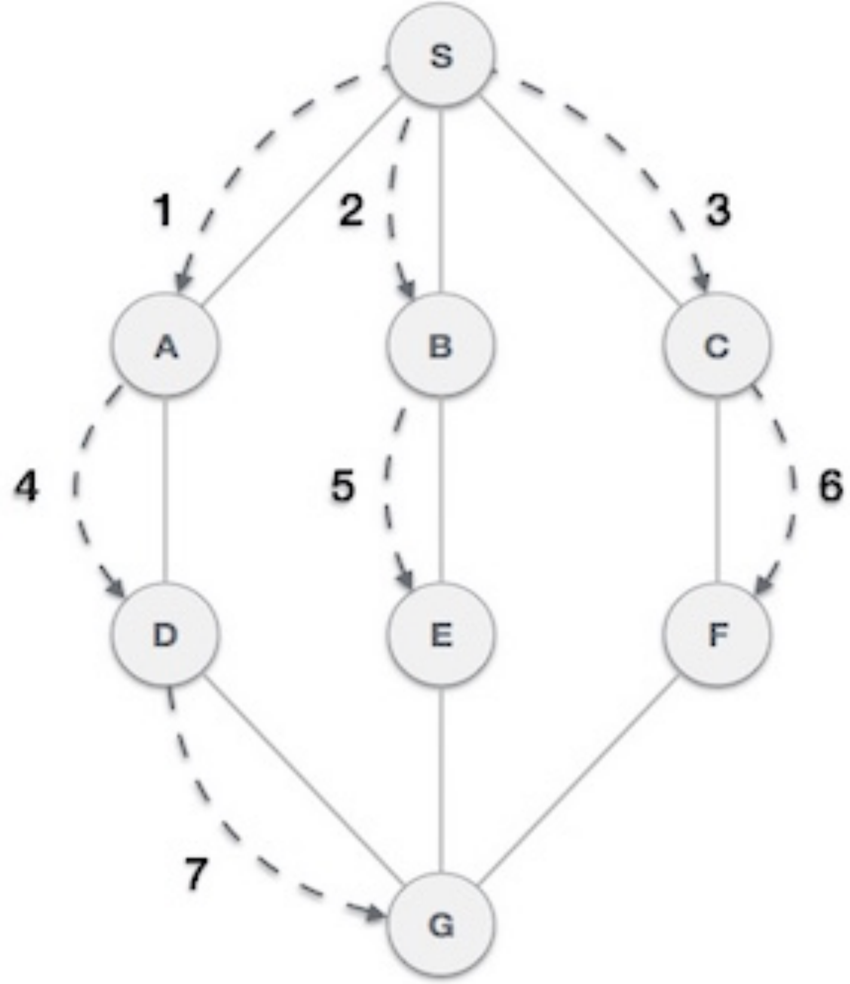
- Problems could be better represented and solved using Graph Algorithms
  - We focus on a traversal problem:
    - Given G=(V, E) and a vertex v
    - Find all w in V such that w connects v

- Two Graph Traversals:
  - Breadth First Search (BFS)
  - Depth First Search (DFS)
    - Getting the connected components

- Next: Minimum Spanning Trees ➔ ➔ ➔

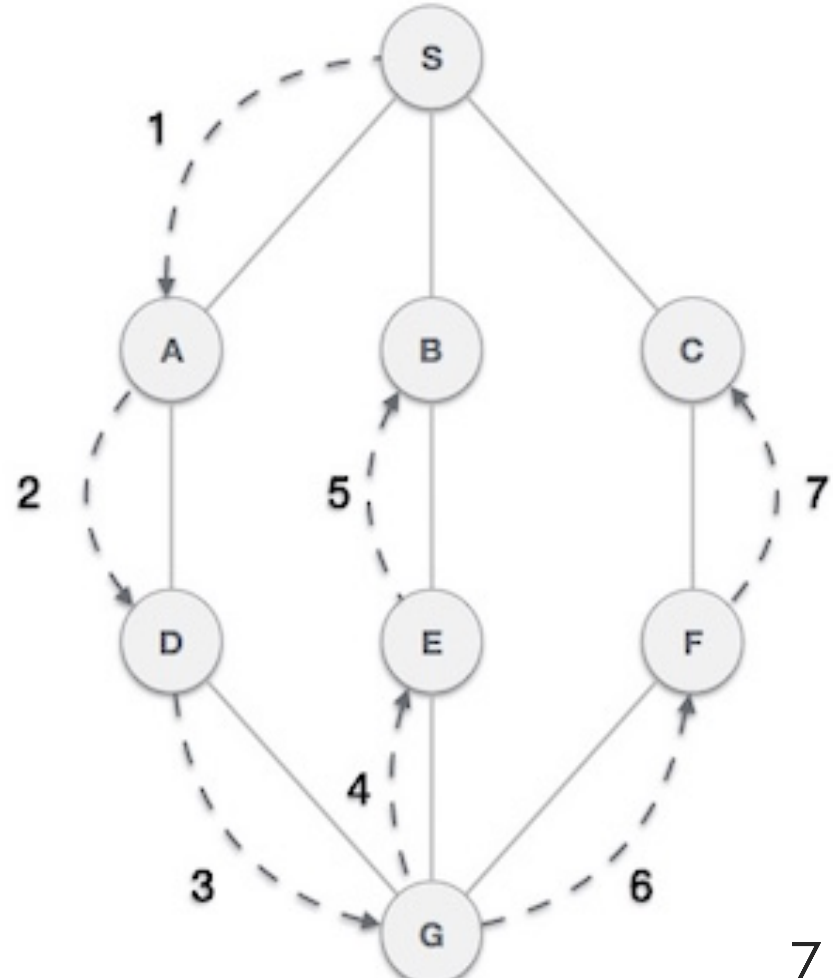
Breadth First Search

- Look at the Graph
- Breadth ward Search
- Visit all nodes in the next level and then move to another level to start the search for.



Depth First Search

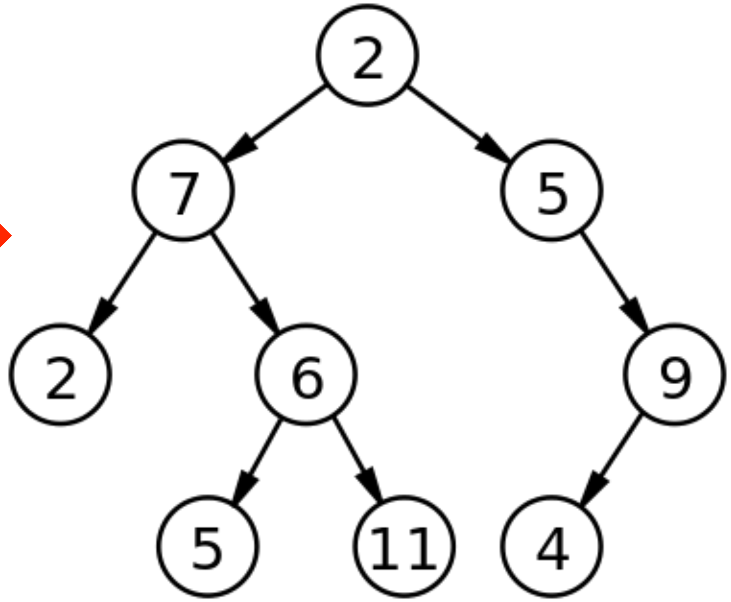
- Look at this example
- Depth ward Search
- Visit all nodes in that chosen path until addition of nodes does not create a cycle.



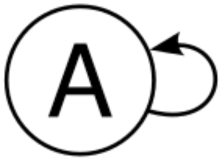
What is a tree?

- Abstract Data Type

Is this a tree?? → → ➔



- Tree??

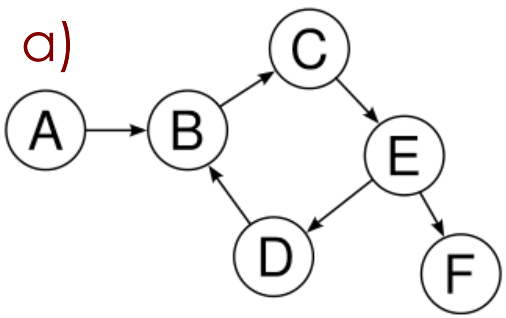


- A collection of nodes (starting at a root node), where each node is a data structure consisting of a value, together with a list of references to nodes (the "children")
- A tree is a data structure made up of nodes or vertices and edges without having any cycle

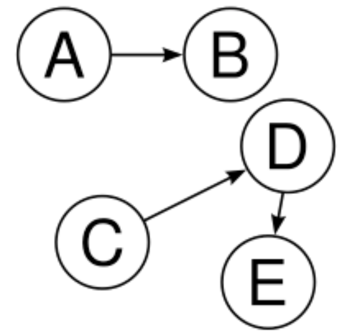
Trees – A Few Examples

- Look at the following trees

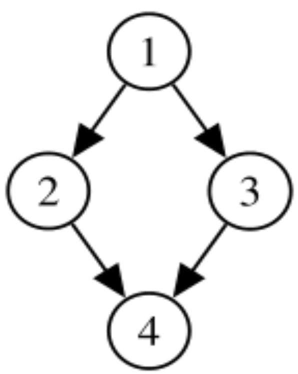
a)



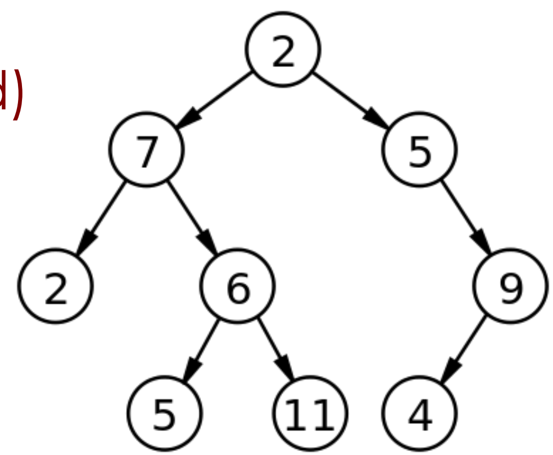
b) More than one root



c)



d)



Graph (Tree) Traversal

- Types of Traversals
- Pre-order

- In-order

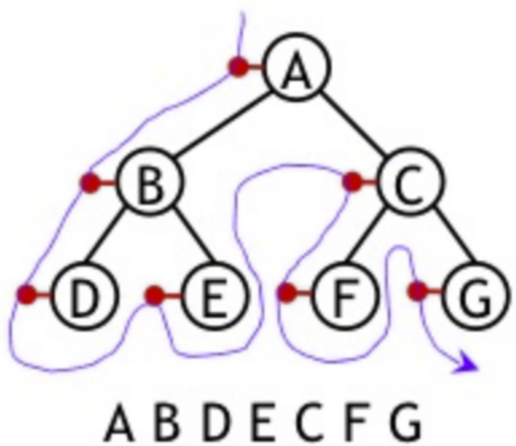
- Post order

- Level Ordering

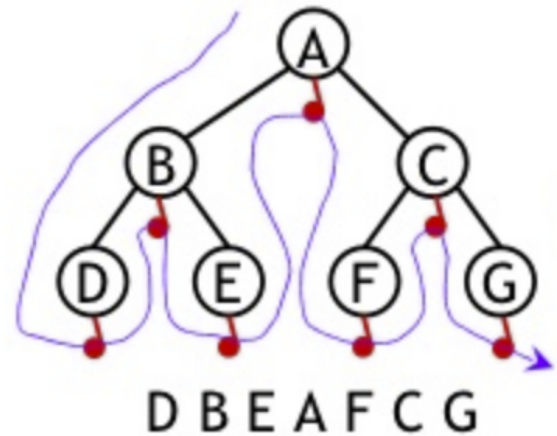
Tree Traversal - Examples

- A few examples

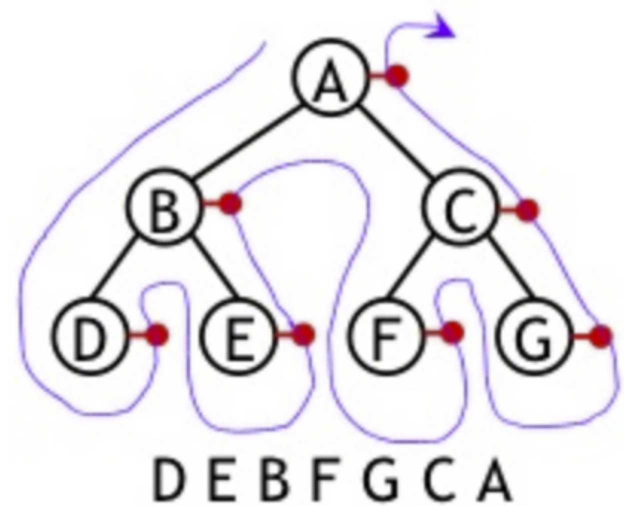
Pre-order



In-order

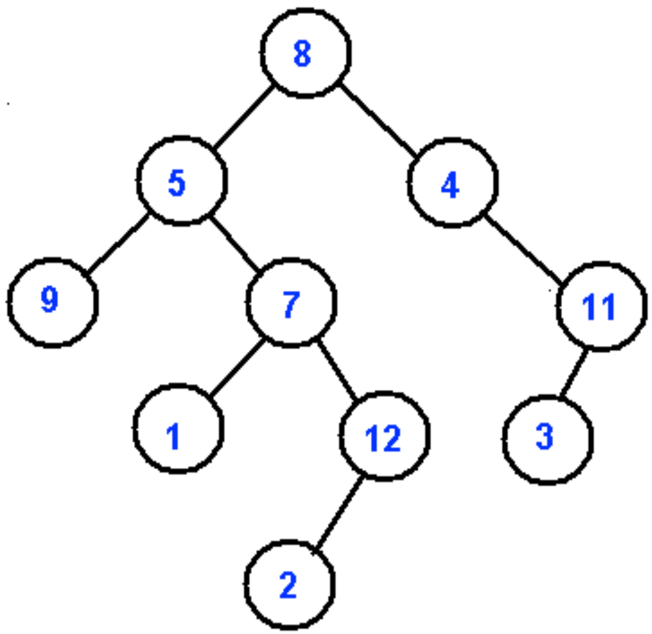


Post-order



Tree Traversal - Illustration

- Look at the tree



Pre-order
8, 5, 9, 7, 1, 12, 2, 4, 11, 3

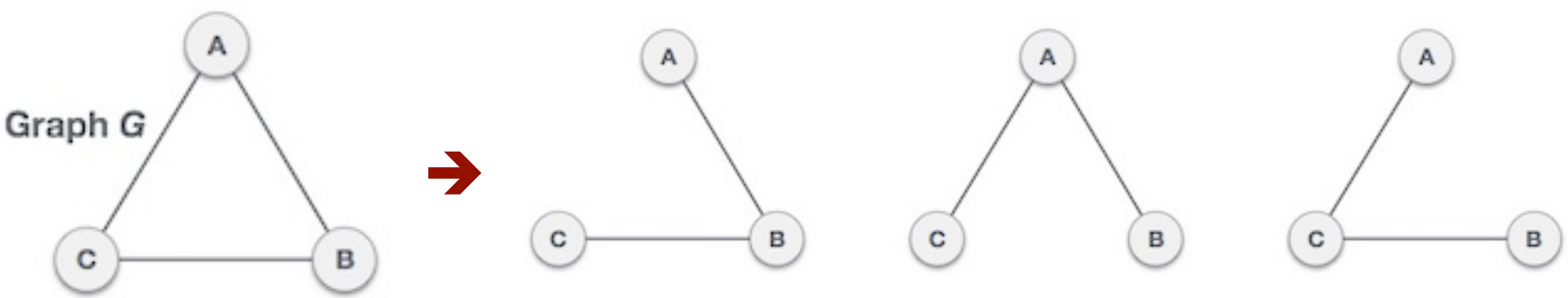
In-order
9, 5, 1, 7, 2, 12, 8, 4, 3, 11

Post-order
9, 1, 2, 12, 7, 5, 3, 11, 4, 8

Level-order
8, 5, 4, 9, 7, 11, 1, 12, 3, 2

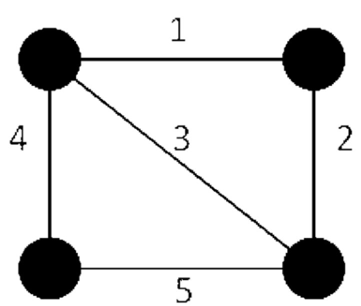
Spanning Trees

- A spanning tree is a subset of Graph G, which has all the vertices covered with minimum possible number of edges
- A spanning tree does not have cycles and it cannot be disconnected
- Every connected and undirected Graph G has at least one spanning tree

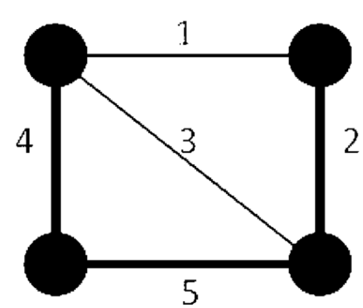


Minimum Spanning Trees

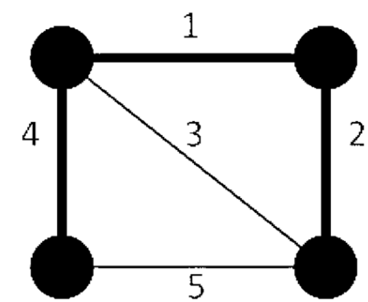
- A minimum spanning tree is a subset of the edges of a connected, edge-weighted directed graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight



Undirected Graph



Spanning Tree
Cost = 11(=4+5+2)



Minimum Spanning Tree
Cost = 7(=4+1+2)

Kruskal Algorithm

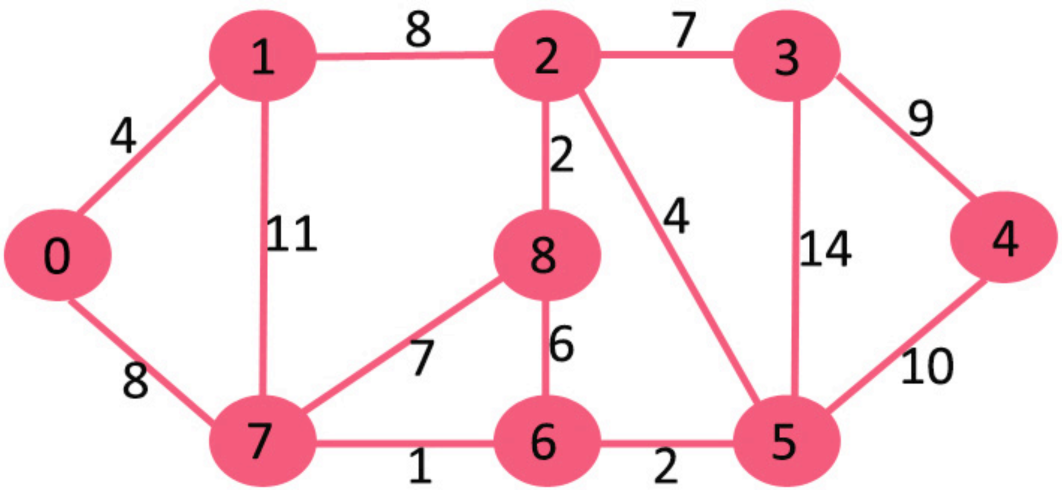
The basic Idea:

- Sort all the edges in non-decreasing order of their weight.
- Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If cycle is not formed, include this edge. Else, discard it.
- Repeat the above steps until there are (V-1) edges in the spanning tree.

Sort Edges by weights

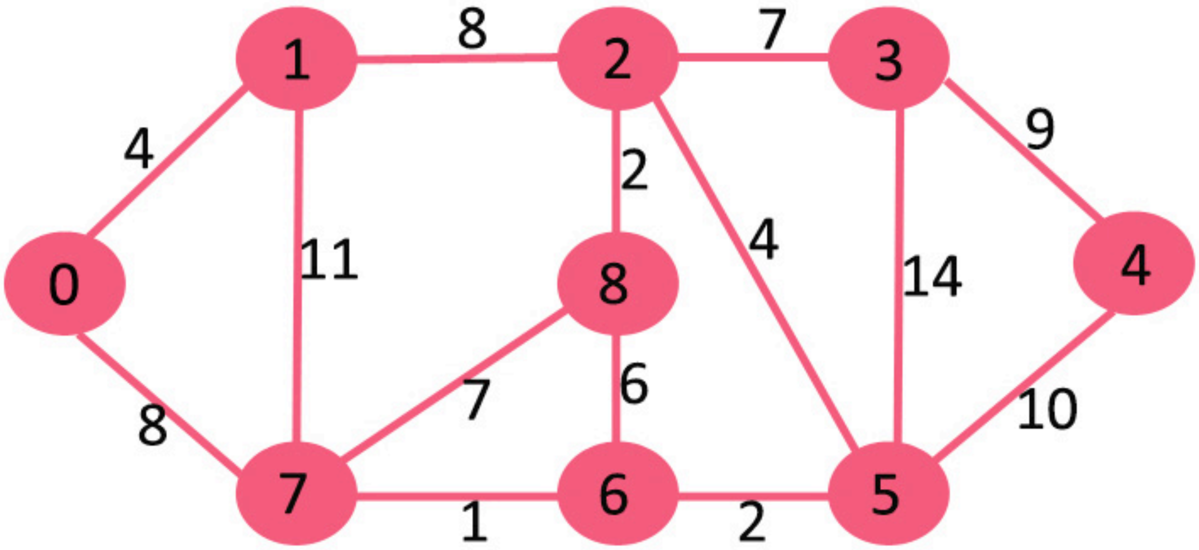
- Sorted Weights

Weight	Src	Dest
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5



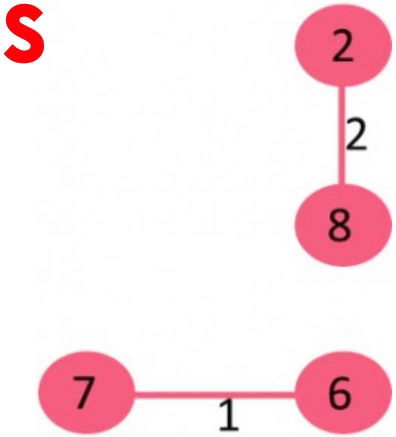
Kruskal Algo- Illustration

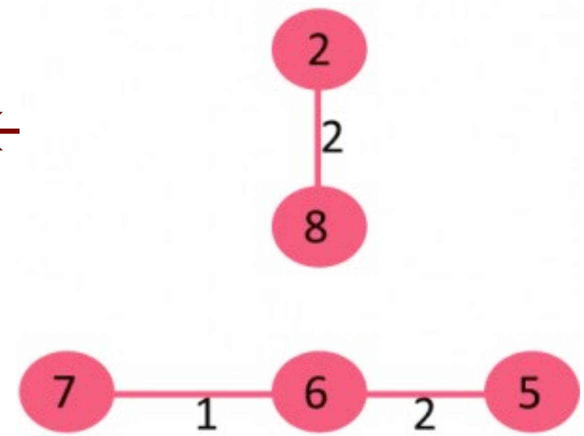
- Look at the following graph:

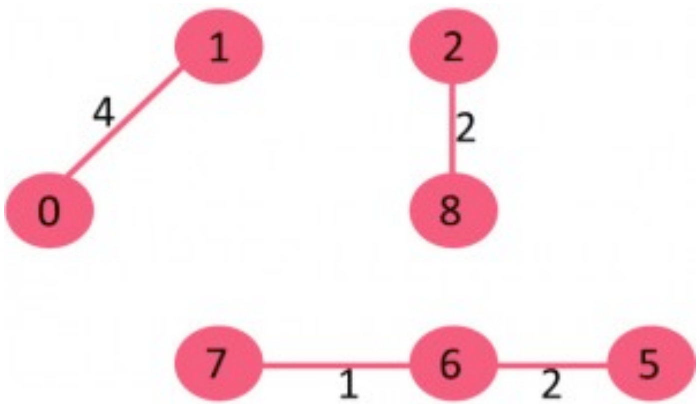


Kruskal Algo- Steps

- Steps:

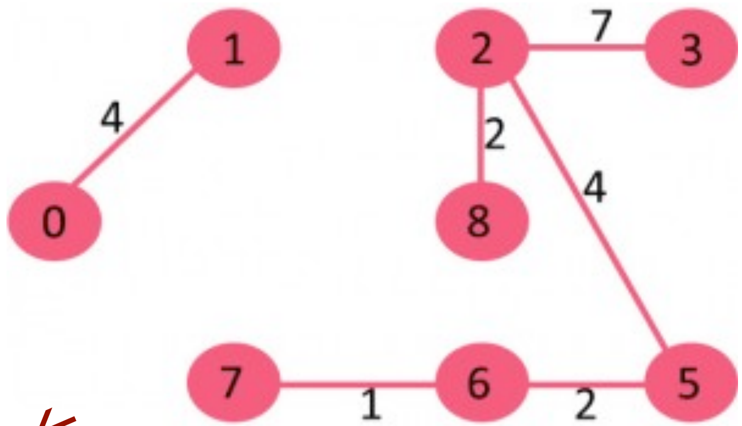


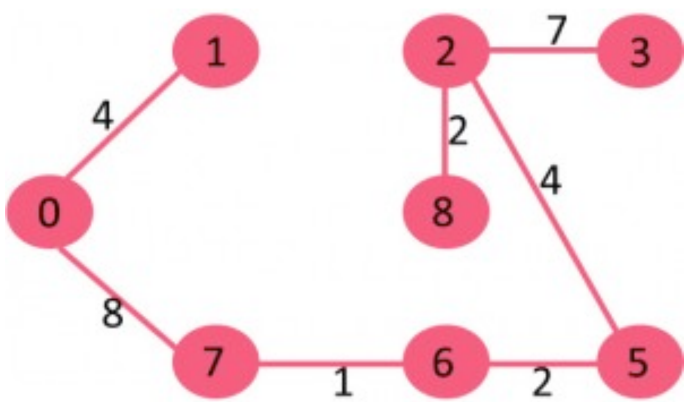


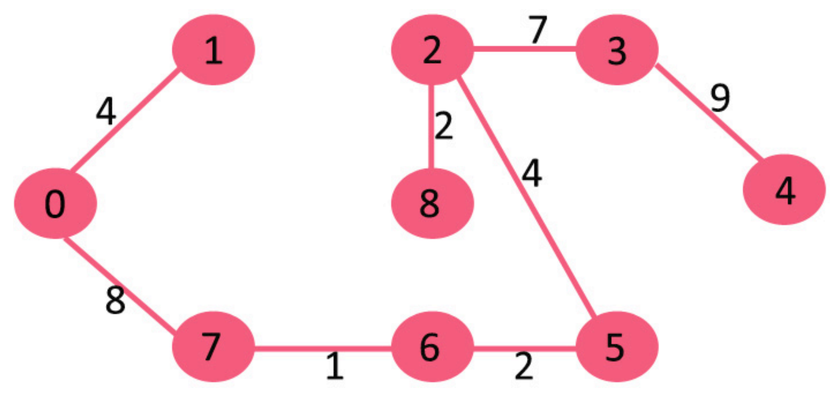


Kruskal Algo- Steps

- Steps:



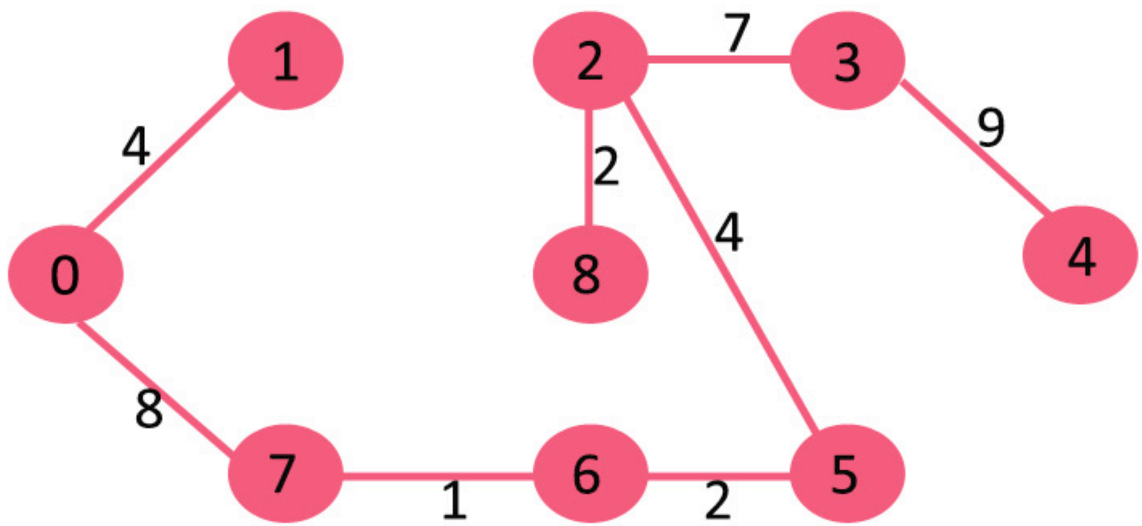




19

Kruskal Algo- Steps

- Minimum cost:



$$4 + 8 + 1 + 2 + 4 + 2 + 7 + 9$$
$$= 37 \text{ (Correct ?!)}$$

Kruskal Algo - Complexity

- Time Complexity:
  - O(ElogE) or O(ElogV)

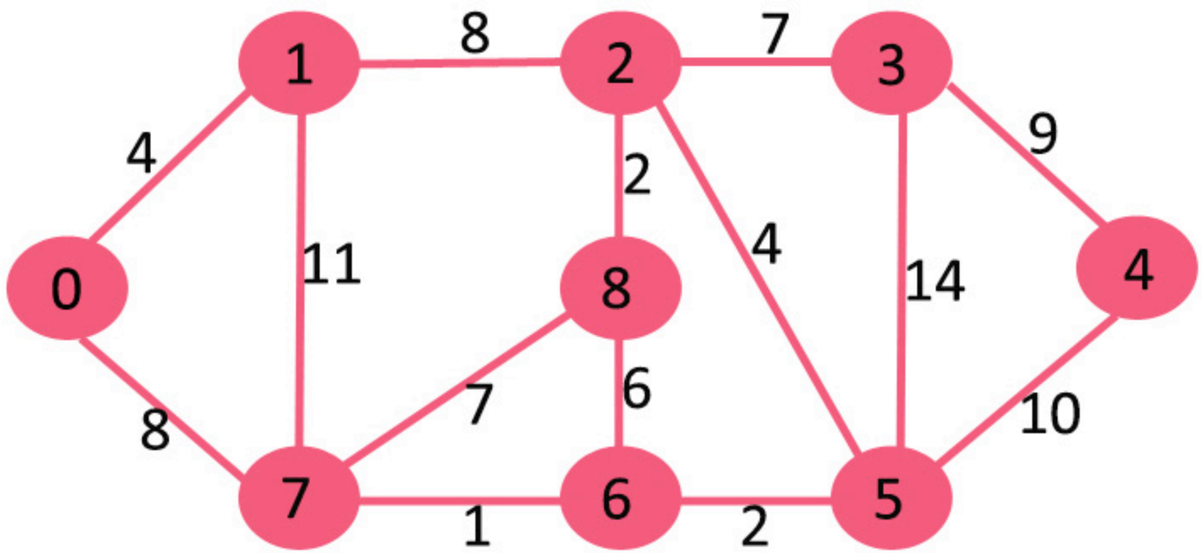
  - Sorting of edges takes O(ELogE) time
  - After sorting, we iterate through all edges and apply find-union algorithm
  - The find and union operations can take atmost O(LogV) time
  - So overall complexity is O(ELogE + ELogV) time
  - The value of E can be atmost $O(V^2)$, so O(LogV) are O(LogE) same
  - Therefore, overall time complexity is O(ElogE) or O(ElogV)

Prim's Algorithm for MST

1) Create a set *mstSet* that keeps track of vertices already included in MST.
2) Assign a key value to all vertices in the input graph. Initialize all key values as INFINITE. Assign key value as 0 for the first vertex so that it is picked first.
3) While mstSet doesn't include all vertices
   a) Pick a vertex *u* which is not there in *mstSet* and has minimum key value.
   b) Include *u* to mstSet.
   c) Update key value of all adjacent vertices of *u*. To update the key values, iterate through all adjacent vertices. For every adjacent vertex *v*, if weight of edge *u*-*v* is less than the previous key value of *v*, update the key value as weight of *u*-*v*

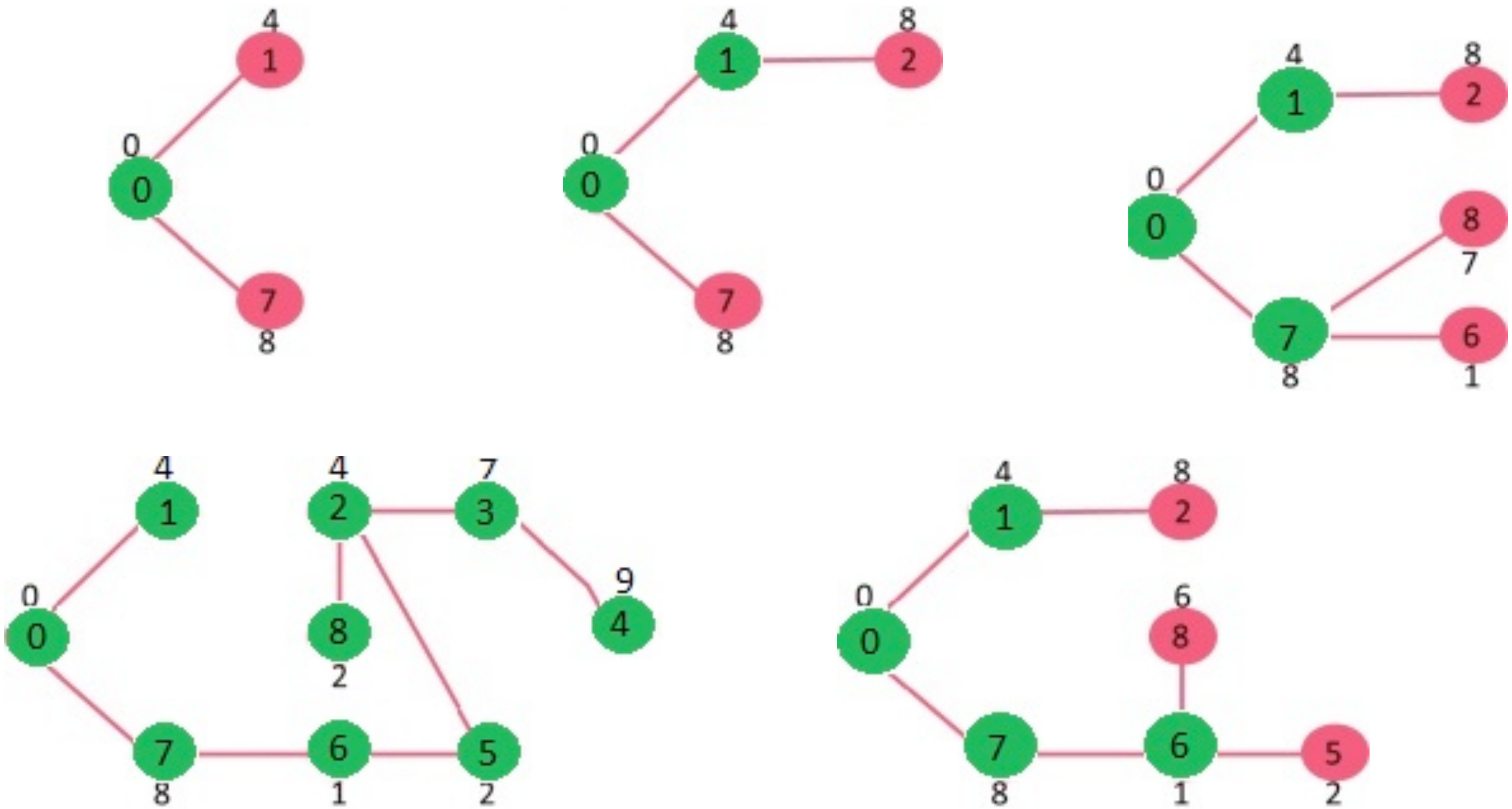
Prim Algo- Illustration

- Look at the following graph:



Prim Algo- Illustration

- Steps:

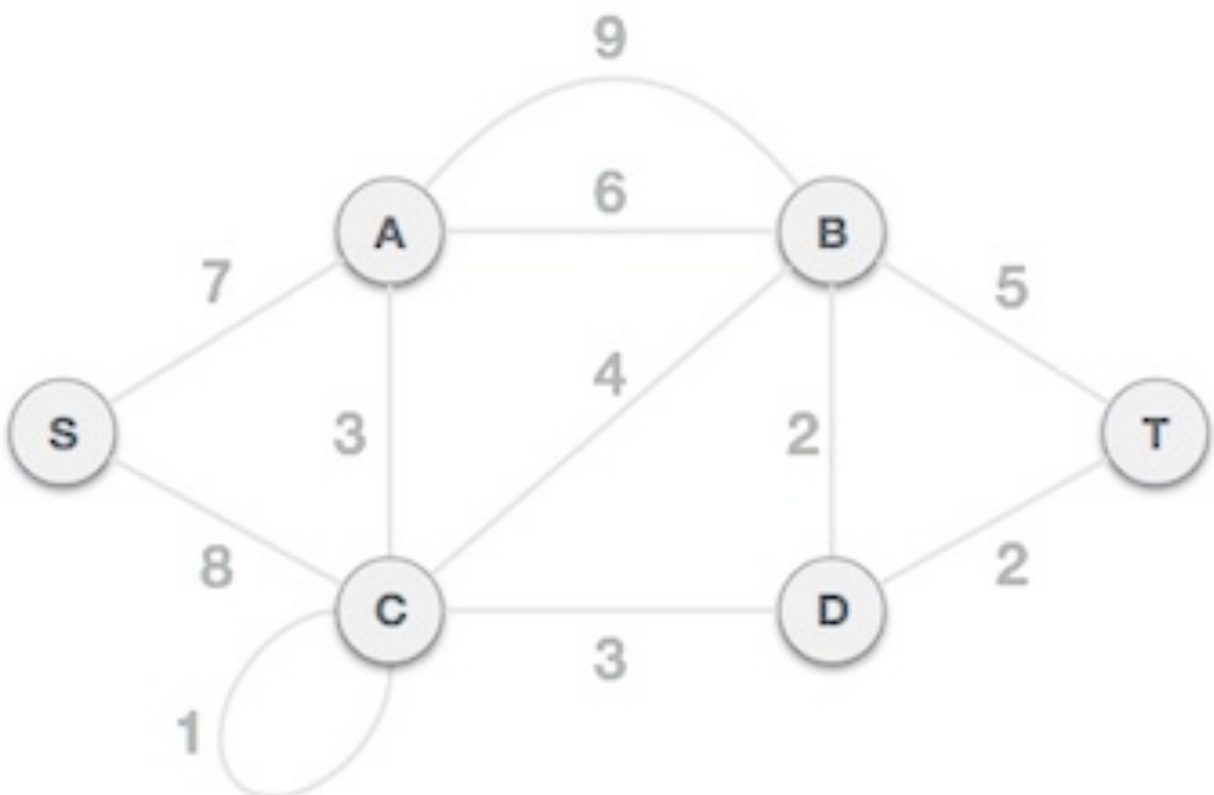


Prim Algo - Simpified

- Remove all loops and parallel edges
- Choose any arbitrary node as root node
- Check outgoing edges and select the one with less cost
- Can you apply the above steps in an example

Prim Algo - Exercise

- Try Prims algorithm



Prim Algo - Complexity

- Time Complexity of the above program is $O(V^2)$
- If the input graph is represented using adjacency list, then the time complexity of Prim's algorithm can be reduced to $O(E \log V)$ with the help of binary heap

Help among Yourselves?

- **Perspective Students** (having CGPA above 8.5 and above)
- **Promising Students** (having CGPA above 6.5 and less than 8.5)

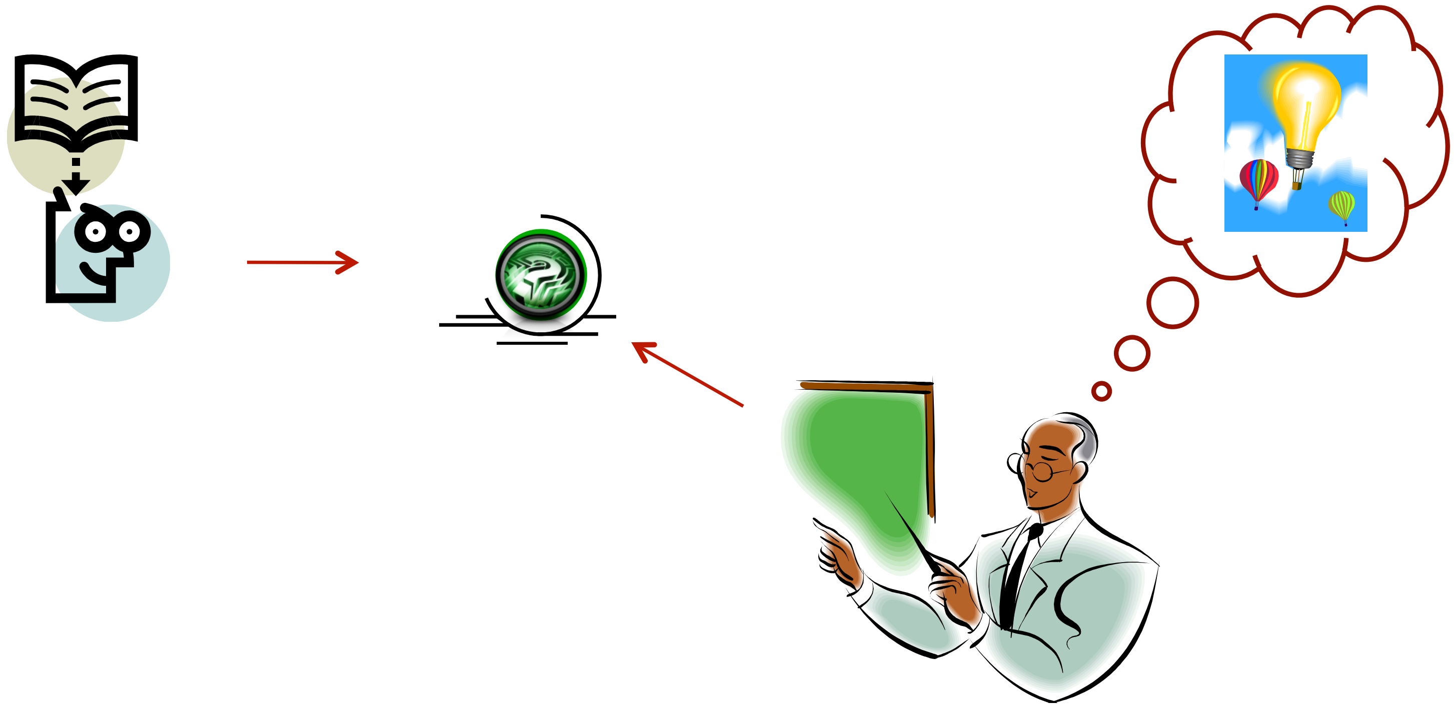
- **Needy Students** (having CGPA less than 6.5)
  - Can the above group help these students? (Your work will also be rewarded)

- You may grow a culture of **collaborative learning** by helping the needy students

Assistance

- You may post your questions to me at any time
- You may meet me in person on available time or with an appointment
- TA s would assist you to clear your doubts.
- You may leave me an email any time (email is the best way to reach me faster)

Thanks …



… Questions ???