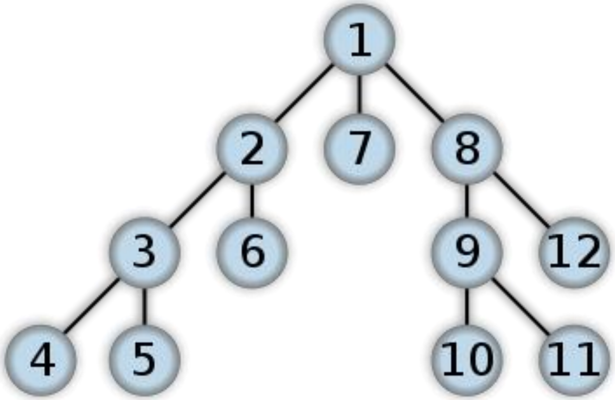
Build X : Algorithms
Week 6

- Week 5 - Winner && Challenges
- Prim's Algorithm
- Kruskal
- Warshall
- Roy-Floyd
- Short Intro to Heap Data Structure (I know you had DST today)
- Prim's Algorithm using Heap

Week 5 - Challenges

Depth-First Search
-100pt-

Basic operation for graphs
Uses the LIFO(Last in, First out) technique



Week 5 - Challenges

Shortest Path in a Graph
-100pt-

Dijkstra's Algorithm* – O(e*log v)

*was discussed last week

Is it useless?

- Why do we need to know about graphs if we have STL(C++), Java Objects, etc. that do that?
- Why should we create data structures if they are already implemented in libraries?
- Why learn certain algorithms ?
- Why do I need X and Y ?

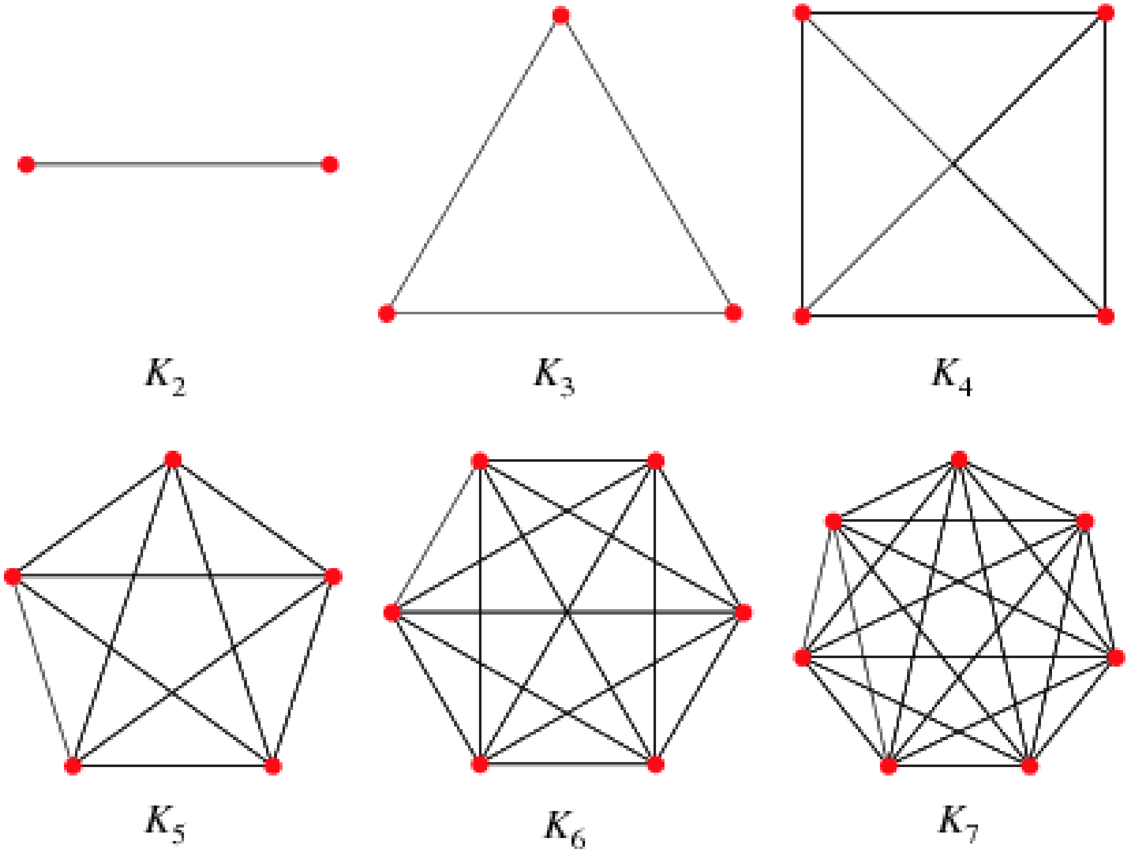
Why?

- Keeps your mind entertained
- Not every problem can be solved with the same solution
- You will know the advantages/disadvantages of a structure/algorithm
- You will know for sure basic operations and their complexities

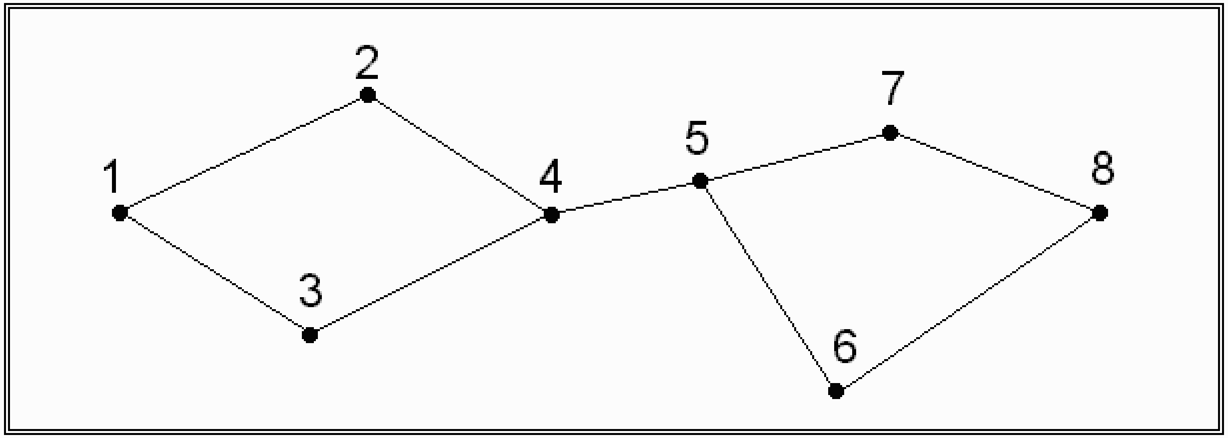
Graph Properties

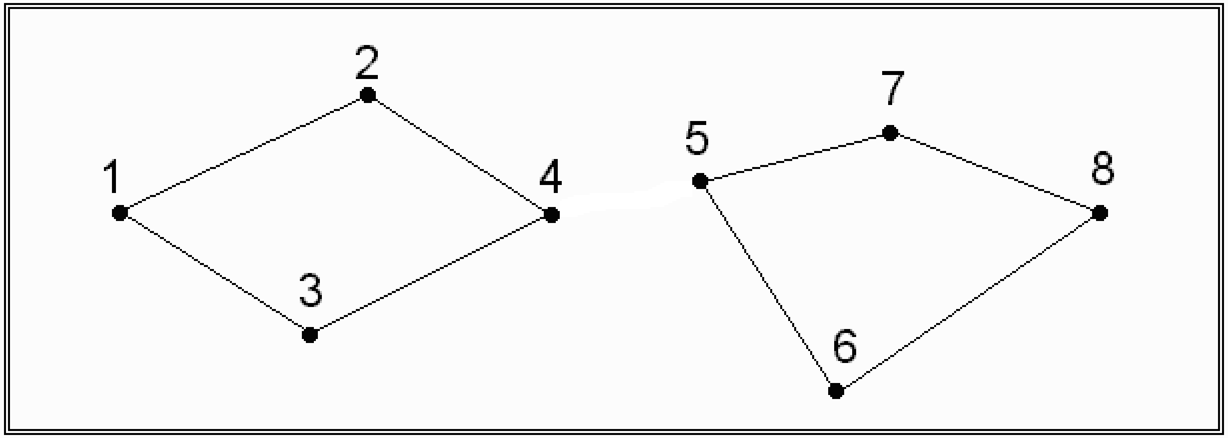
- Complete/Incomplete
- Connected/Disconnected
- Directed/Undirected
- Partial graph
  - Obtained by eliminating some edges
- Sub-graph
  - Obtained by eliminating some nodes and their corresponding edges

Complete



Connected/Disconnected





Directed/Undirected

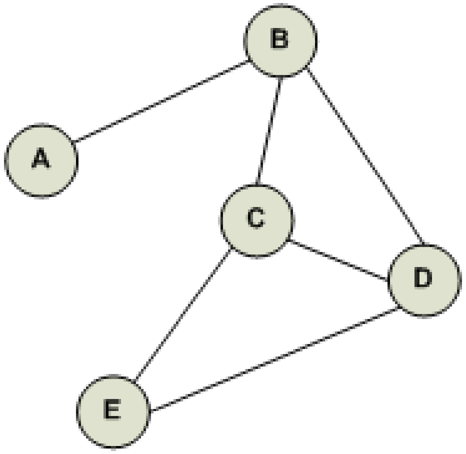


Fig 1. Undirected Graph

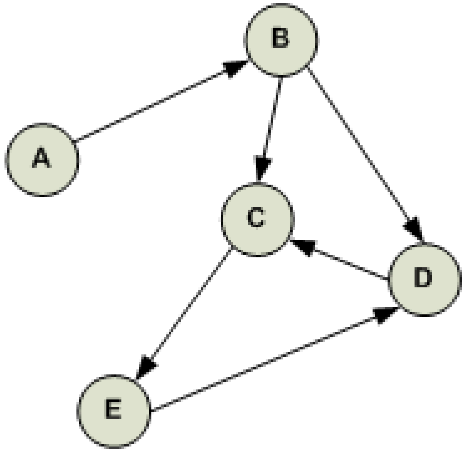
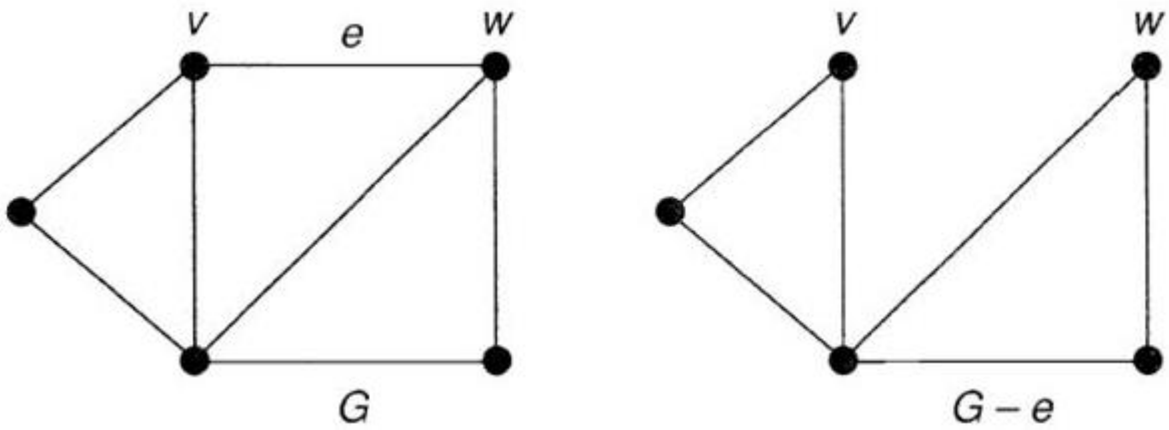
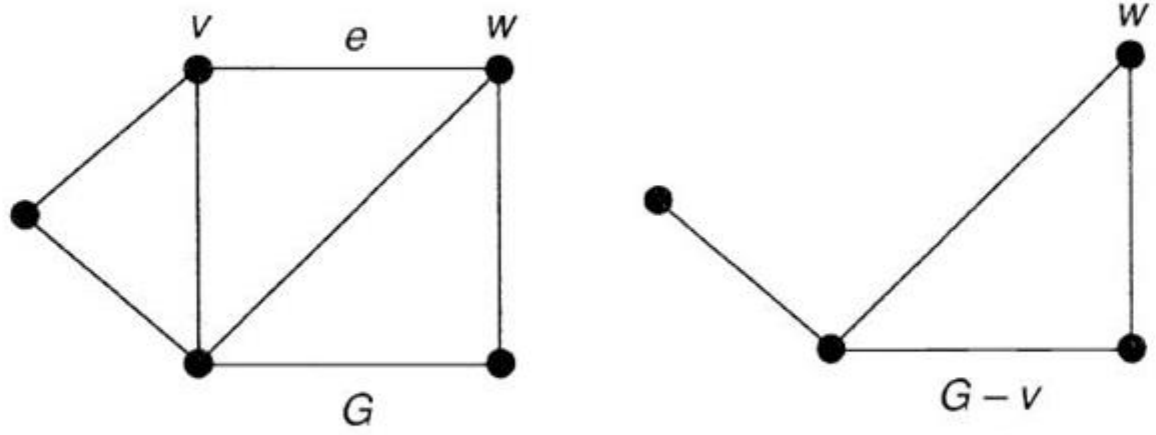


Fig 2. Directed Graph

Partial Graph



Sub-graph



Prim's Algorithm
-Minimal Cost of a Partial Tree-

- Description:
  - Given a connected graph, Prim's Algorithm, builds a partial tree of minimal cost.

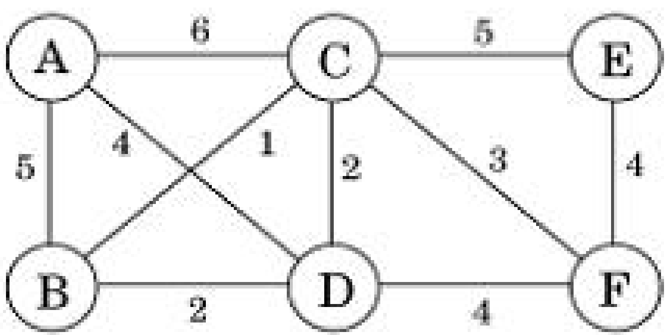
- Definition : A partial tree is a partial connected graph with no cycles.

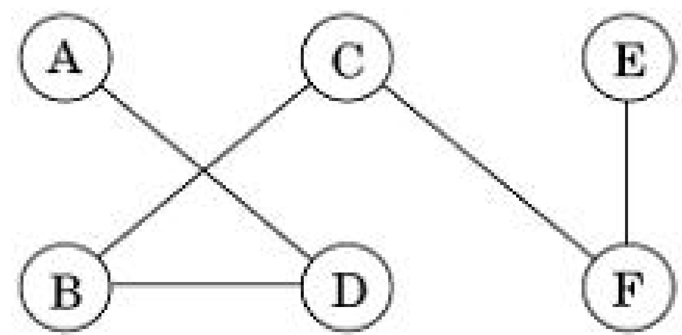
- Time Complexity : $O(n^2)$

Steps

- 1. Select a root (Can be any node from the graph)
- 2. Chose the edge with the minimal cost that exits from current selected node.
- 3. Add the new node to the list
- 4. Repeat from step 2 until all nodes are visited

Example



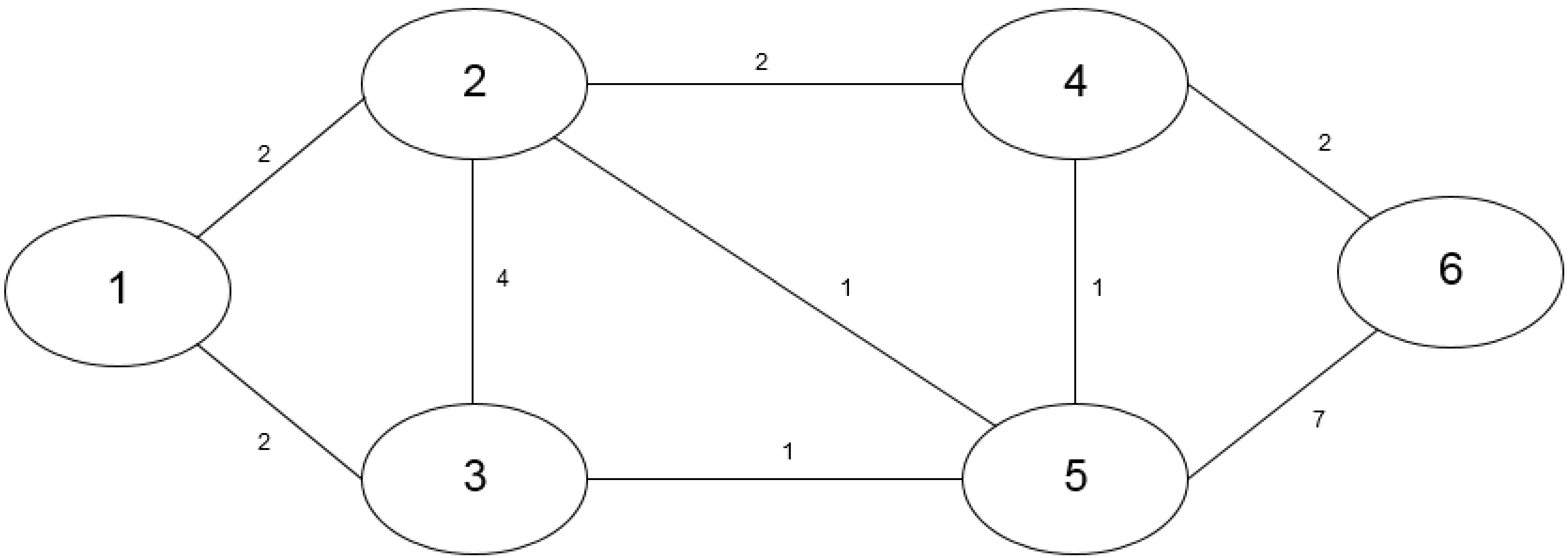


Kruskal's Algorithm -Minimal Cost of a Partial Tree-

- Description:
  - Given a certain graph, Kruskal's Algorithm, builds a partial tree of minimal cost.

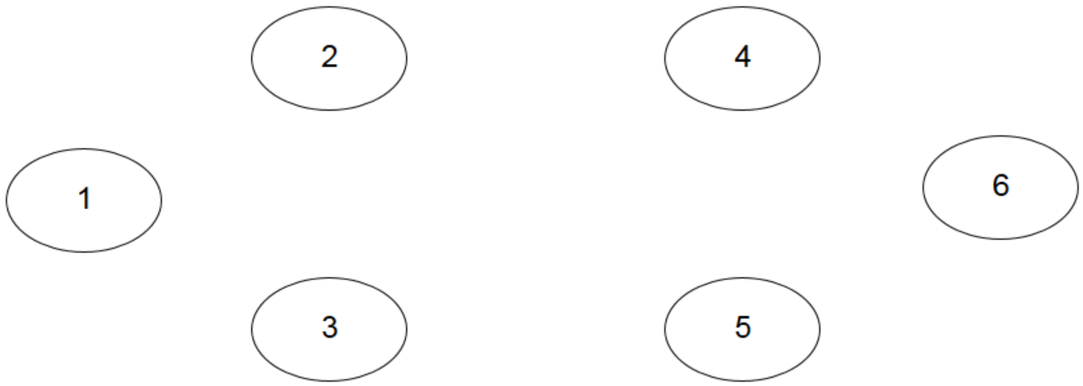
- The algorithm does not select a root as in Prim's version. It searches only for the smallest edges and creates a tree by adding other trees to it.

Example

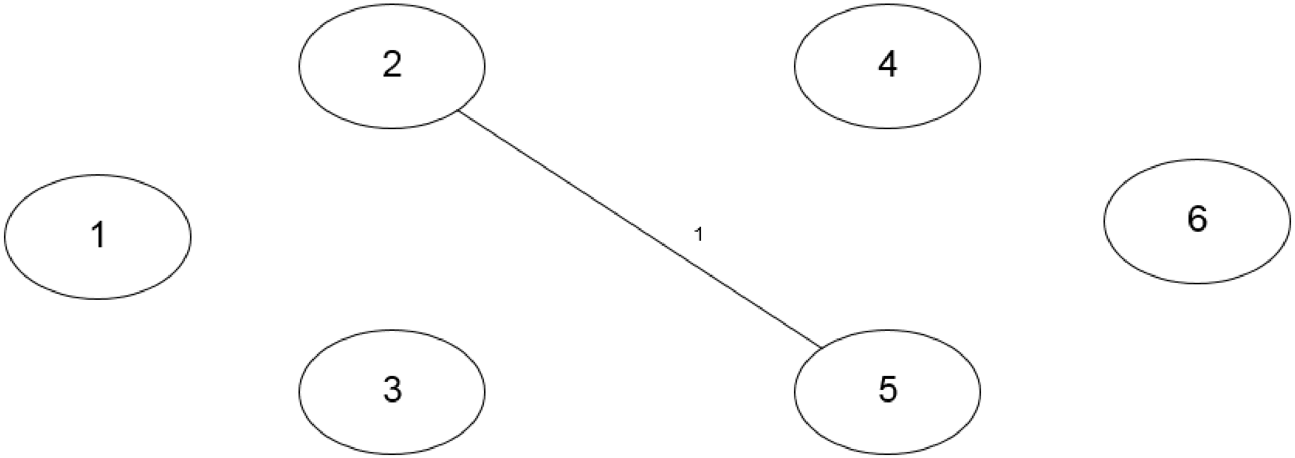


Steps

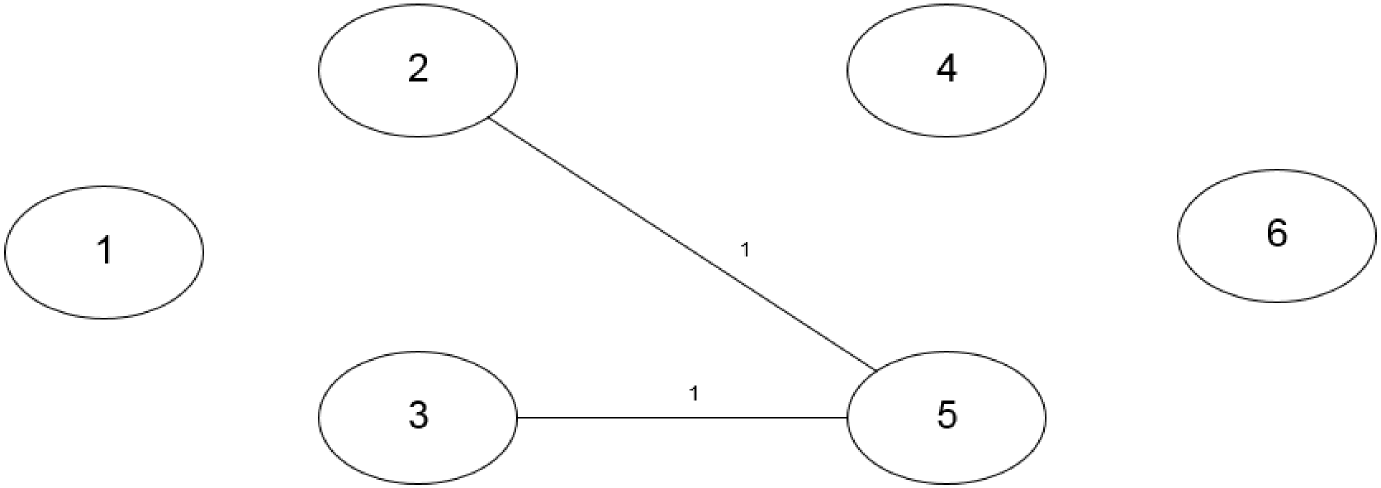
- 1.



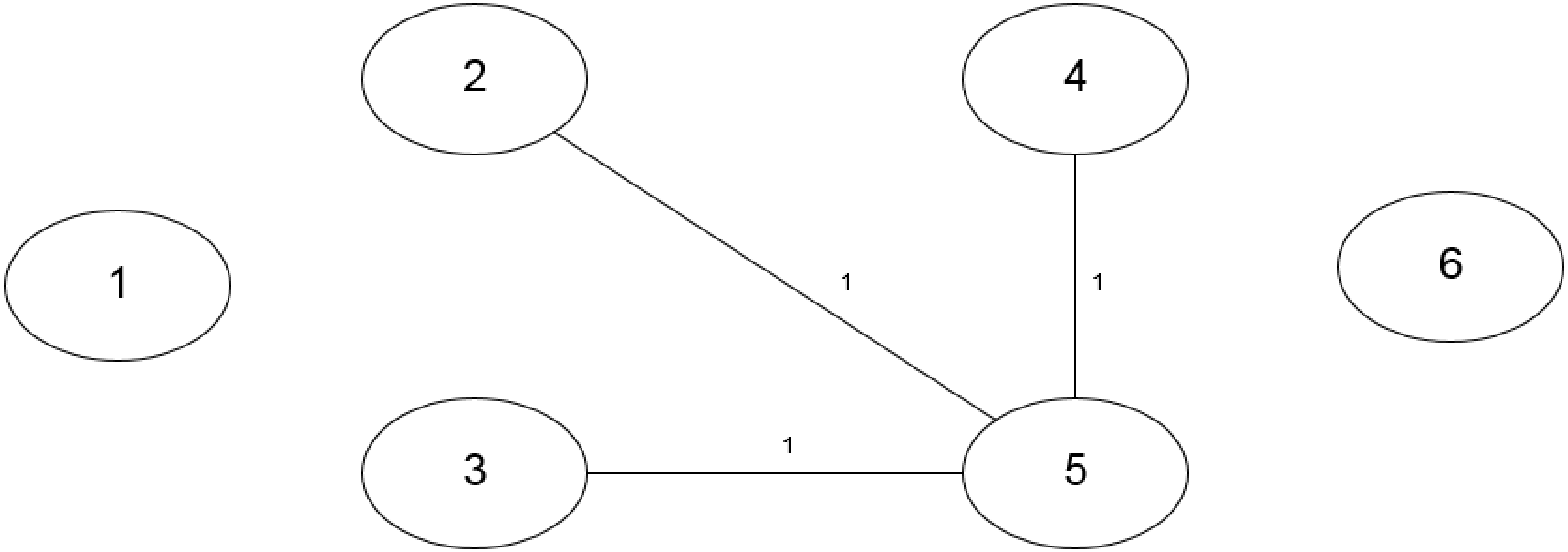
- 2



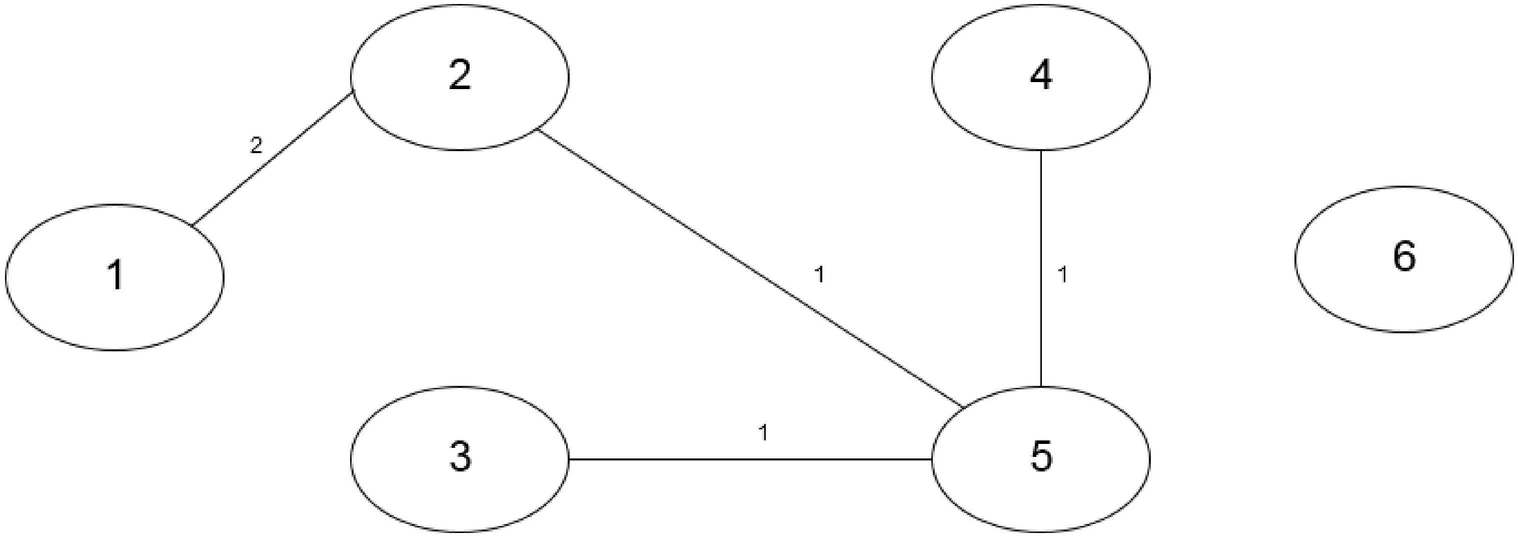
- 3.



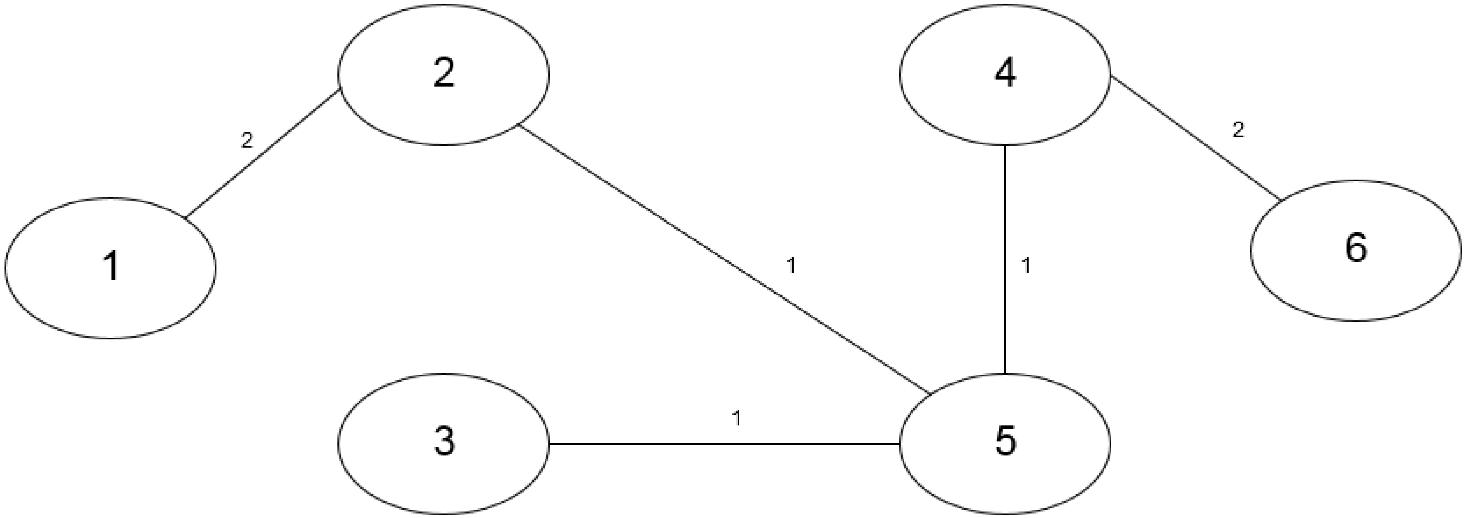
- 4.



- 5.



- 6.



Warshall's Algorithm

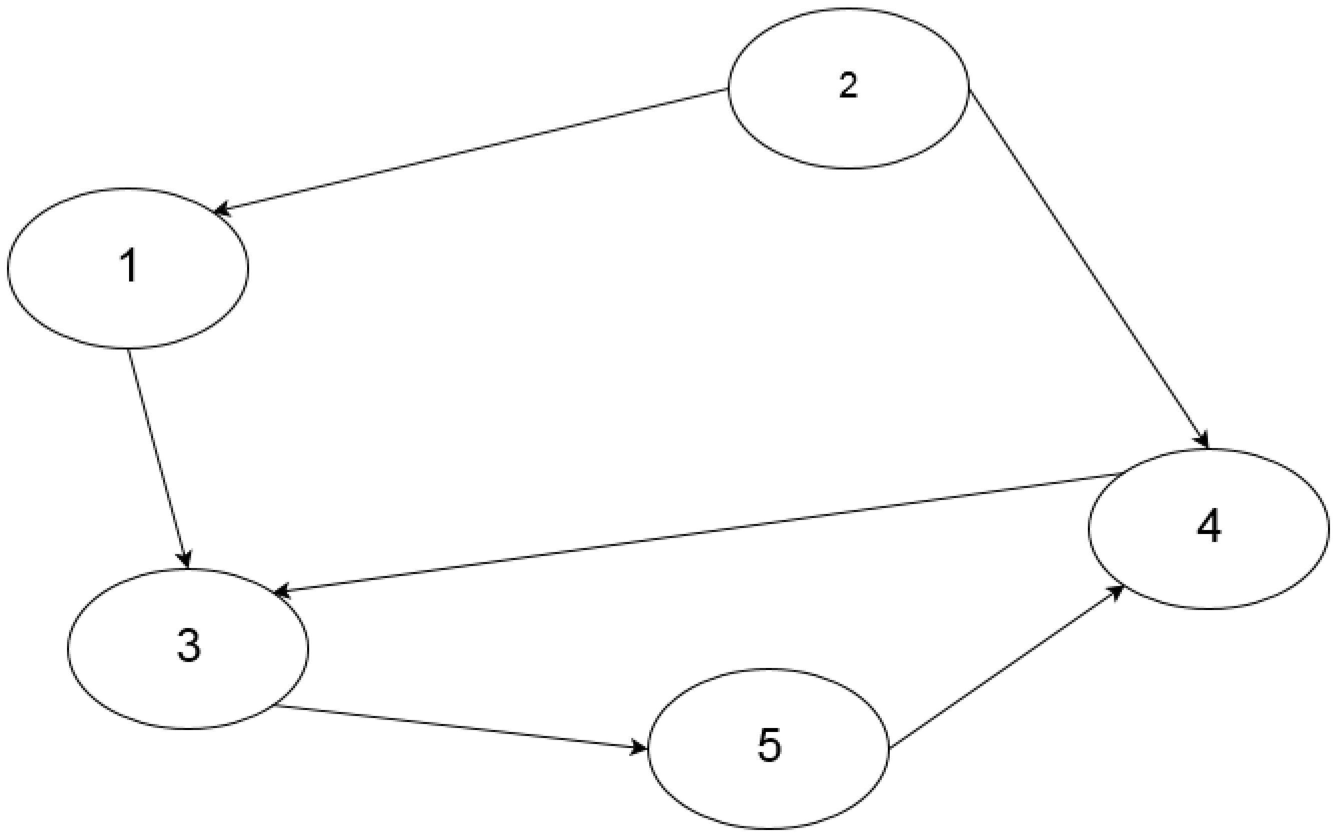
- Description:
  - Assuming we have a directed graph, for every given pair of nodes (x,y) the algorithm determines if there is a path between them.
- Properties:
  - Uses a matrix to store data
  - Returns only true/false if there is a path or not
  - Time Complexity : $O(n^3)$

Steps

- 1. Initialize the matrix with given edges.

- 2. For every single node go through the matrix and apply the general rule
  - D_k [i][j] = D[i][j] **OR** (D[i][k] **AND** D[k][j])

Example



Roy-Floyd's Algorithm

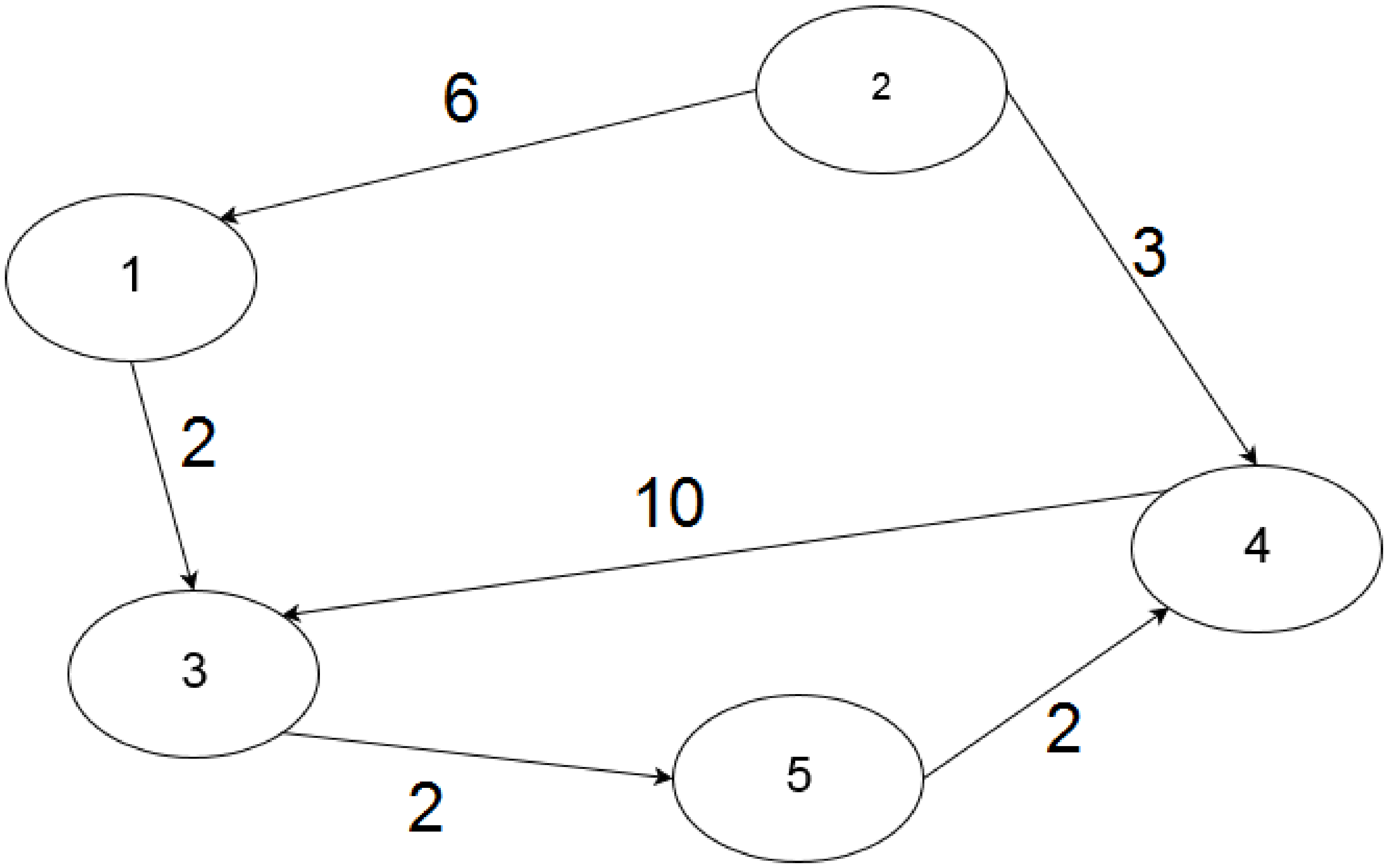
- Description:
  - Given a directed graph with real costs associated to the margins, the algorithm finds the minimum cost to go from node x to node y for each pair of nodes (x,y) of the Graph.
- Disadvantages
  - Does not return any information related to the path itself

Steps

- 1. Create the initial matrix of values
  - Contains the edges that are given in the graph

- 2. For every single node go through the whole matrix and apply the general rule
  - $D_k[i][j] = \min(D[i][j], \{ D[i][k] + D[k][j] \})$

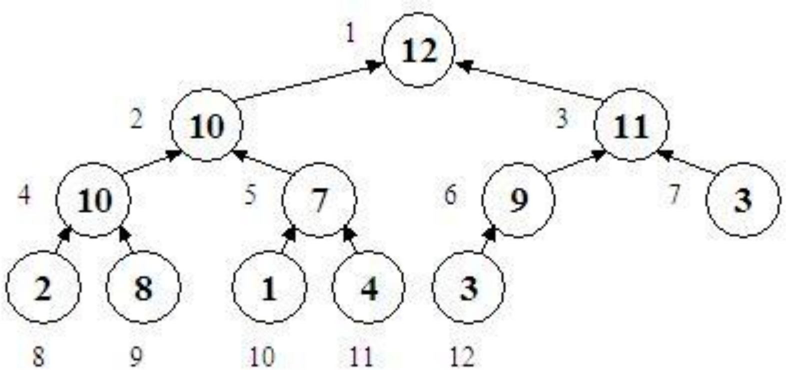
Example



Heap

- What is a heap?
  - We name a heap a vector/list that can be viewed as a binary tree and meets some properties

Yeah, yeah you had DST.



12 | 10 | 11 | 10 | 7 | 9 | 3 | 2 | 8 | 1 | 4 | 3

Properties of a Heap

- It is a binary tree
- All levels must be completed except the last one
  - height of a heap = [$\log_2 N$]
  - Parent of a node > 1 is [node / 2]

- Nodes are in a certain order:
  - Min to max

    or
  - Max to min

Advantages

- Searching for min/max – O(1)
- Creating a new heap from a given vector – O(N)
- Eliminate one element – O(log N)
- Insert a new element – O(log N)
- Sorting elements – Heapsort O(N * log N)

Not recommended for:

- Searching for a specific element – O(N)
  - HashMaps – O(1)

Prim’s Algorithm using Heap

- Prim : O (n^2)
- Prim with Heaps : O(n*log n)

- Instead of using a vector and always insert the element in the array/ search for the minimum(O(n)) use a heap structure.

- Insertion in heap : O(log n)
- Finding the minimum : O(1) - constant

Next week

Special Guest : **Dr. Andrew Coles**

-Algorithms based on Trees-

14^{th} Of March 2016

Special Guest : **Christopher Hampson**

-Logic behind Algorithms-