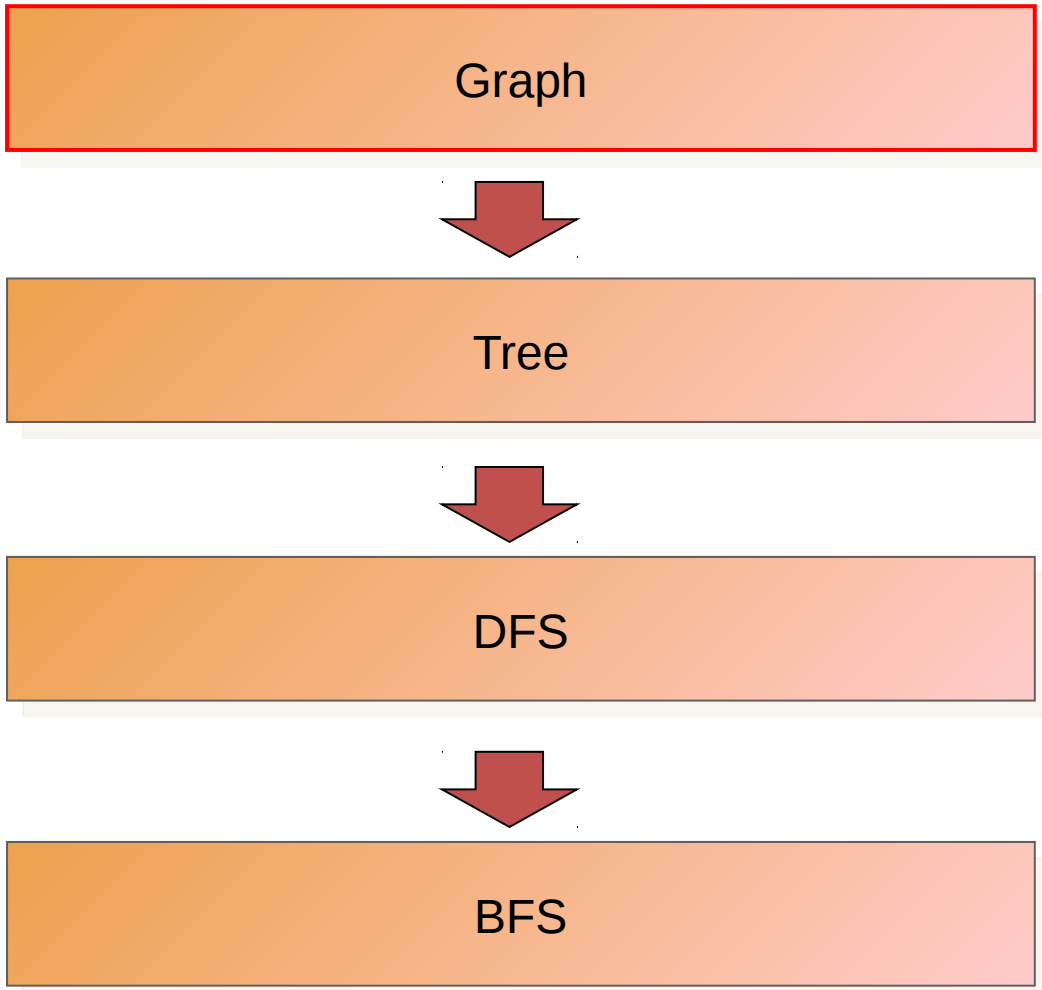
LAO CHON LAM

nckuacm@imslab.org

Department of Computer Science and Information Engineering
National Cheng Kung University
Tainan, Taiwan

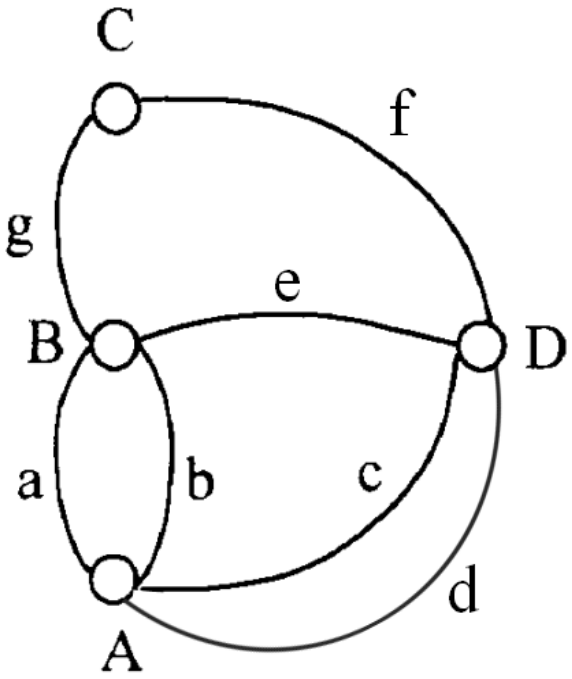
Made By louis7340 & Edit by

Outline



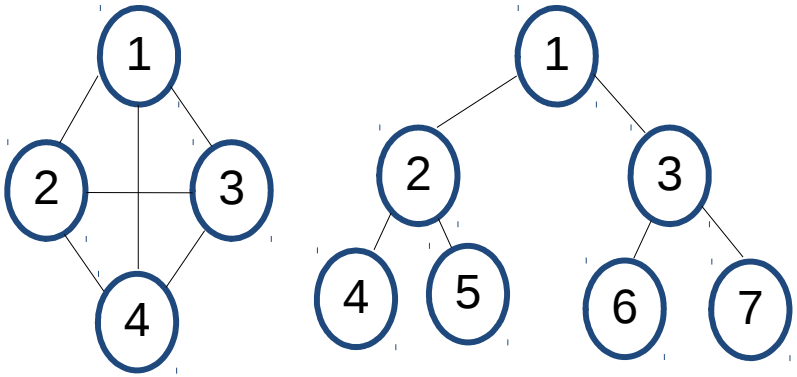
Graph

- Vertex (V)
  - A, D, B, D
- Edge (E)
  - BC, CD,…
- degree (deg)
  - The Branch of a Vertex
- Path (P)
  - Non-duplicate Vertex Connected Seque
    - ADCB
- Cycle (C)
  - A Cycle whose Two Ending Points are th

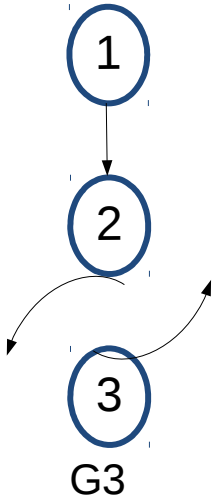


Graph

- Undirected Graph – G1, G2

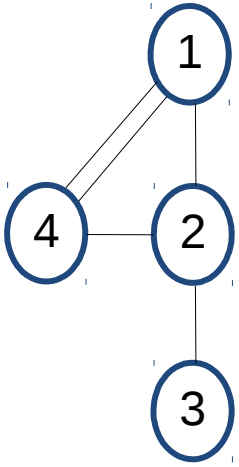


- Directed Graph – G3
  - deg: in-degree and out-degree



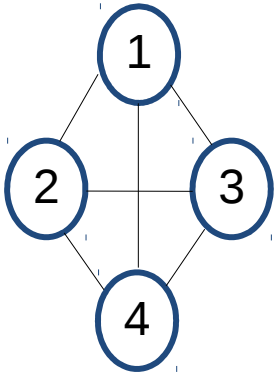
Graph

- Multiple Edge



G4

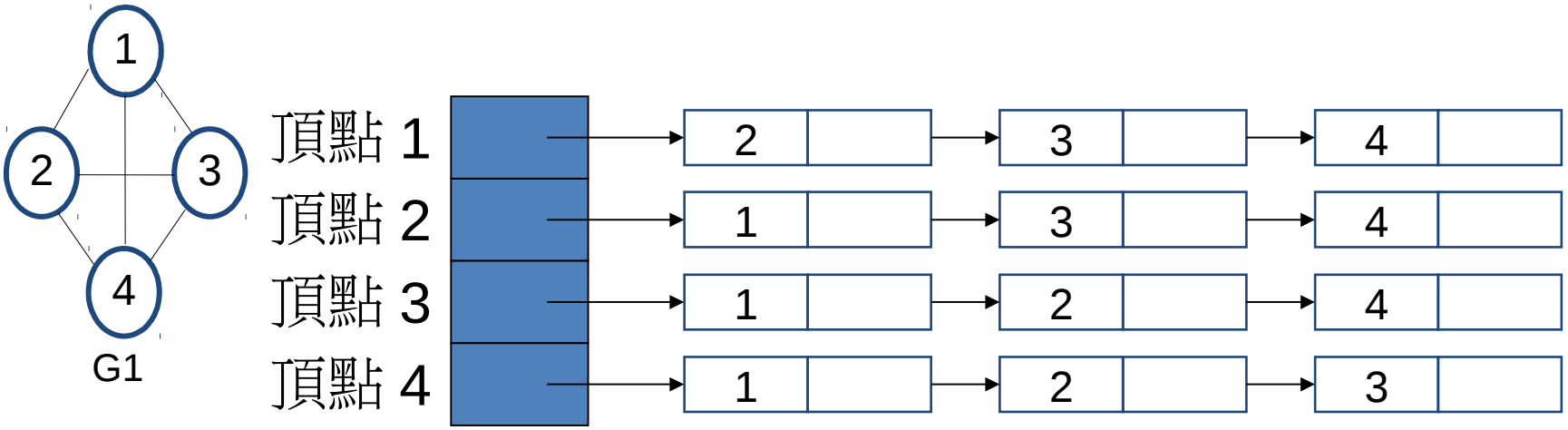
- Complete Graph



G1

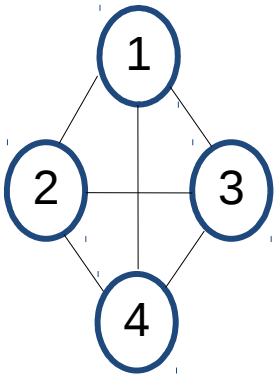
Graph

- Representation
  - adjacent list



Graph

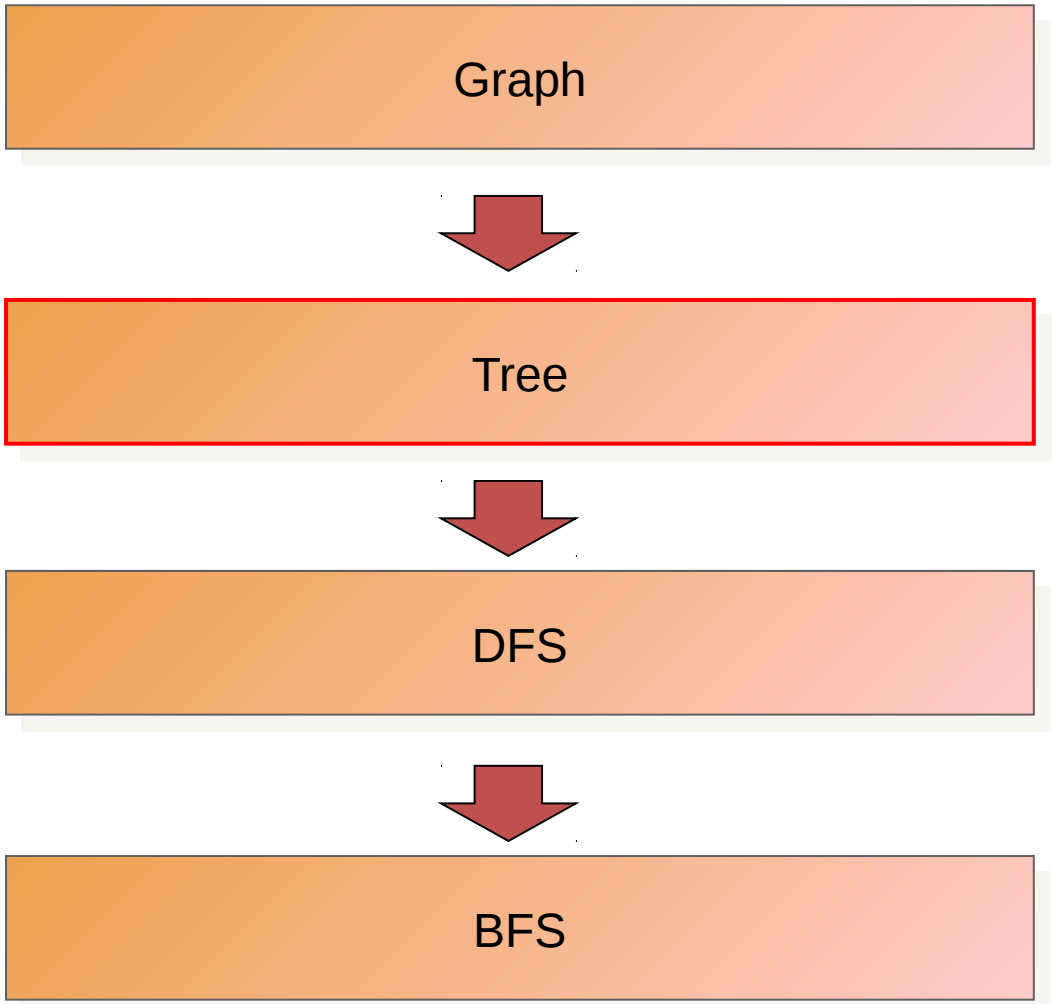
- Representation
  - adjacent matrix

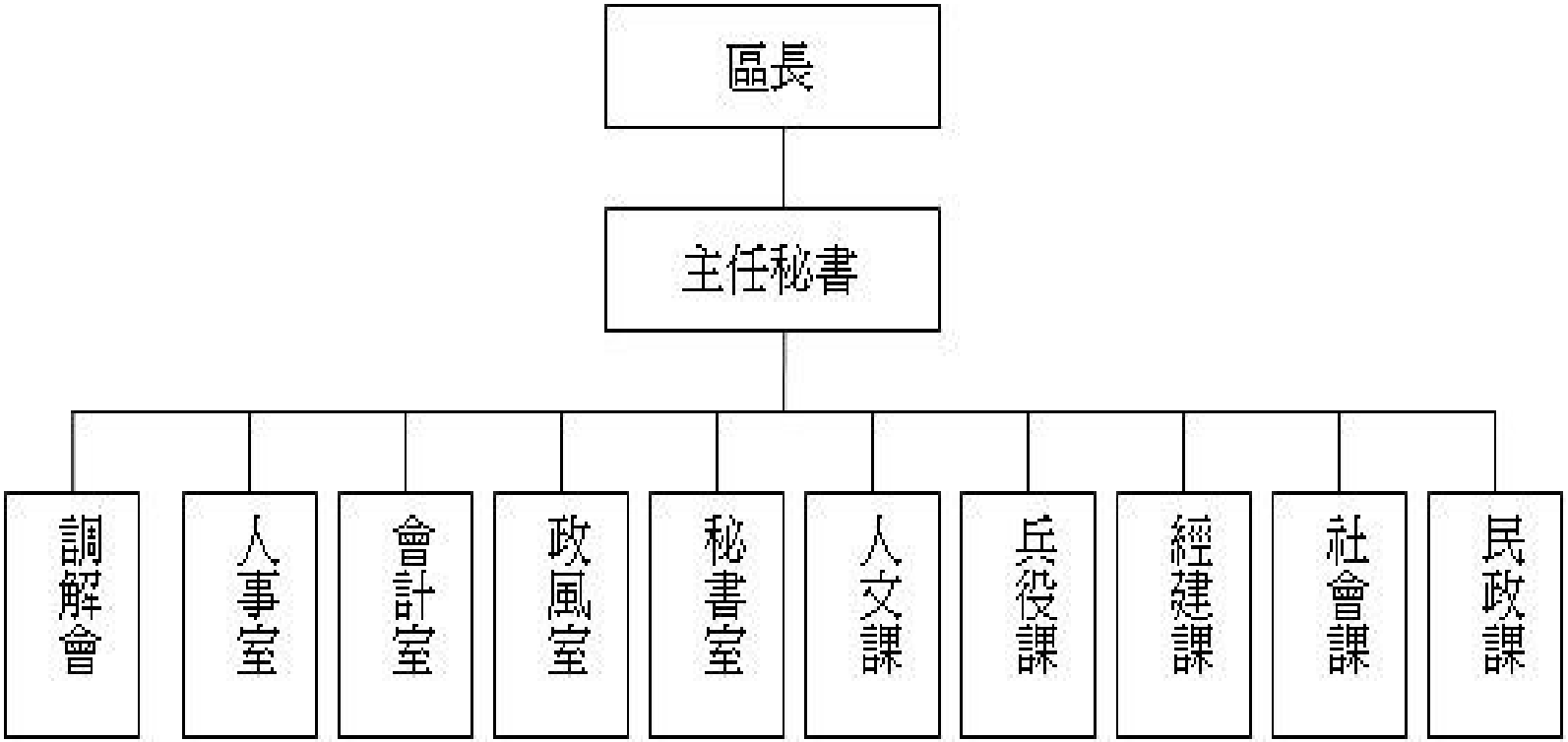


G1

	1	2	3	4
1	0	1	1	1
2	1	0	1	1
3	1	1	0	1
4	1	1	1	0

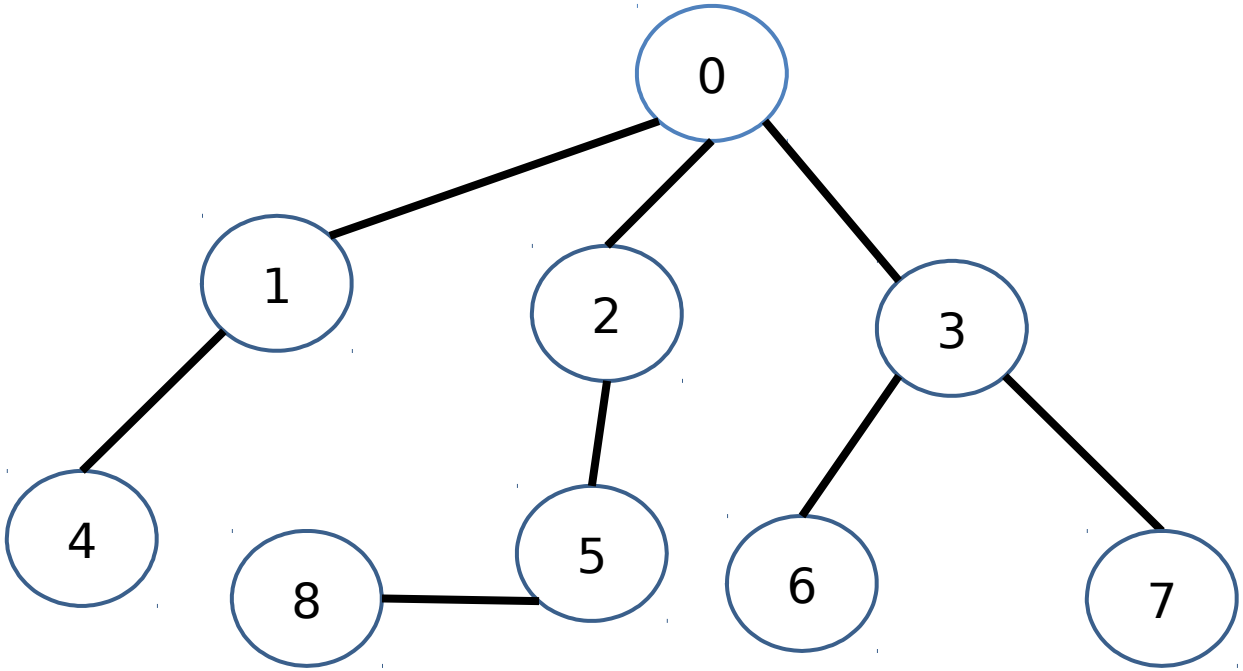
Outline




區長
主任秘書
調解會
人事室
會計室
政風室
秘書室
人文課
兵役課
經建課
社會課
民政課

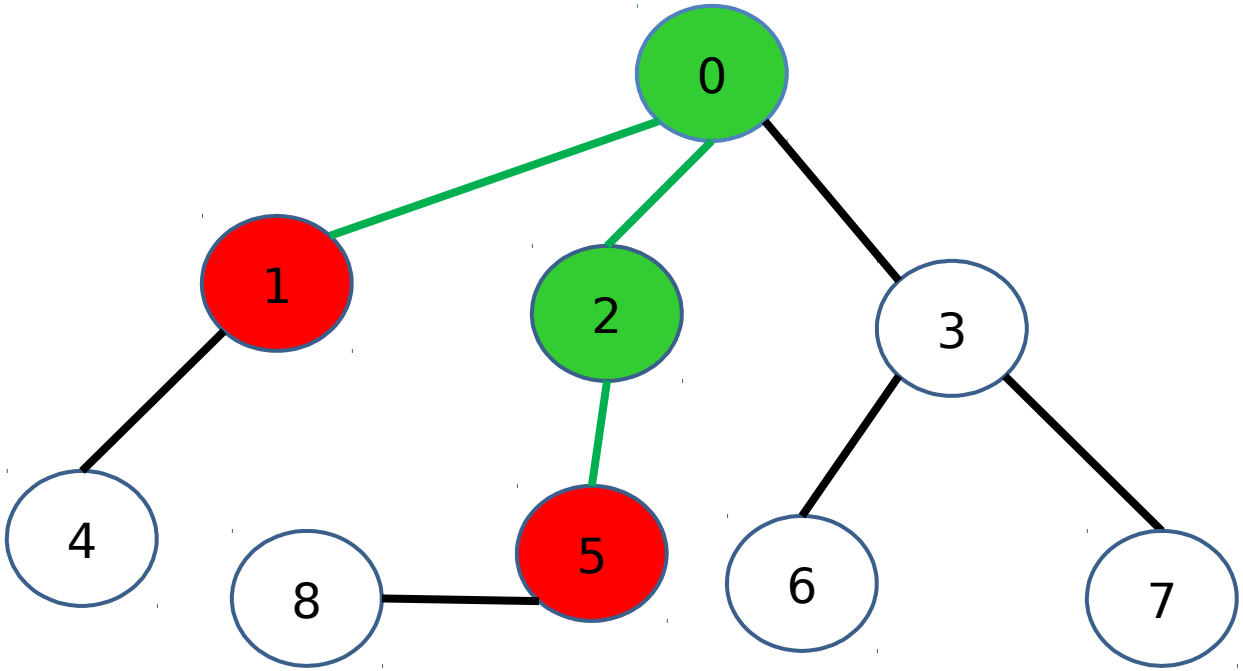
Tree

- A data structure consists of nodes. (at least two)



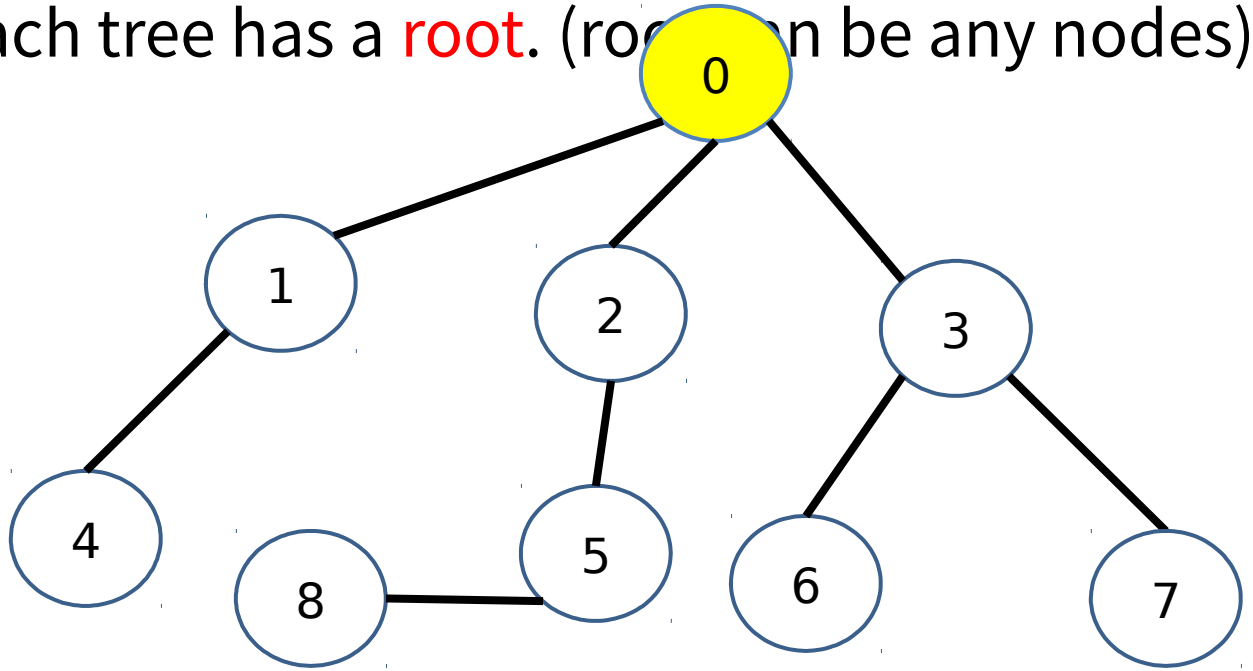
Tree

- A data structure consists of nodes. (at least two)
- Every pair of nodes must have one and only one path.(connected)



Tree

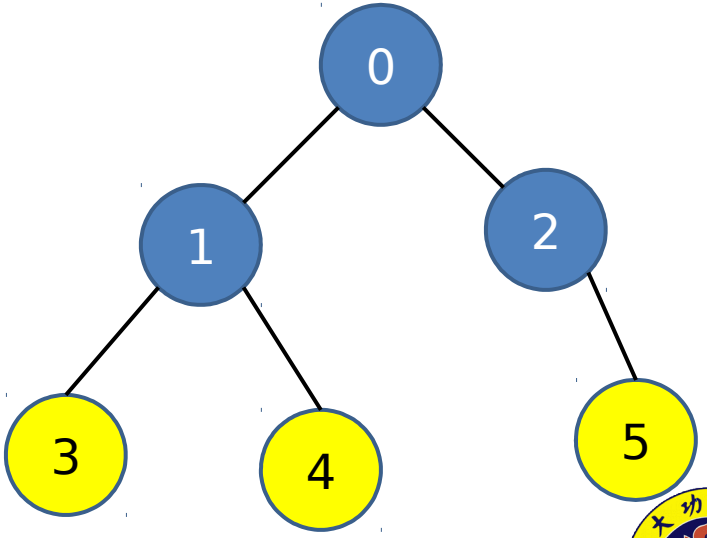
- A data structure consists of nodes. (at least two)
- Every pair of nodes must have one and only one path.(connected)
- Each tree has a root. (root can be any nodes)



Tree

Relations: (using node 0 as root)

-**Leaf :** nodes with no branch. (3,4,5)



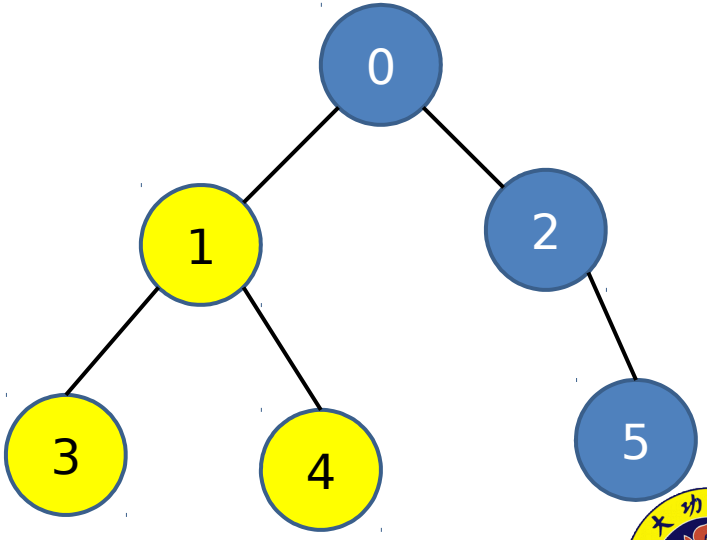
Tree

Relations: (using node 0 as root)

-**Leaf :** nodes with no branch.

-**Parent/Children :** has succesor nodes –Parent (1)

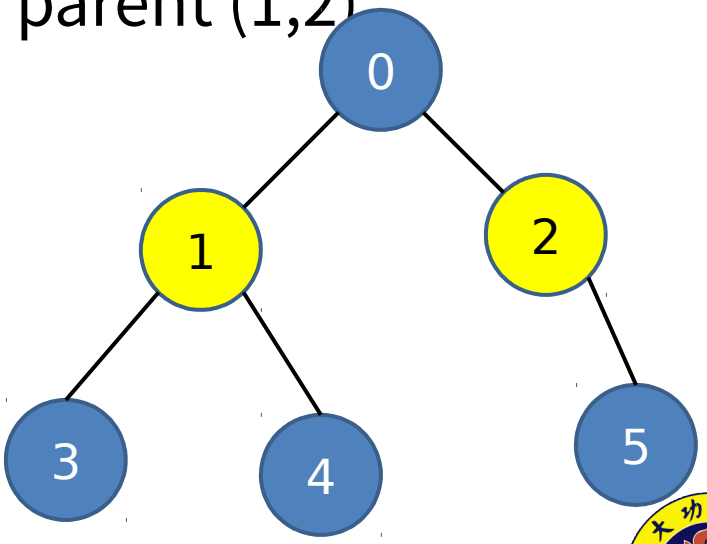
has predecessor node-children (3,4)



Tree

Relations: (using node 0 as root)

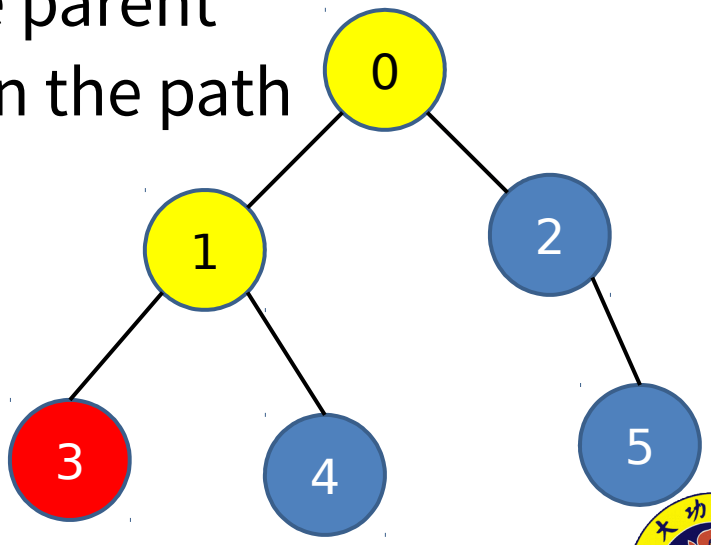
- **Leaf :** nodes with no branch.
- **Parent/Children :** has succesor nodes -Parent has predecessor node-children
- **Sibling:** children with same parent (1,2)



Tree

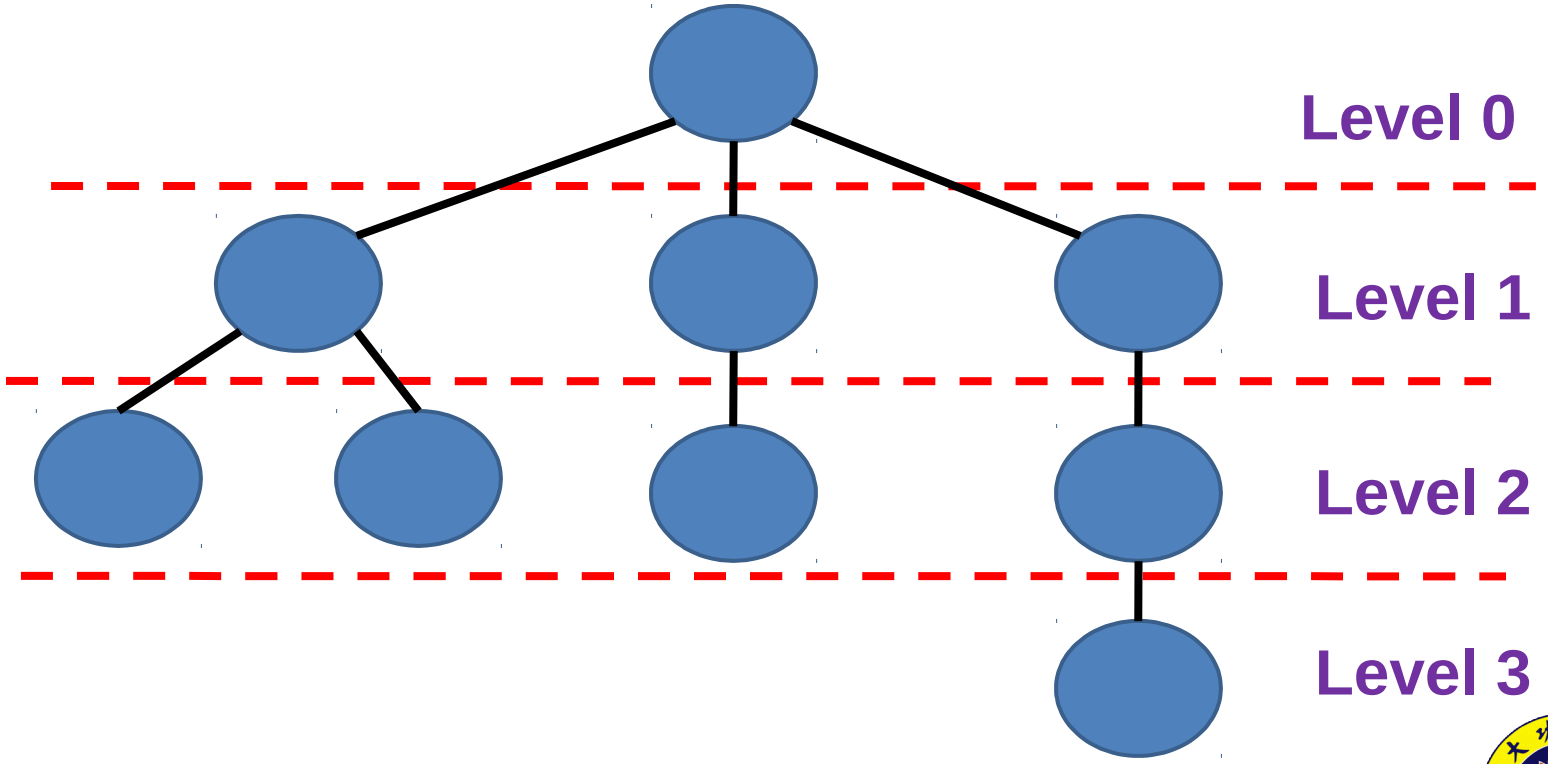
Relations: (using node 0 as root)

- **Leaf :** nodes with no branch.
- **Parent/Children :** has succesor nodes -Parent has predecessor node-children
- **Sibling:** children with same parent
- **Ancestor:** nodes that are on the path from root to itself. (3’ s ancestor: 0,1)



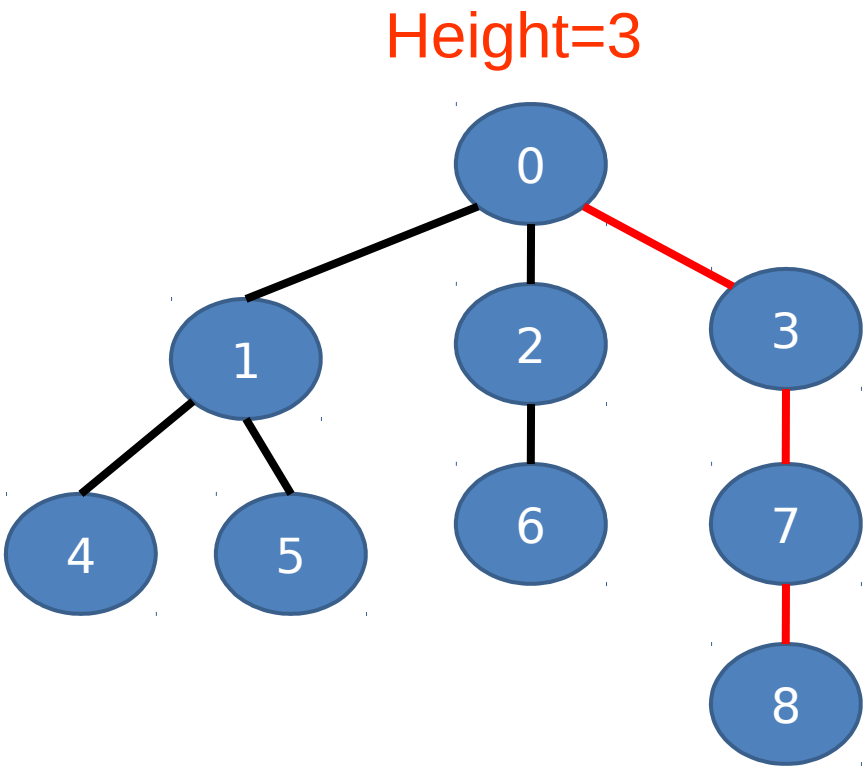
Tree

- **Tree Level:** according to the distance to the root.



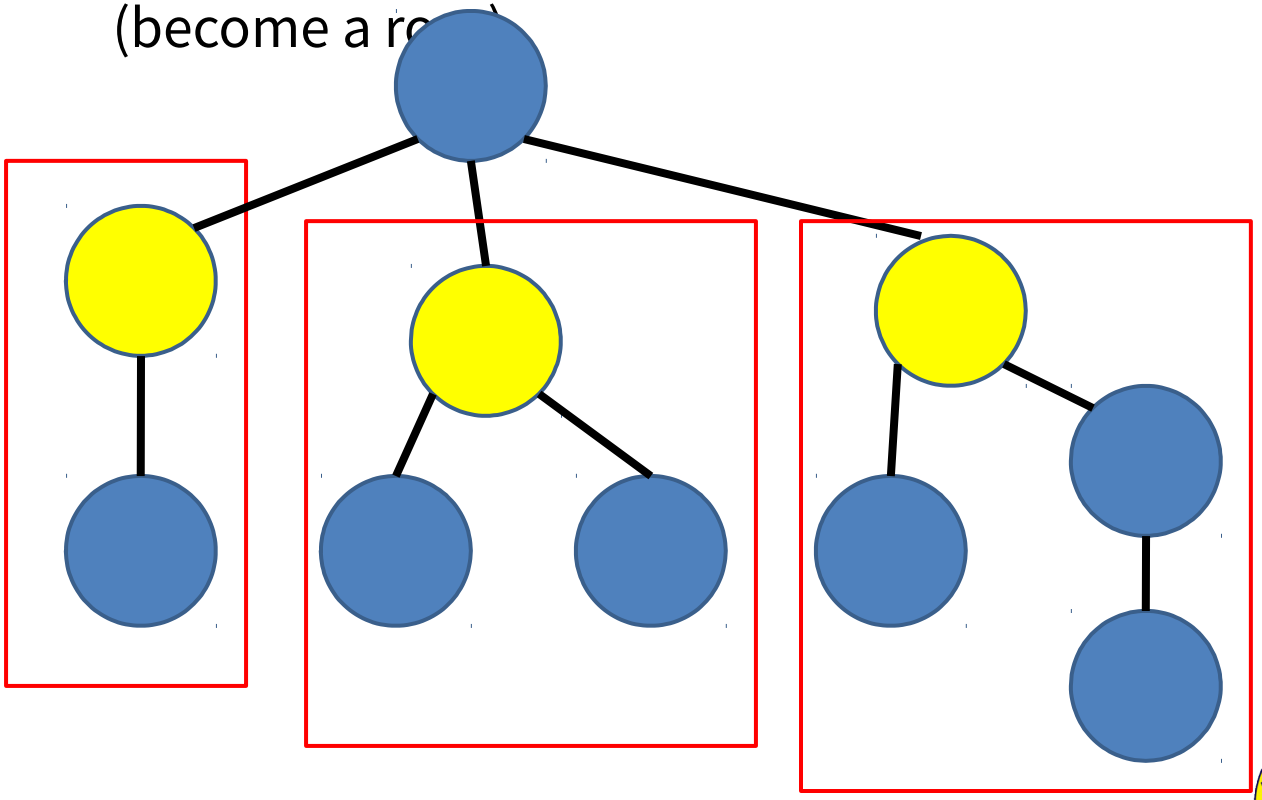
Tree

- **Tree Height(Depth):** the maximum distance from the root.



Tree

- **SubTree:** nodes that are not the root can form a subt ree



Tree

- **Feature:**

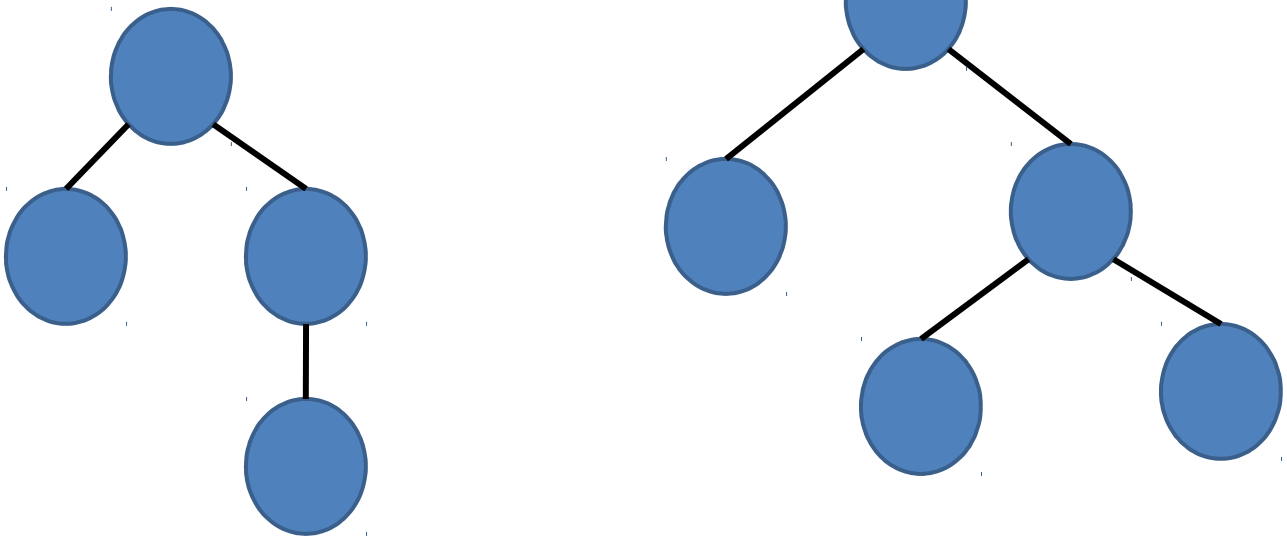
  -no cycle

  -every pair of nodes are connected

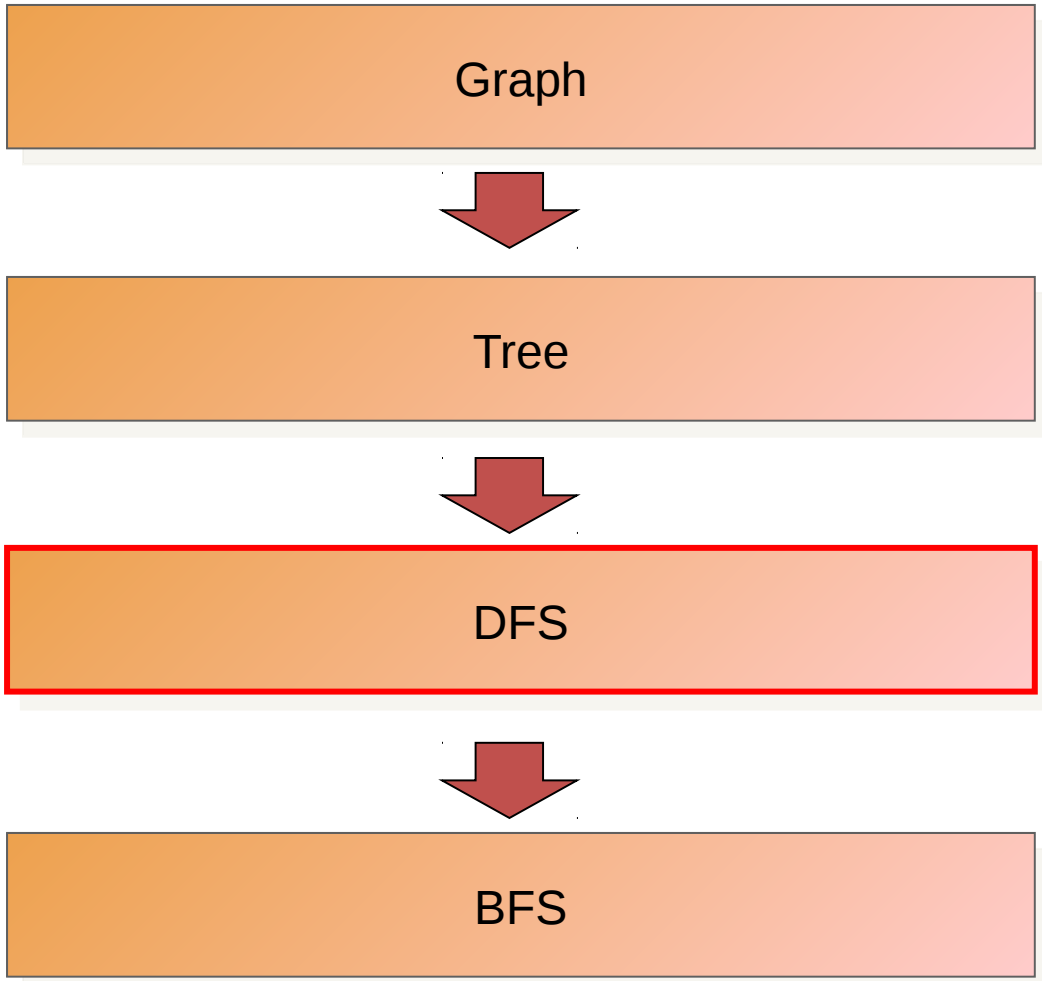
  -every pair of nodes has only one path

Tree

- **Forest:** n trees are disjoint (n ≧ 0)
  - ☒can be empty



Outline



DFS

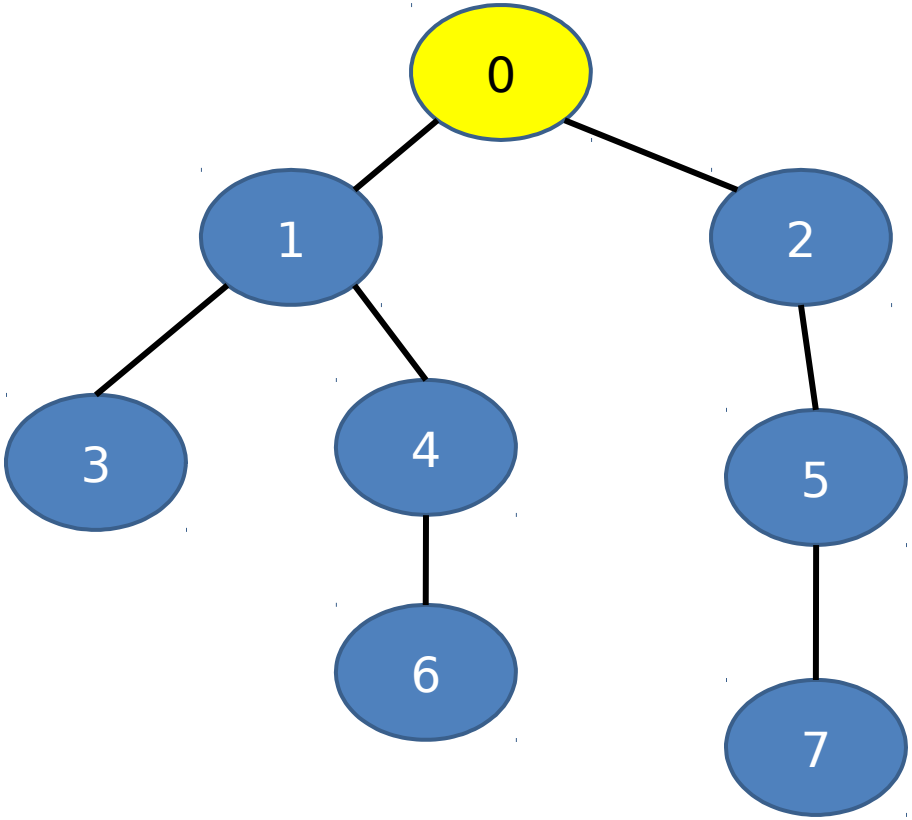
(D)epth-(F)irst-(S)earch

DFS

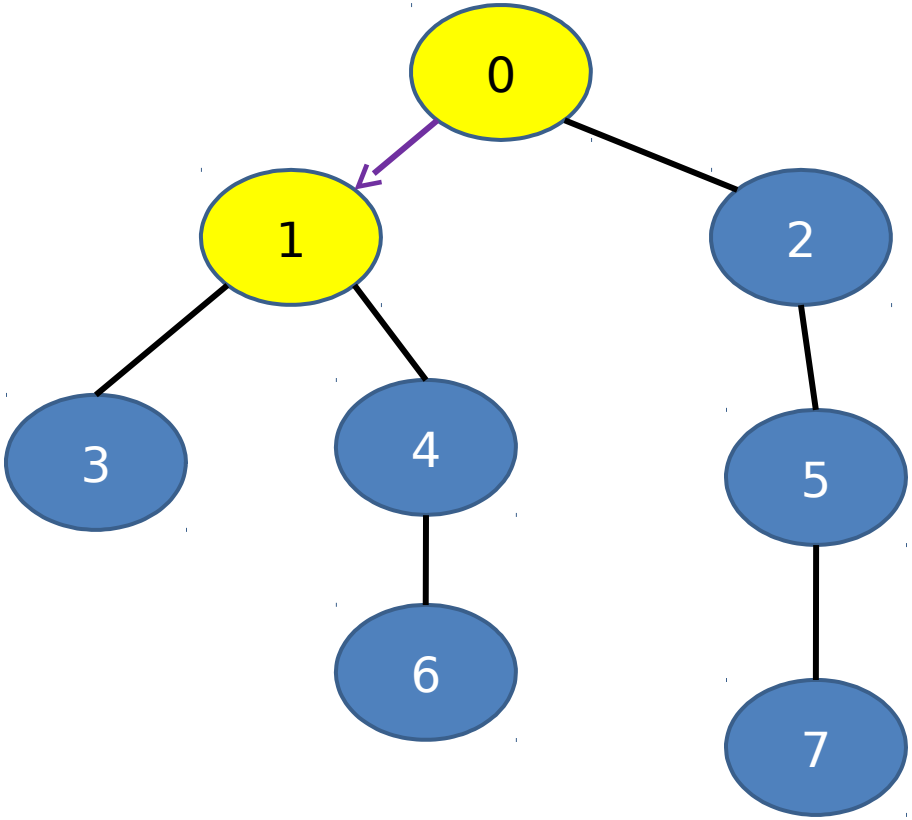
(D)epth (F)irst (S)earch

Stack

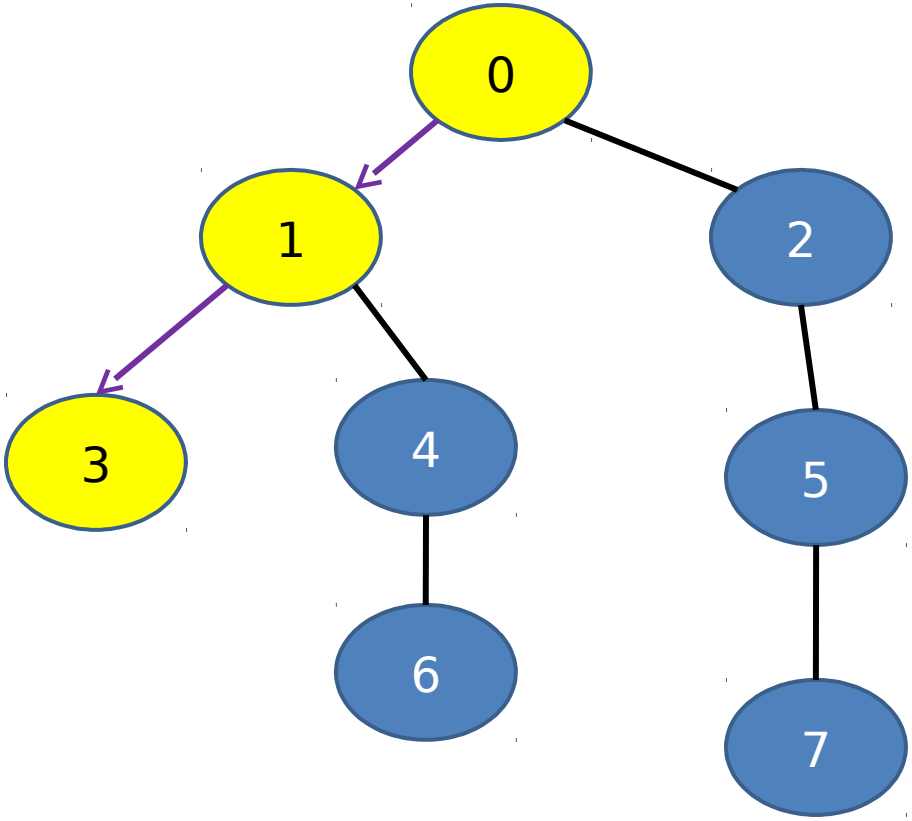
DFS



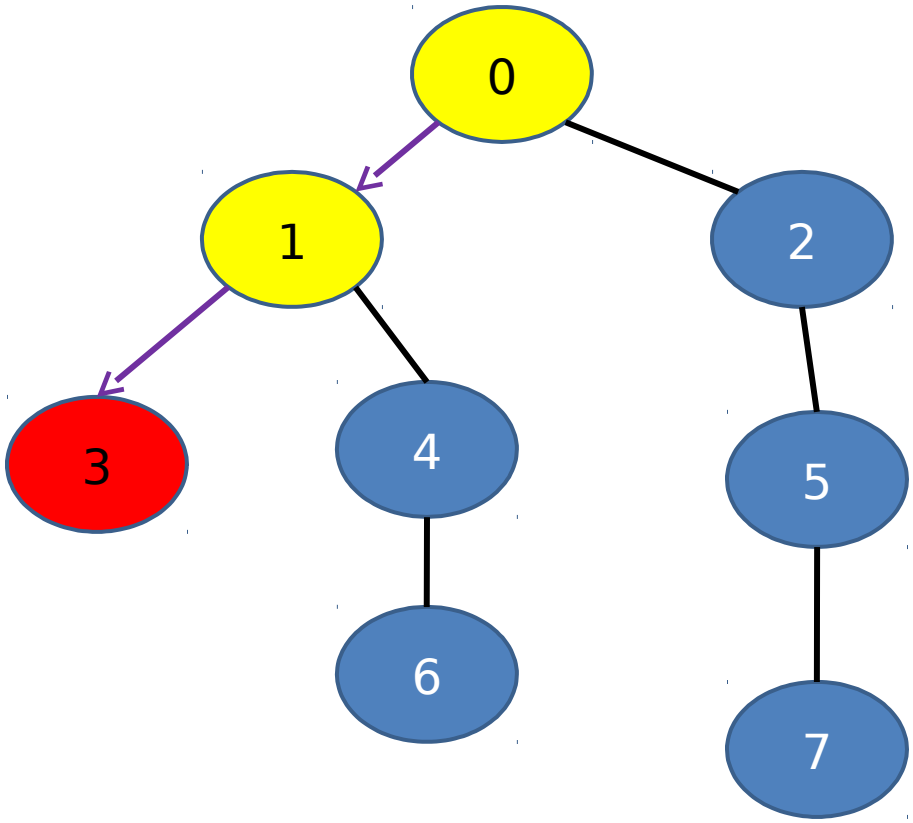
DFS



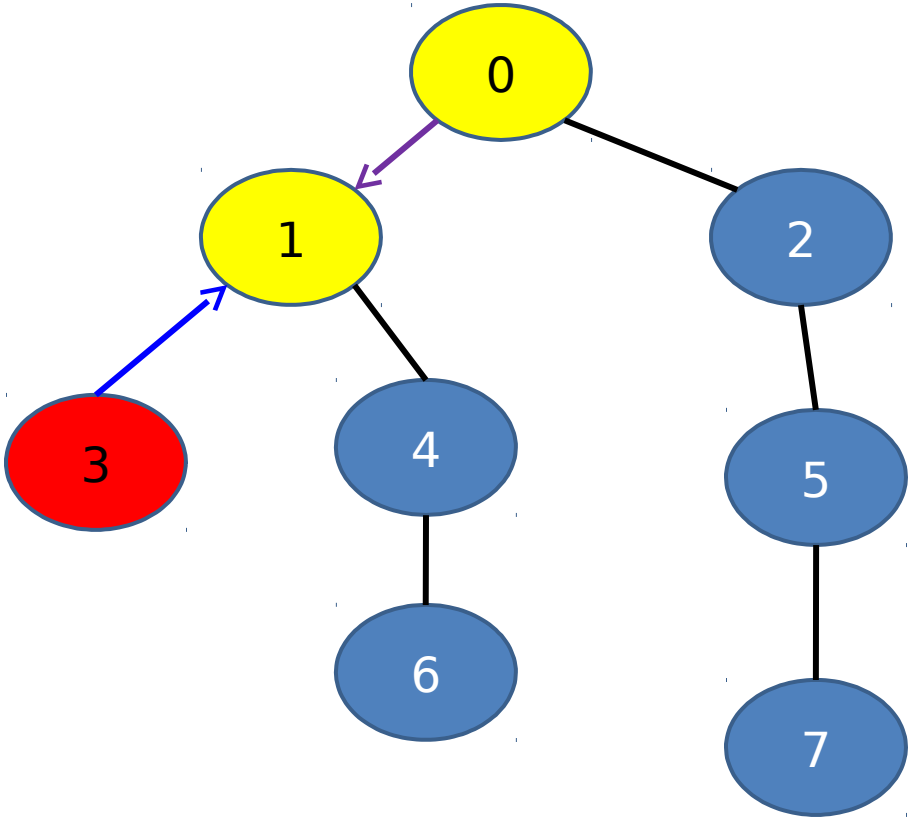
DFS



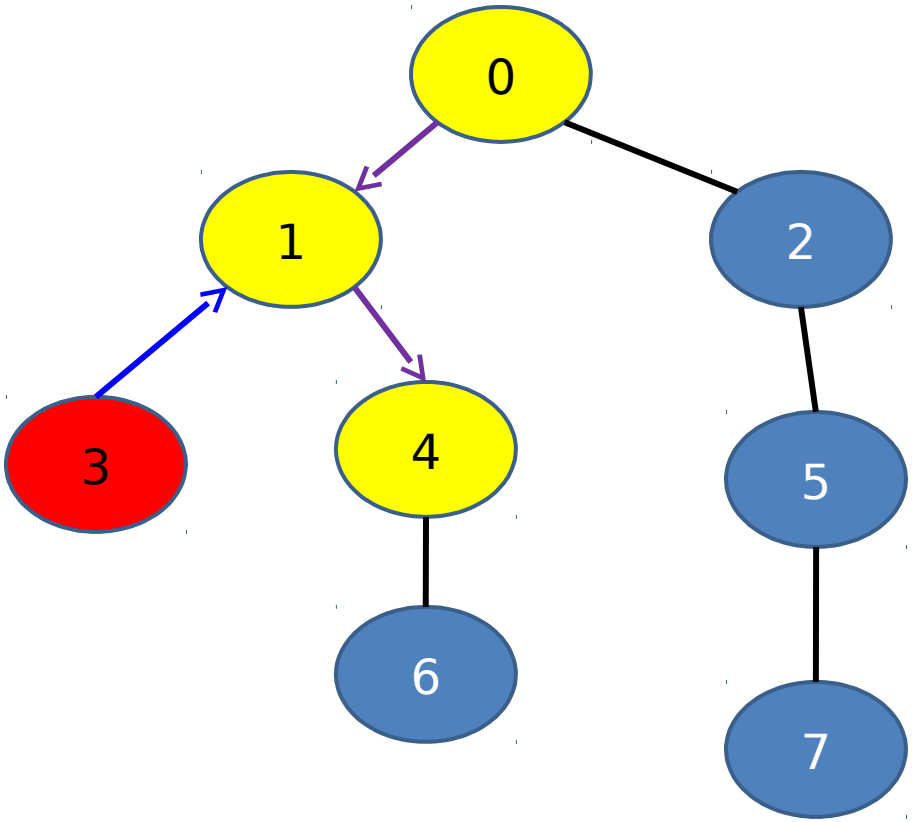
DFS



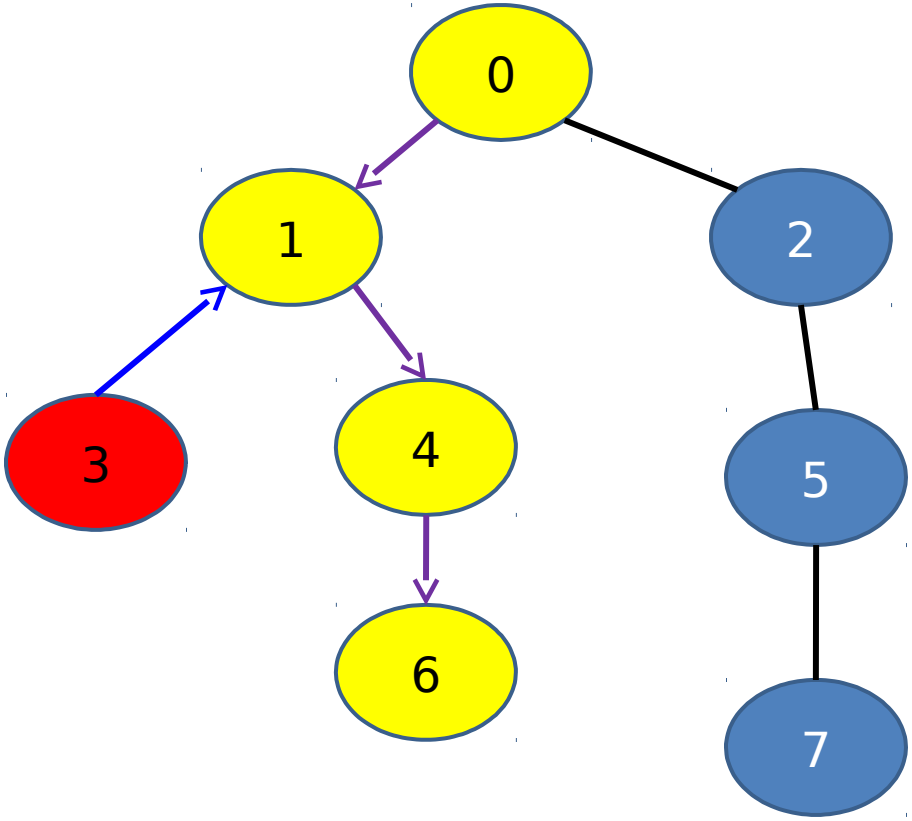
DFS



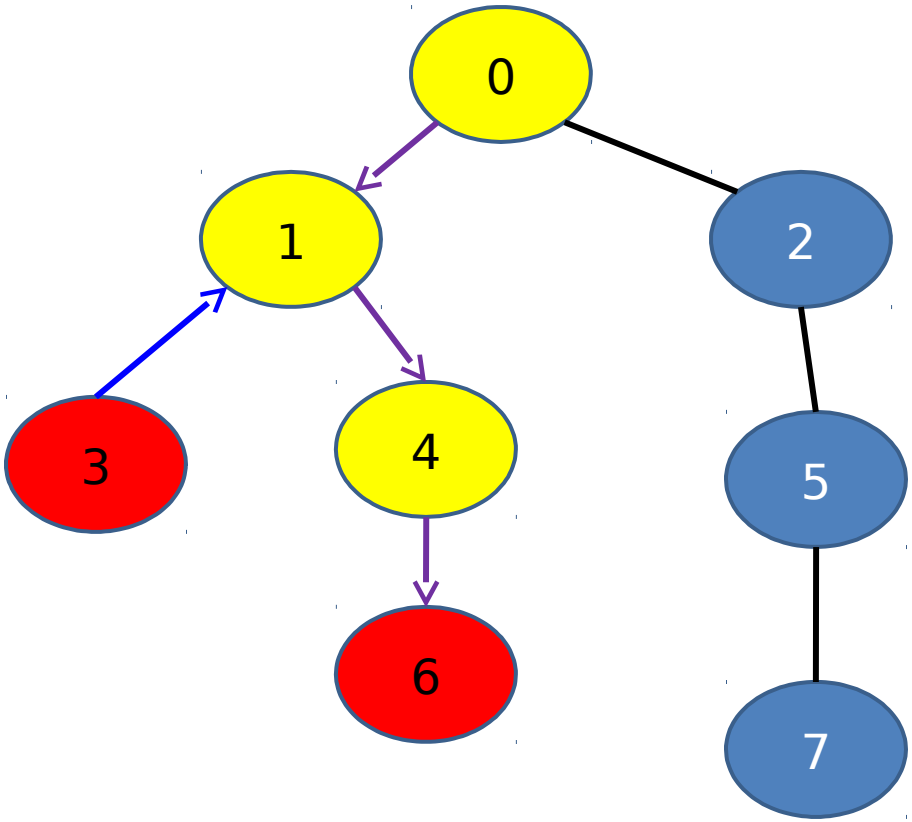
DFS



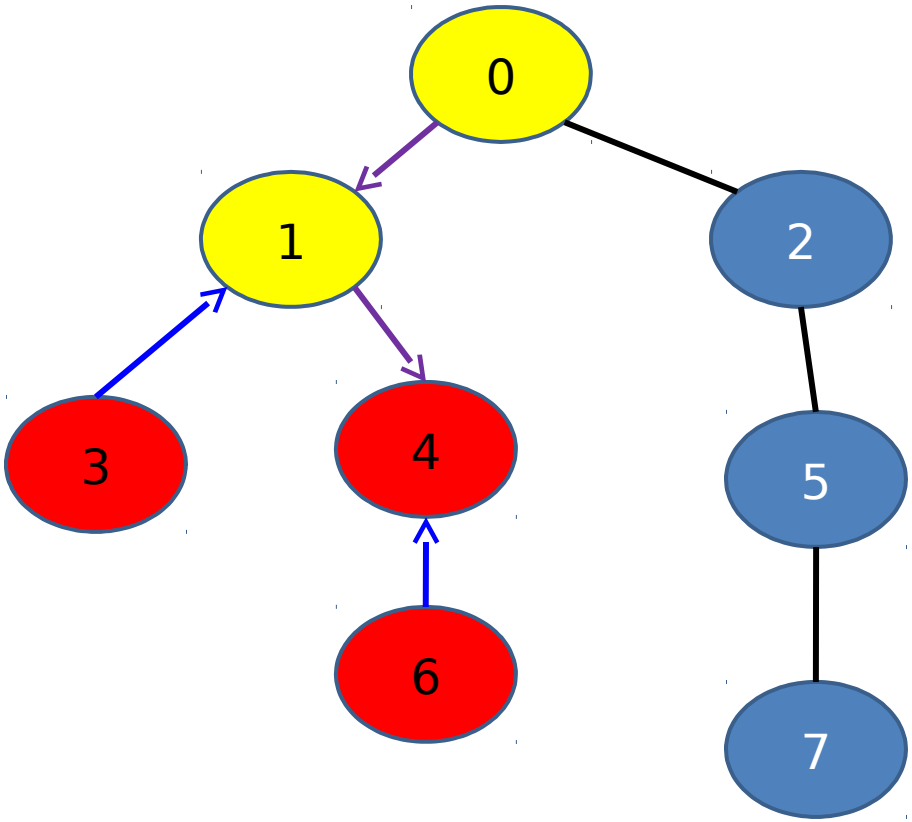
DFS



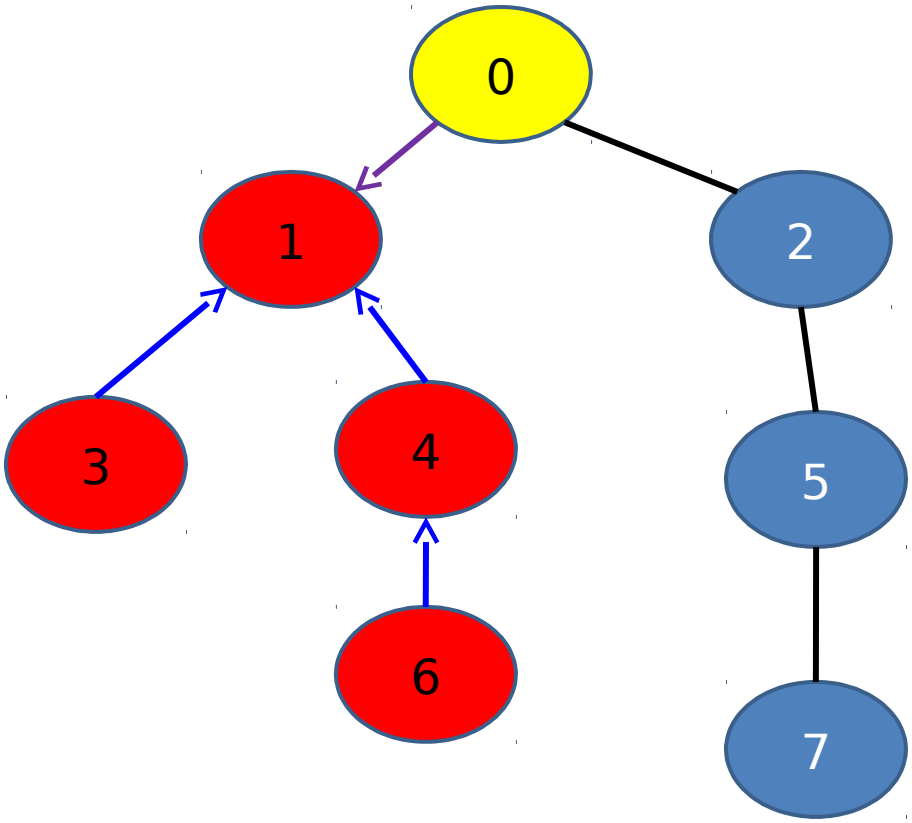
DFS



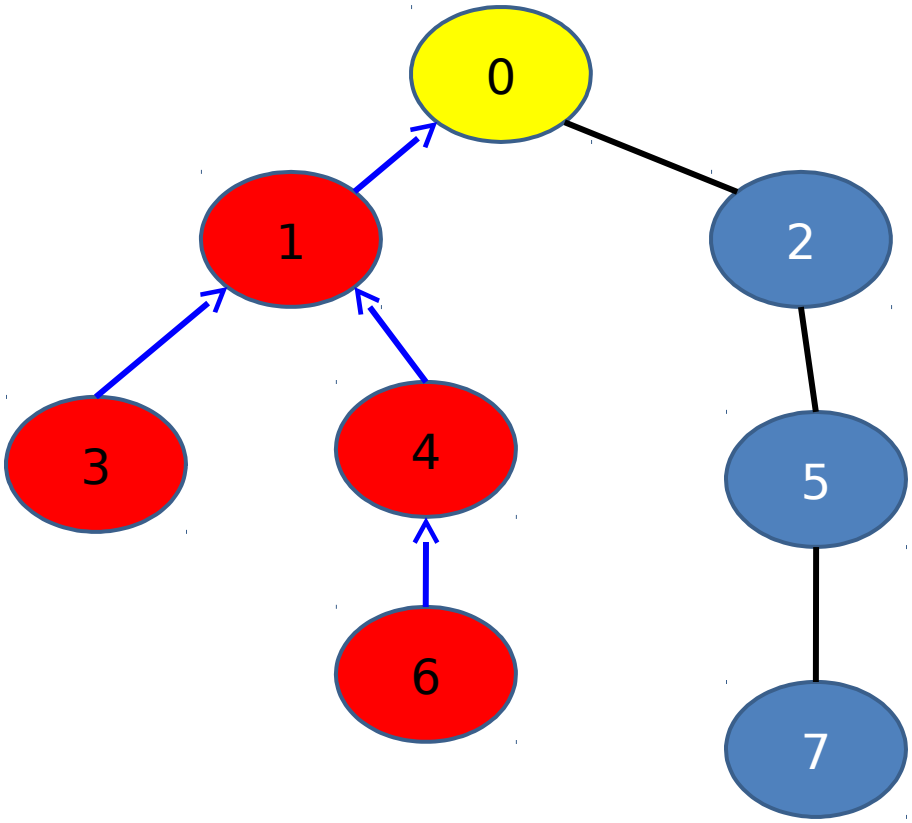
DFS



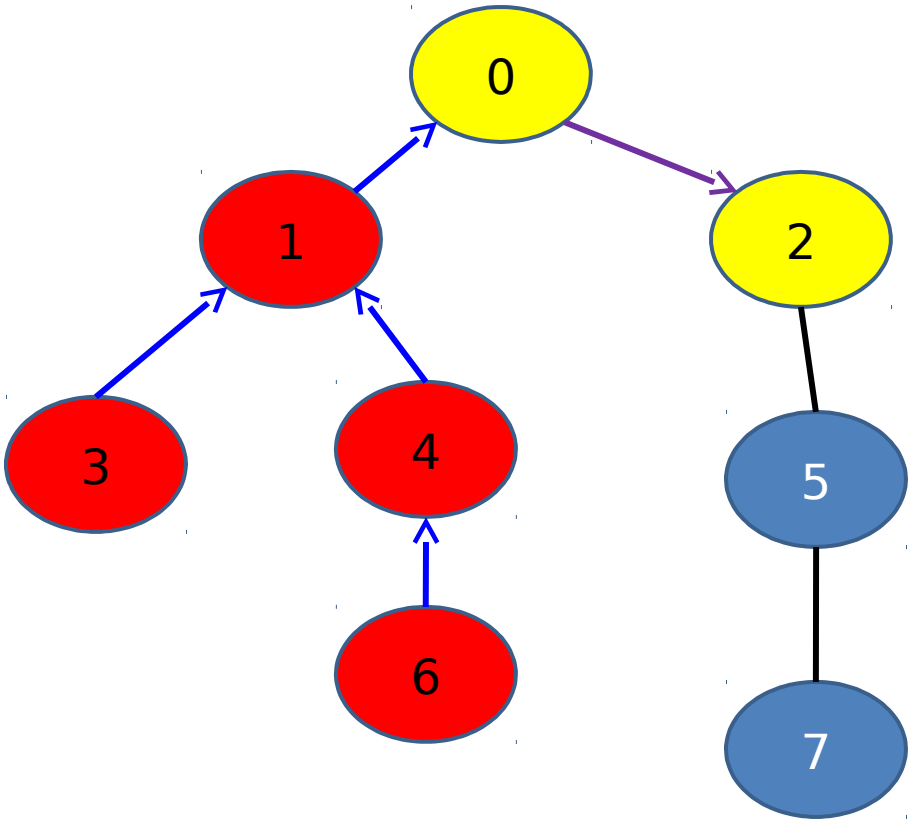
DFS



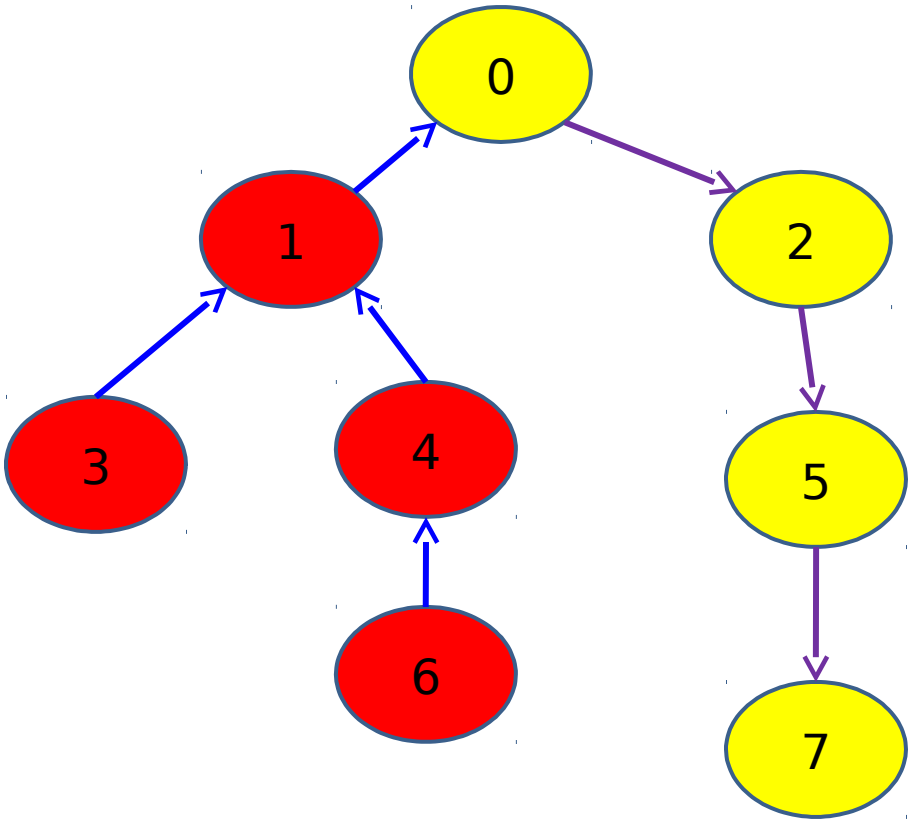
DFS



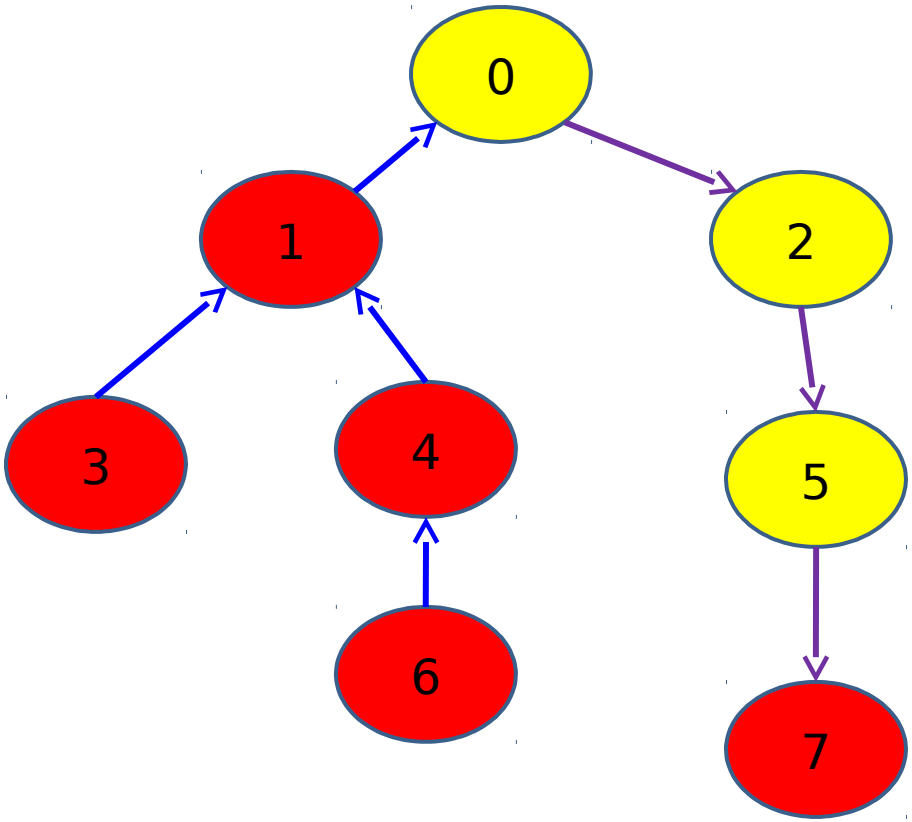
DFS



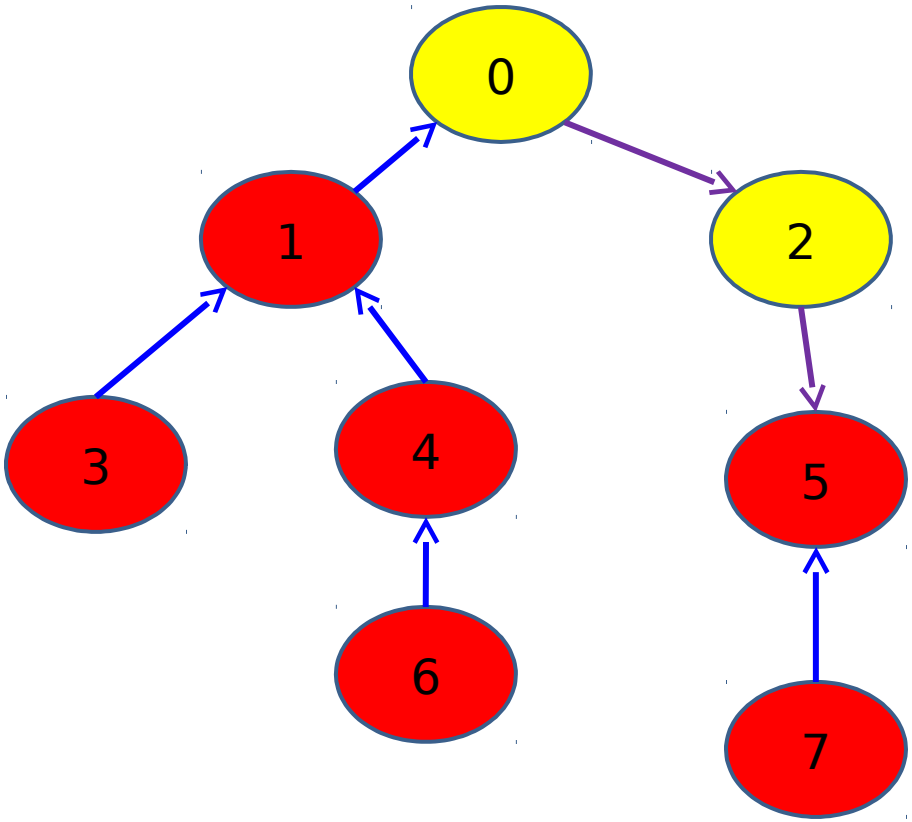
DFS



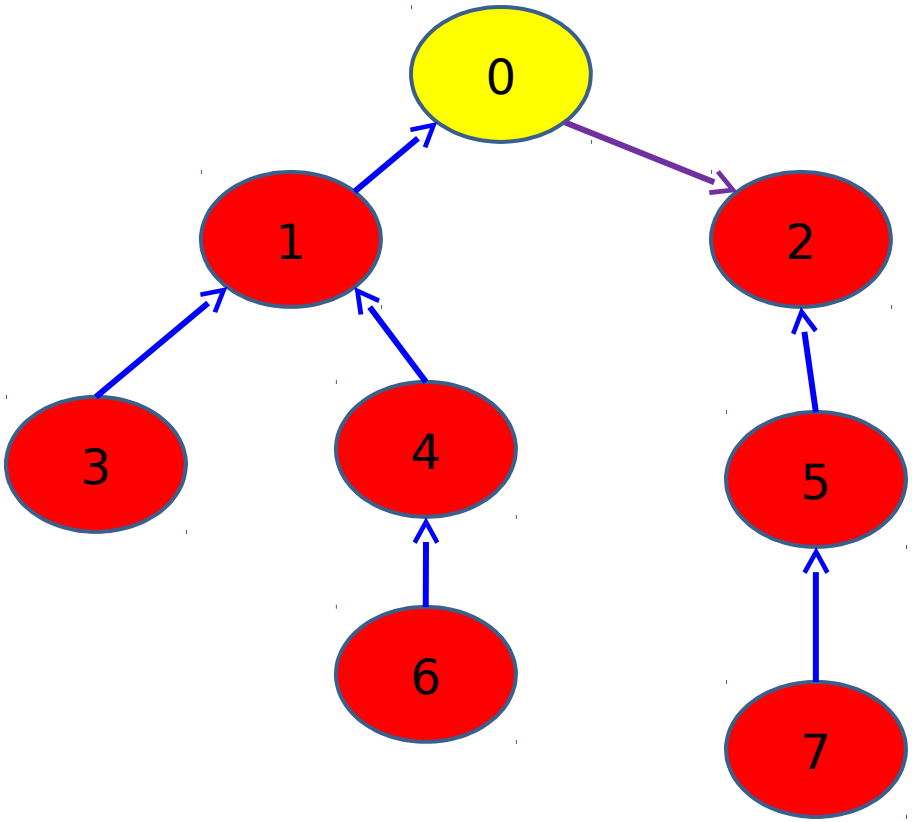
DFS



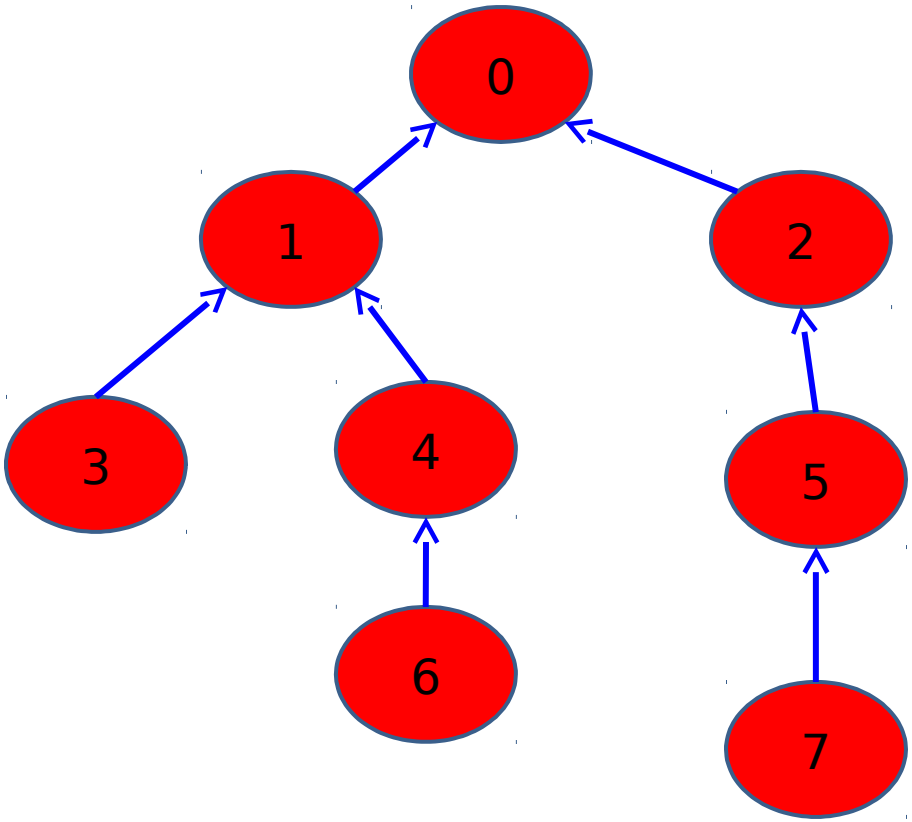
DFS



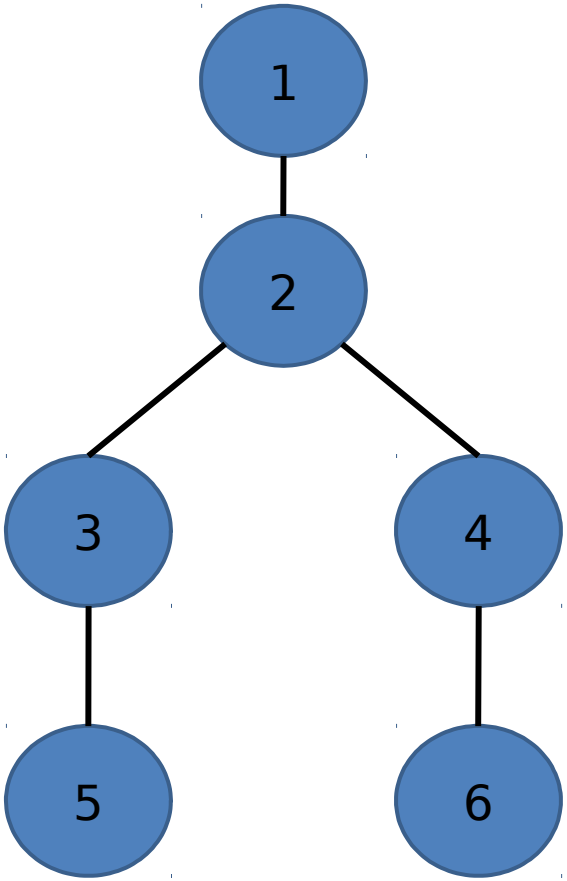
DFS



DFS

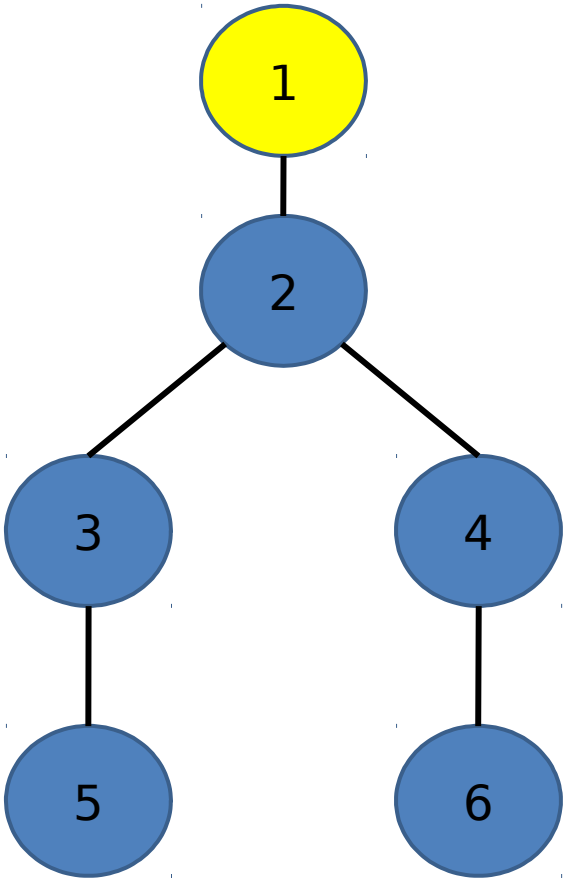


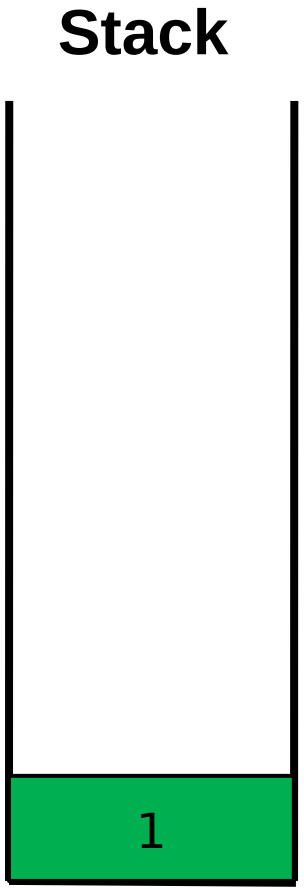
DFS



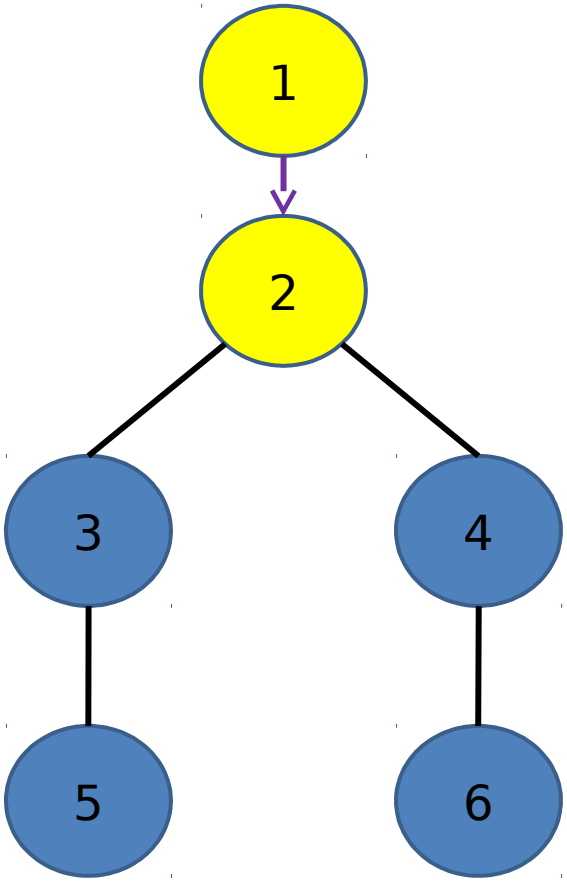
Stack

DFS

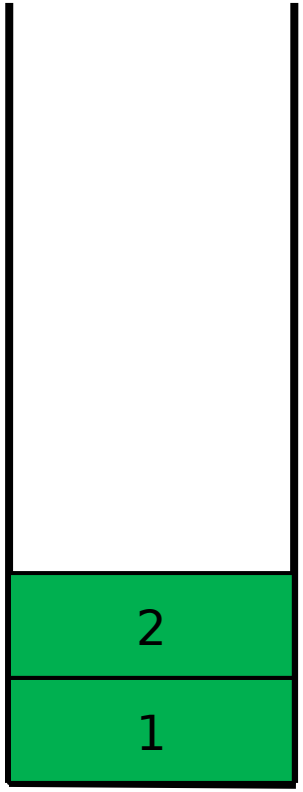




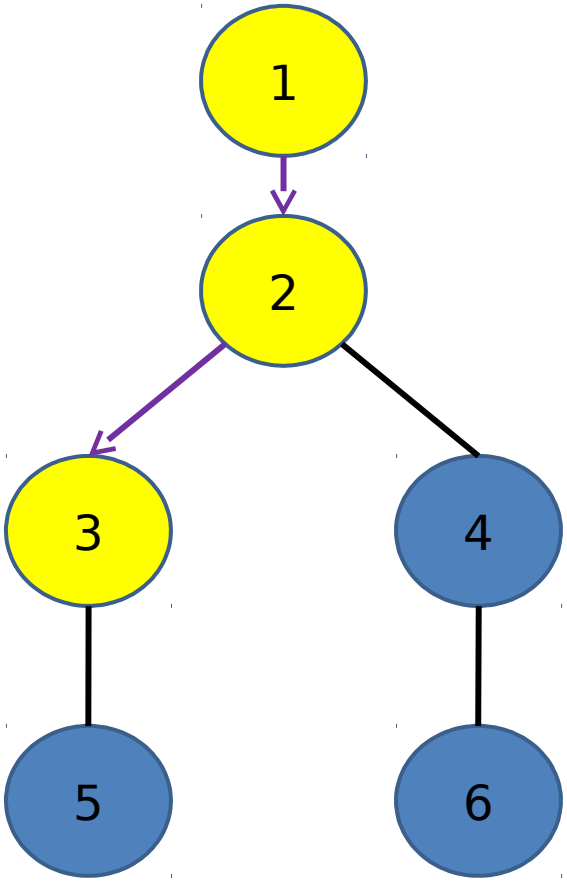
DFS



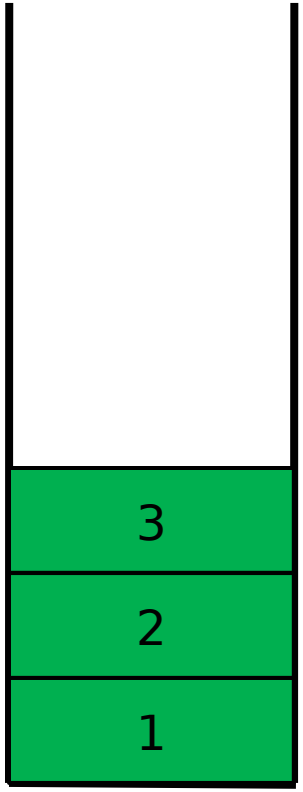
Stack



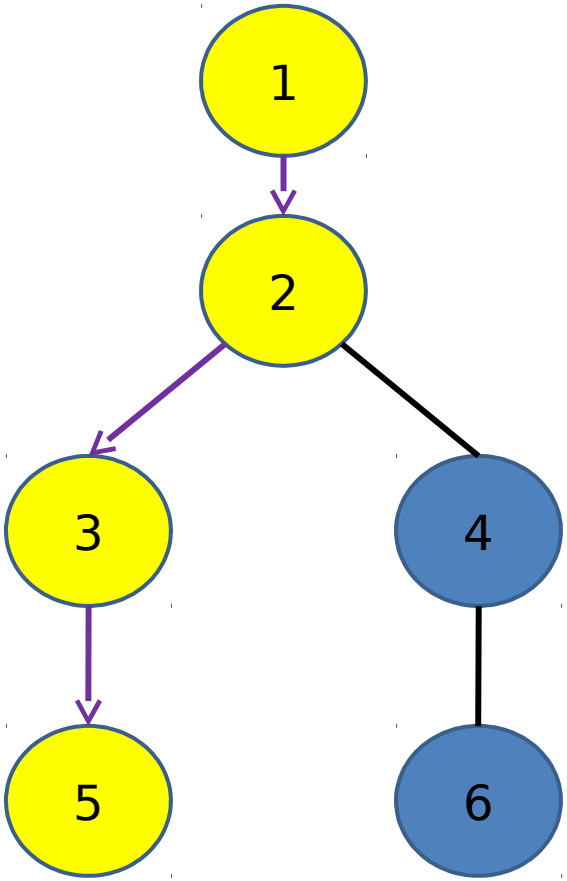
DFS



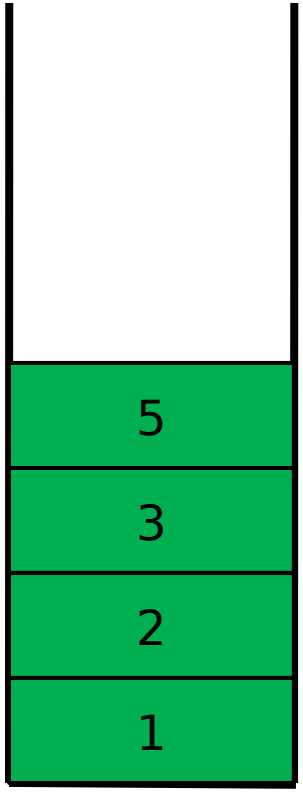
Stack



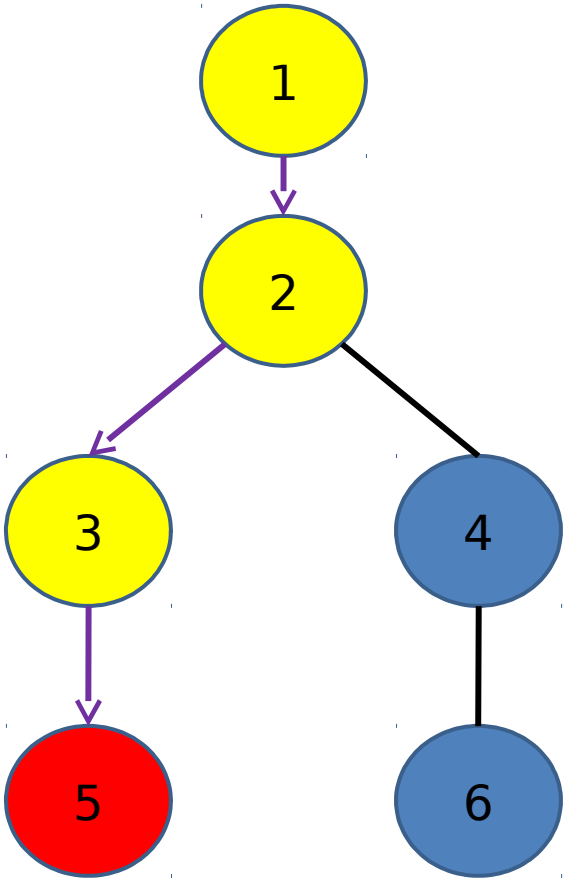
DFS



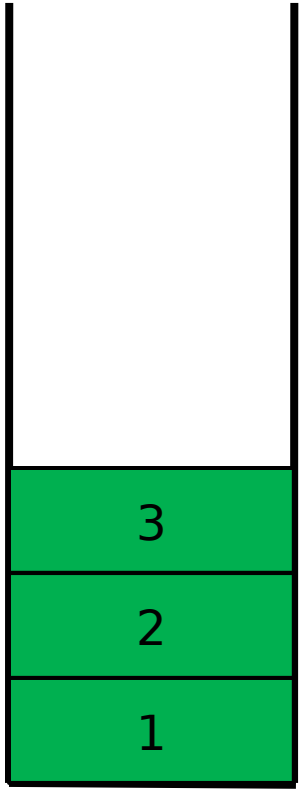
Stack



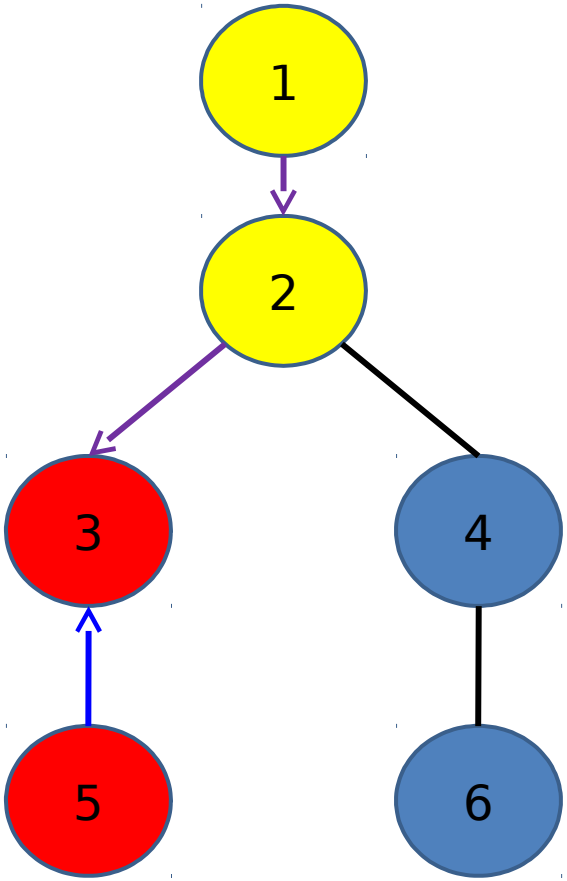
DFS



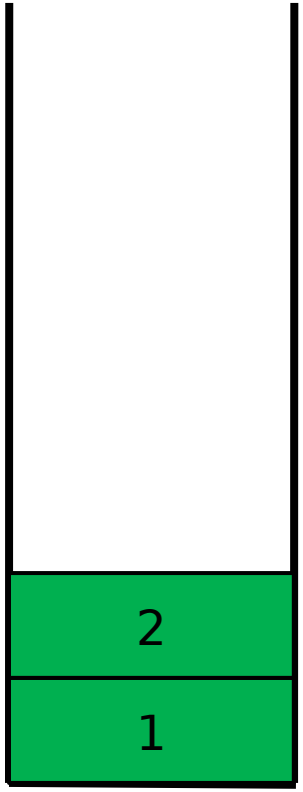
Stack



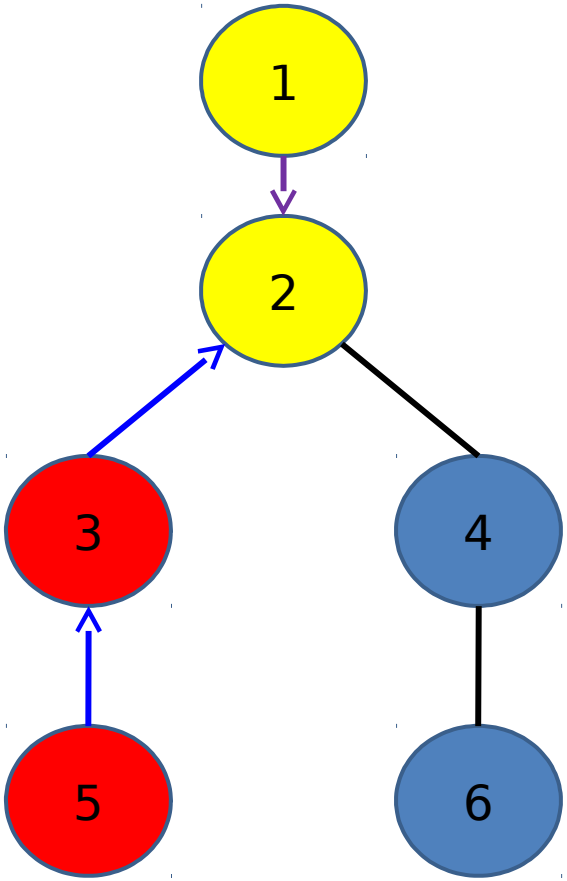
DFS

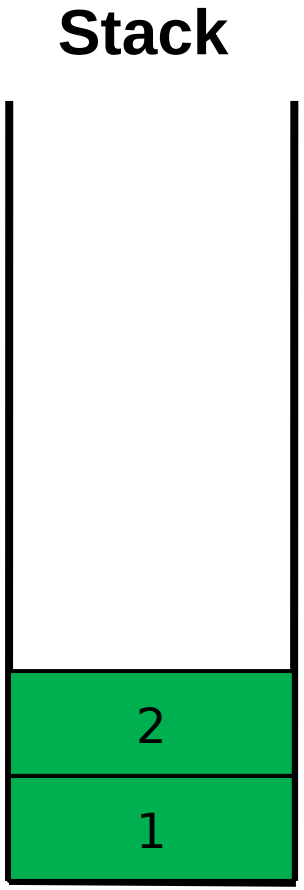


Stack

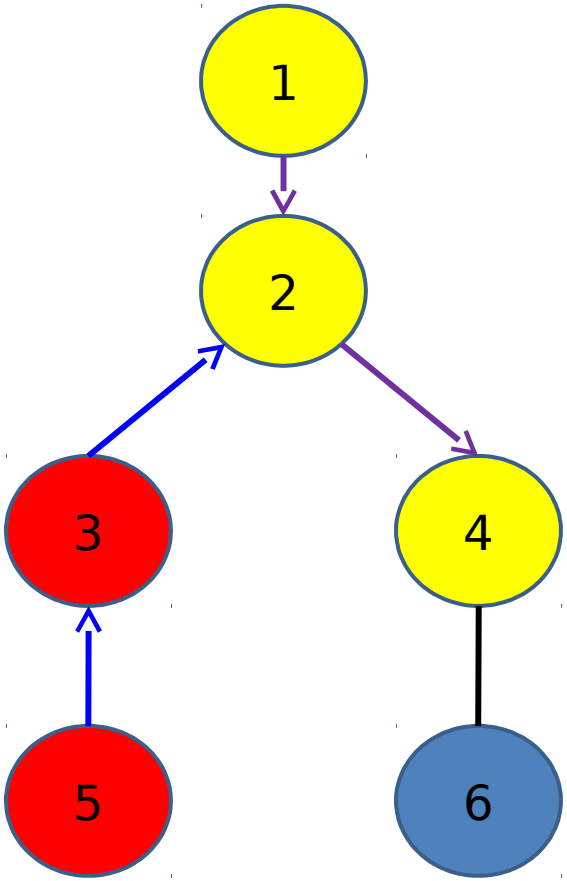


DFS

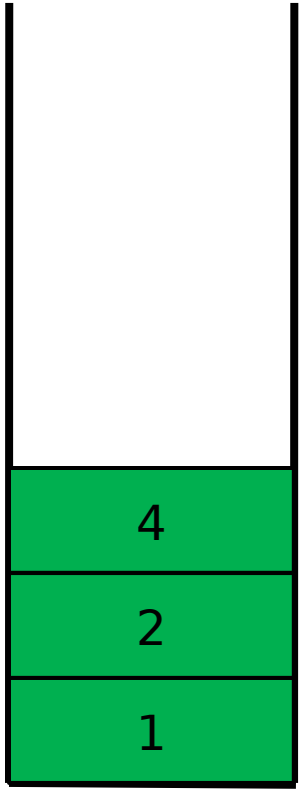




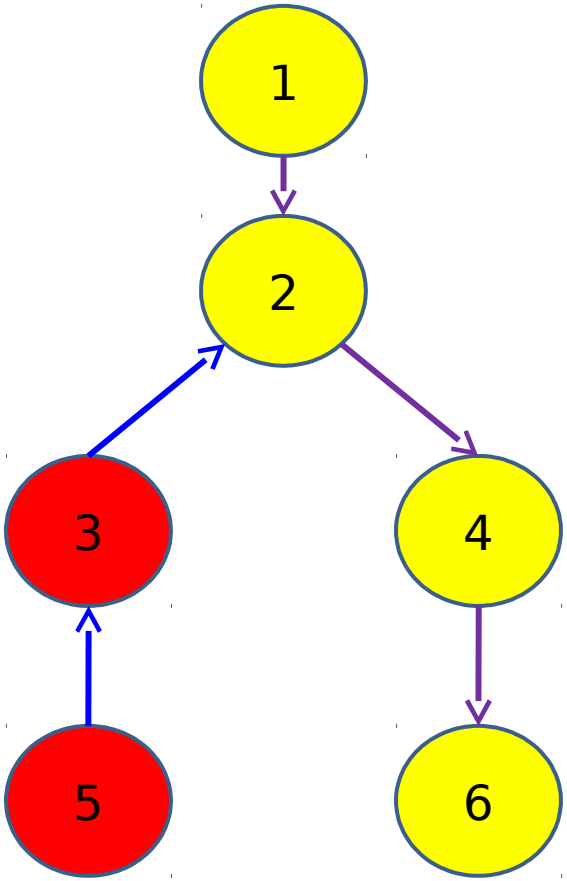
DFS



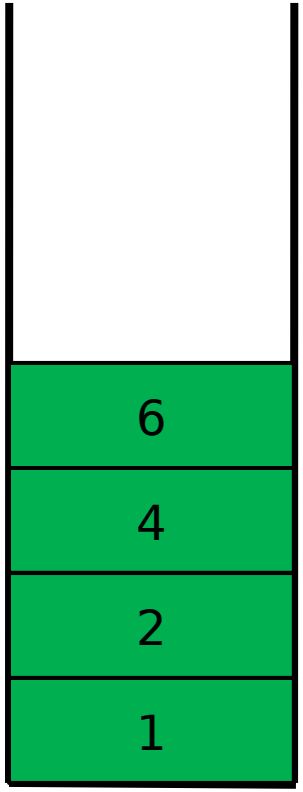
Stack



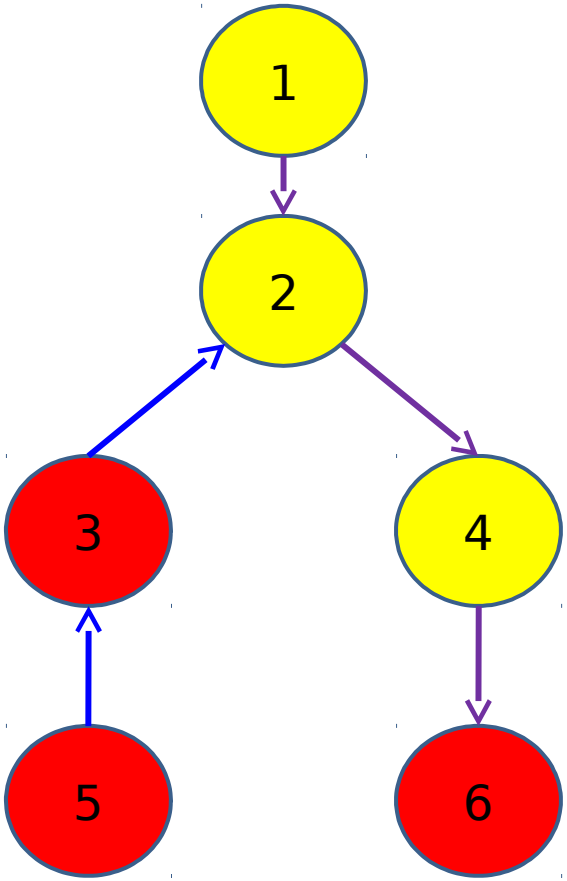
DFS



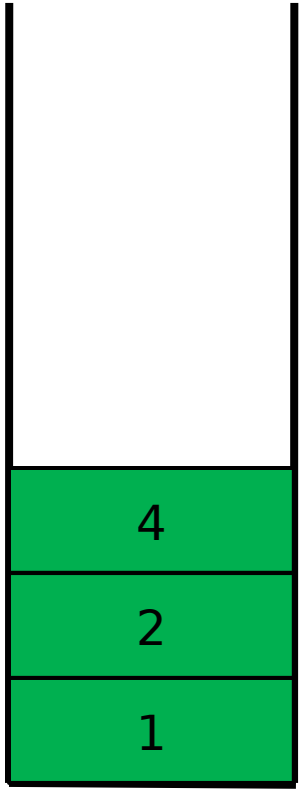
Stack



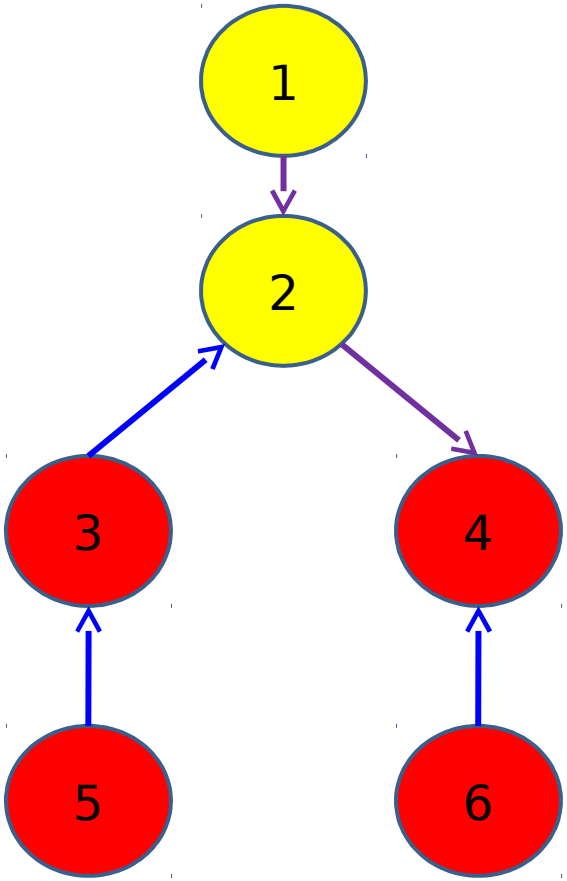
DFS

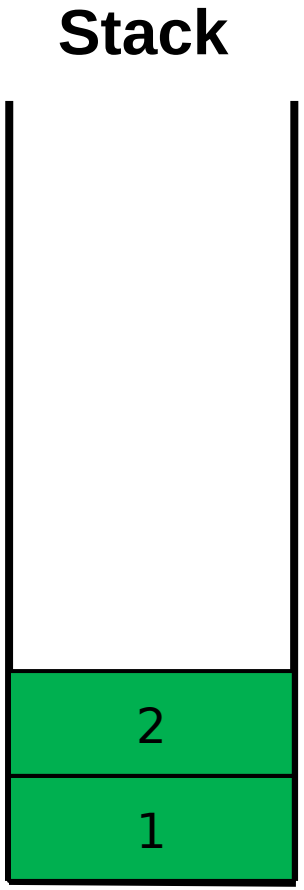


Stack

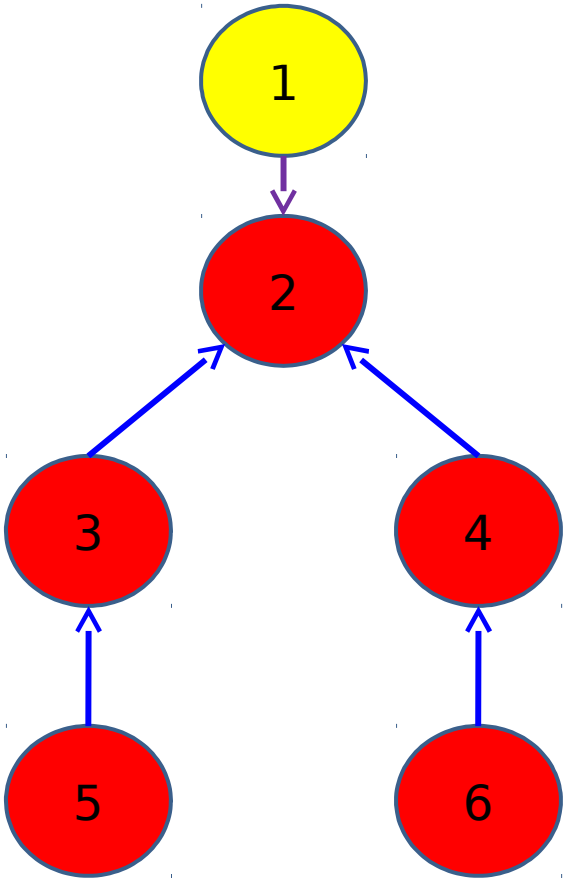


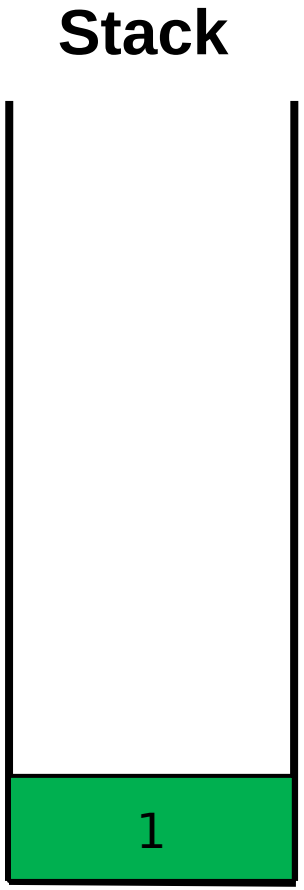
DFS



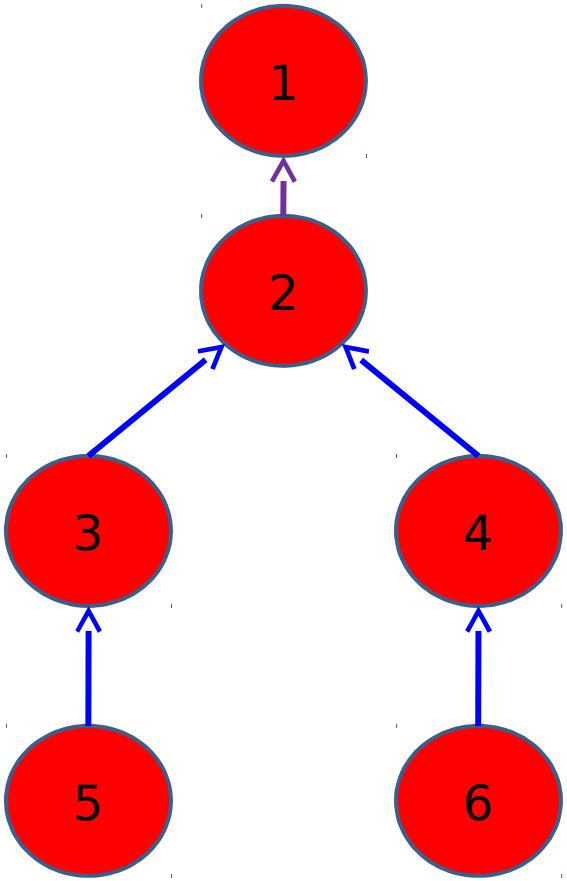


DFS

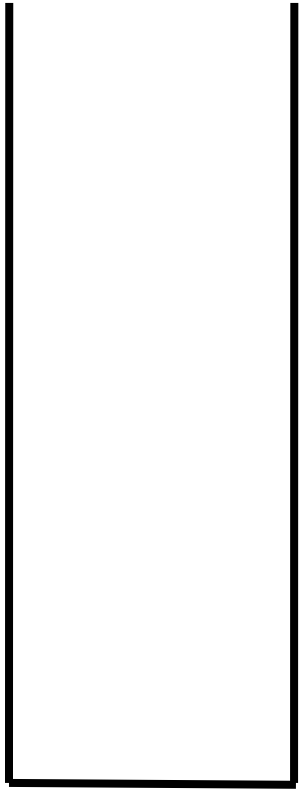




DFS



Stack



DFS

- Source Code(adjacency list)

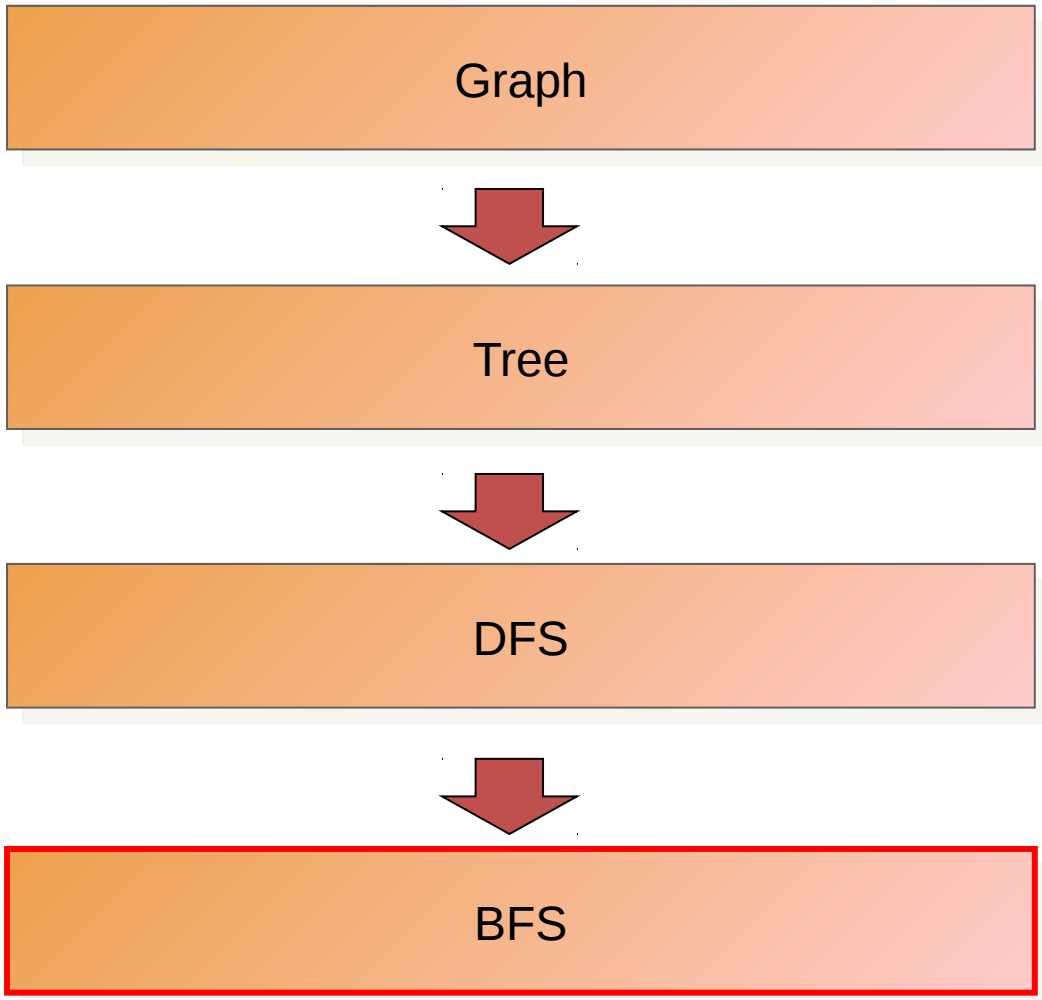
```
void DFS(int cur)
{
    vis[cur]=true;

    for(int i=0;i<adj[cur].size();i++)
    {
        int next=adj[cur][i];
        if(!vis[next])
            DFS(next);
    }
    return;
}
```

DFS

- Practice

[UVA-572] Oil Deposits

Outline



BFS

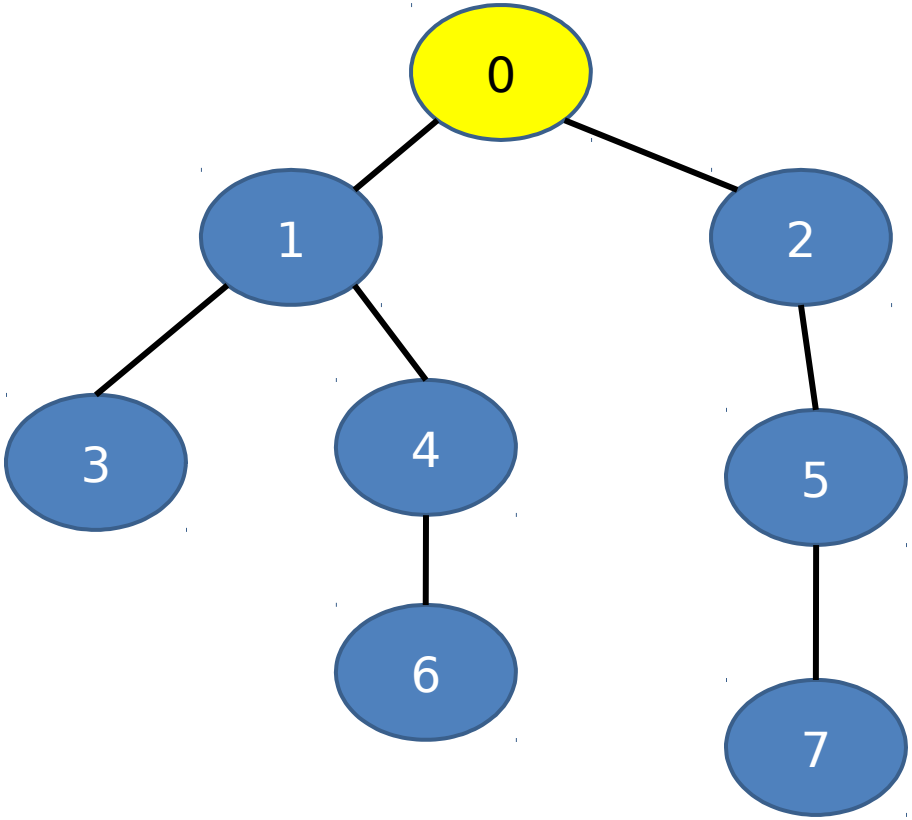
(B)readth-(F)irst-(S)earch

BFS

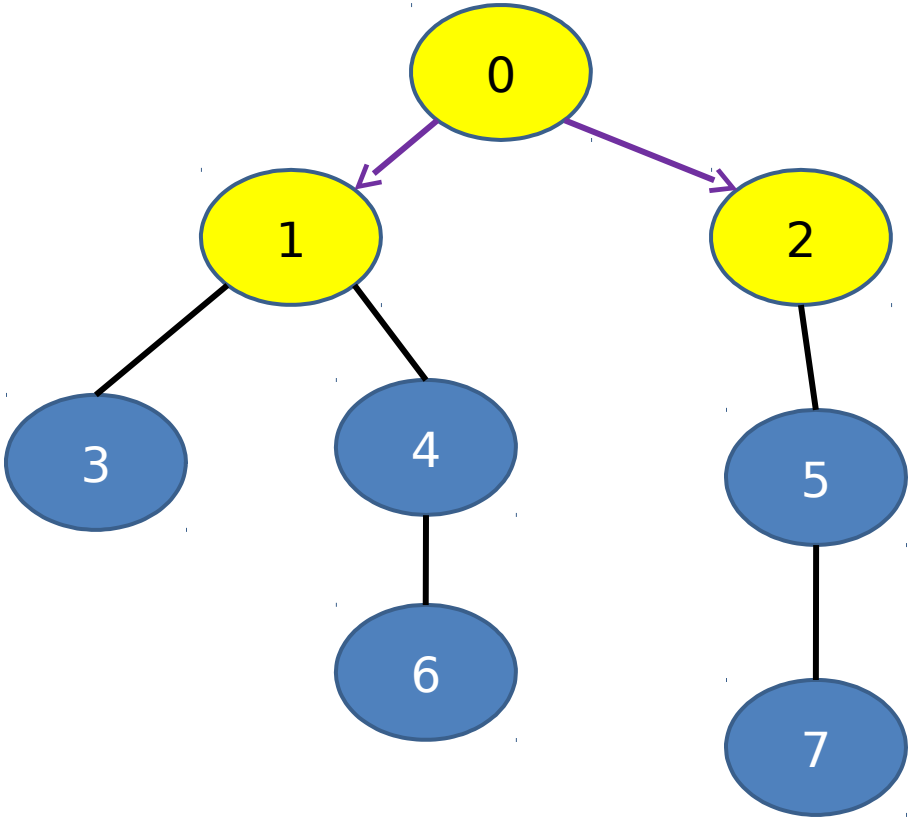
(B)readth-(F)irst-(S)earch

Queue

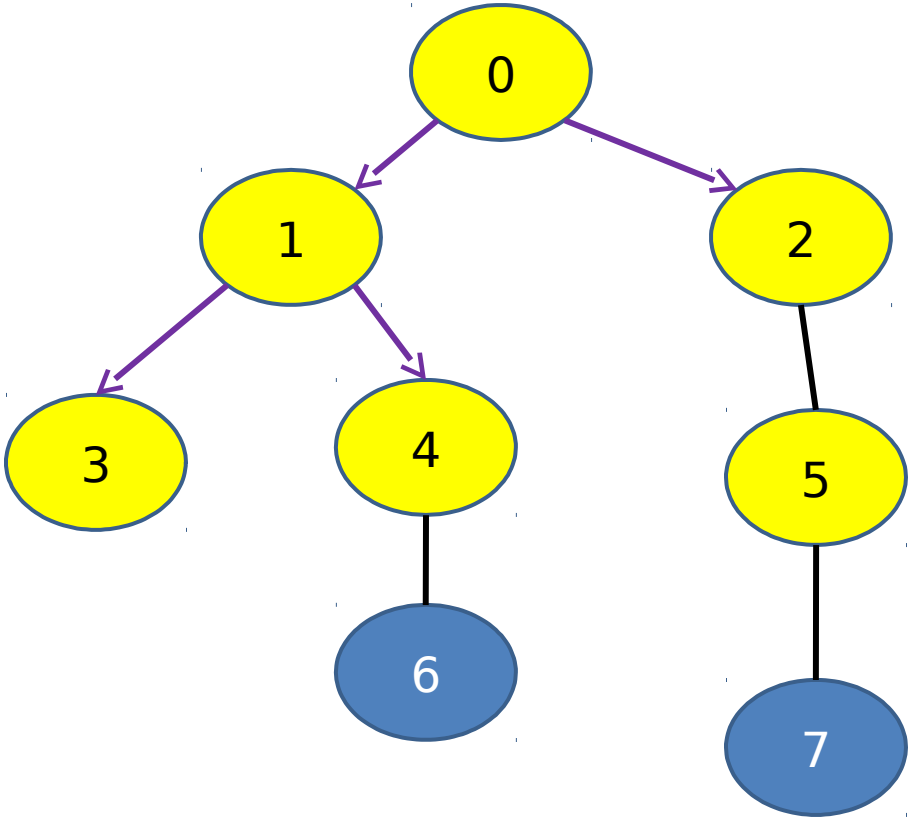
BFS



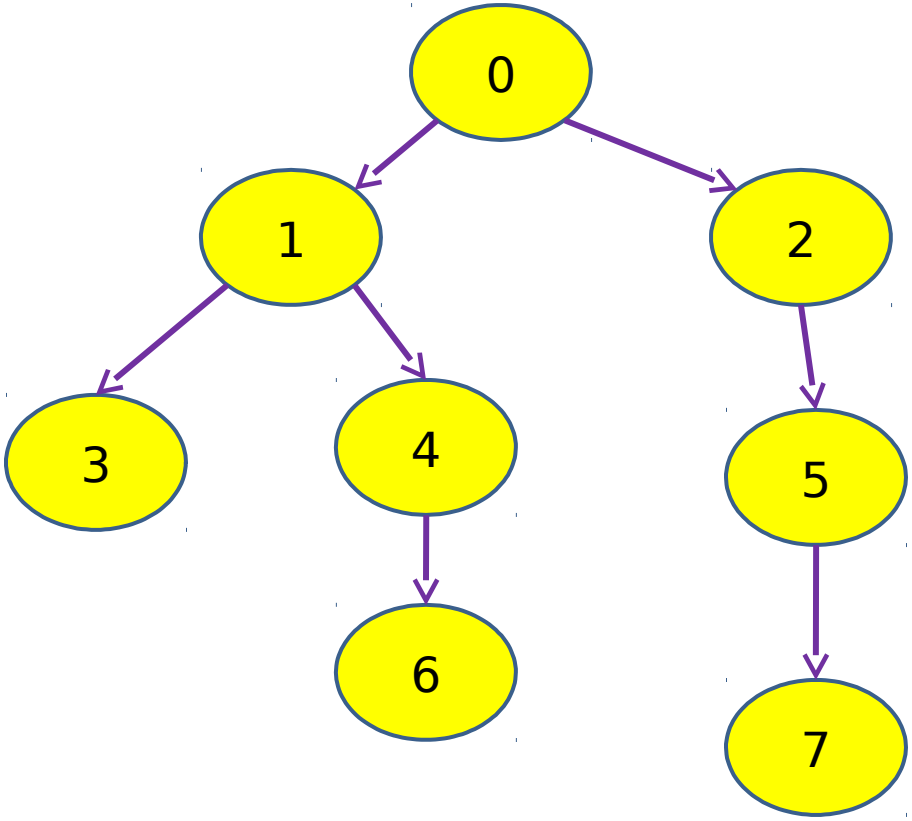
BFS



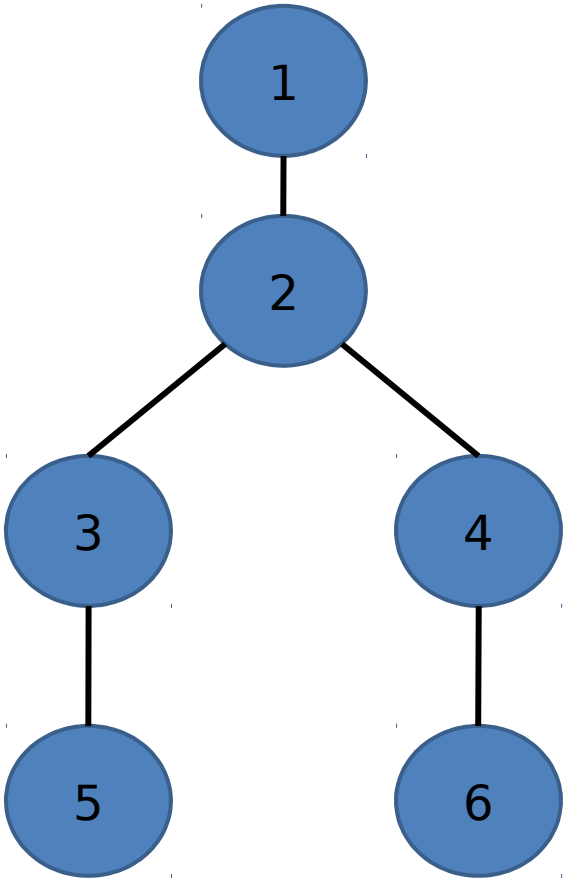
BFS



BFS

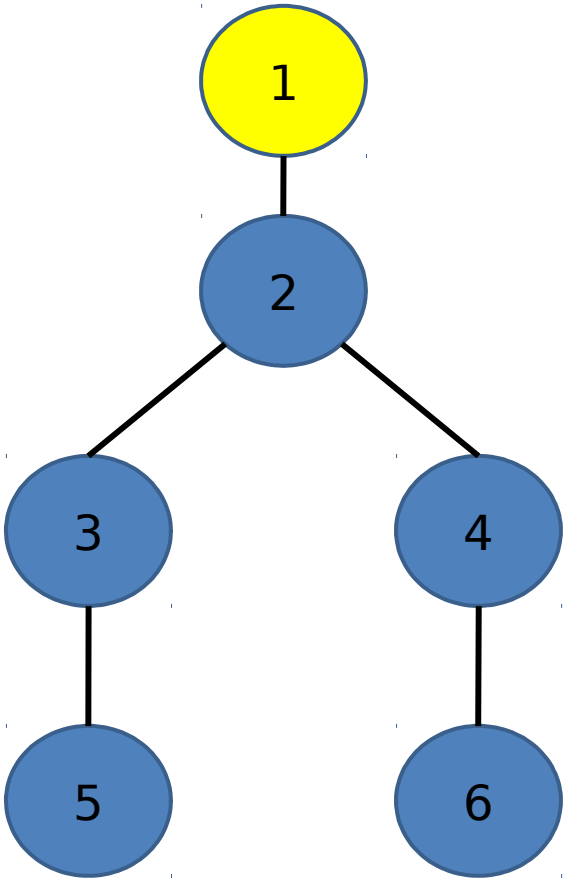


BFS



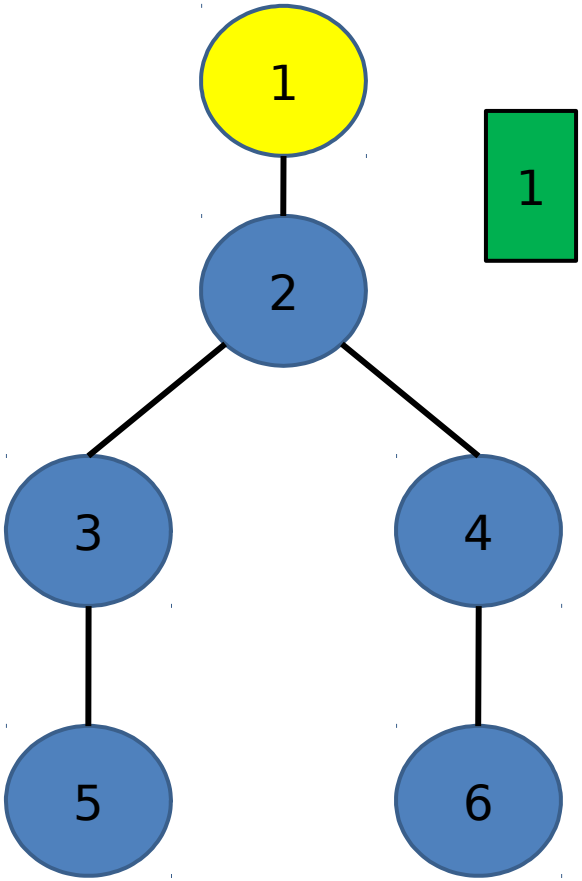
Queue

BFS



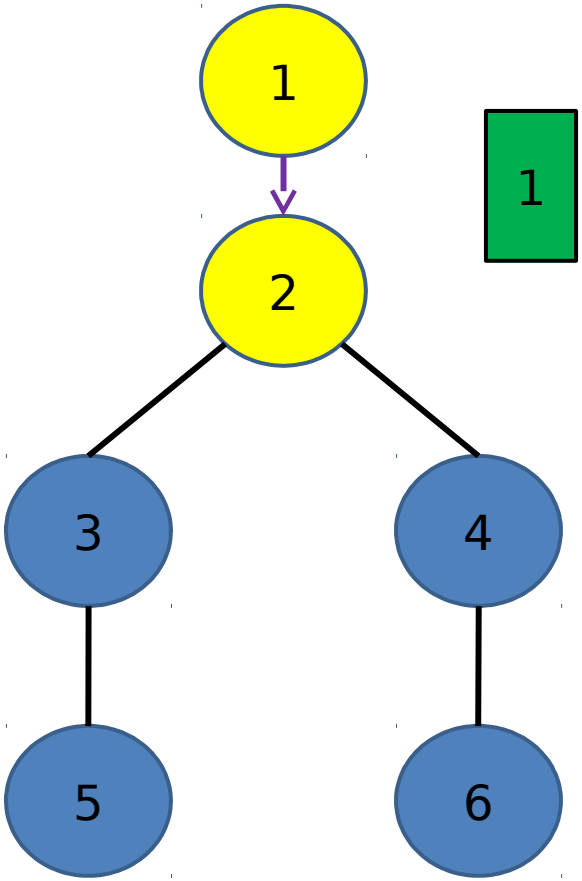
Queue

BFS



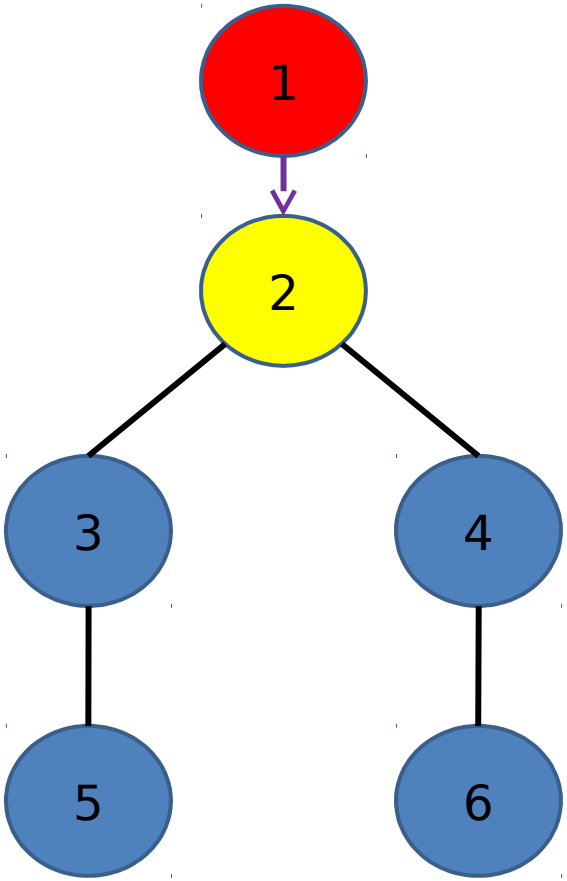
Queue

BFS



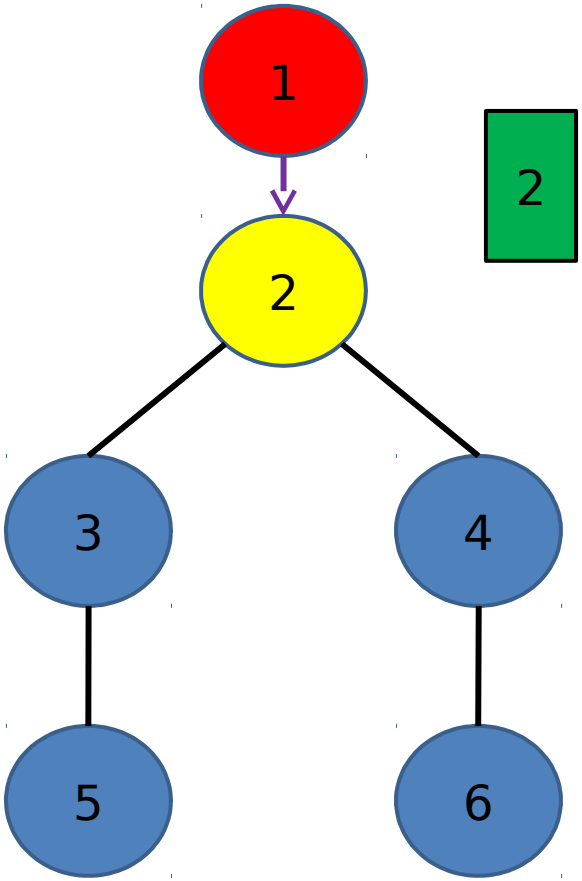
Queue

BFS



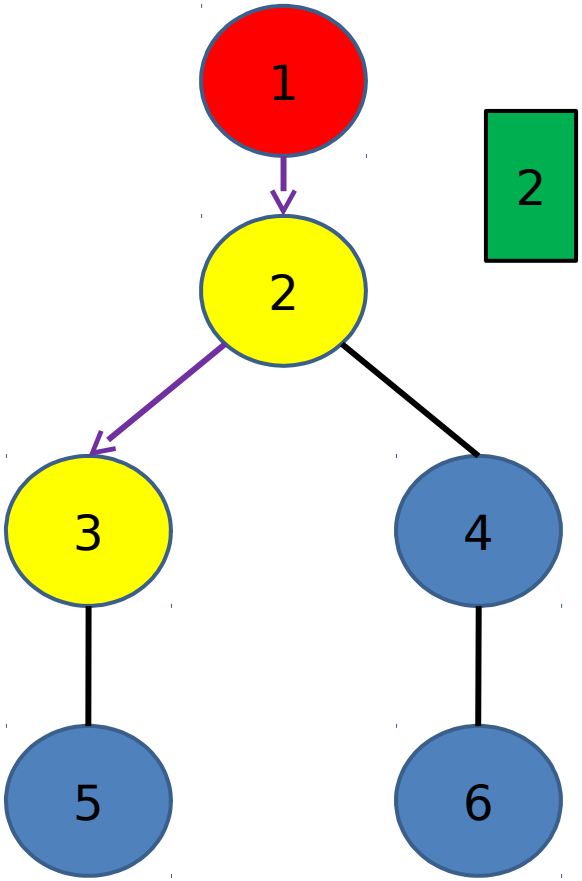
Queue

BFS



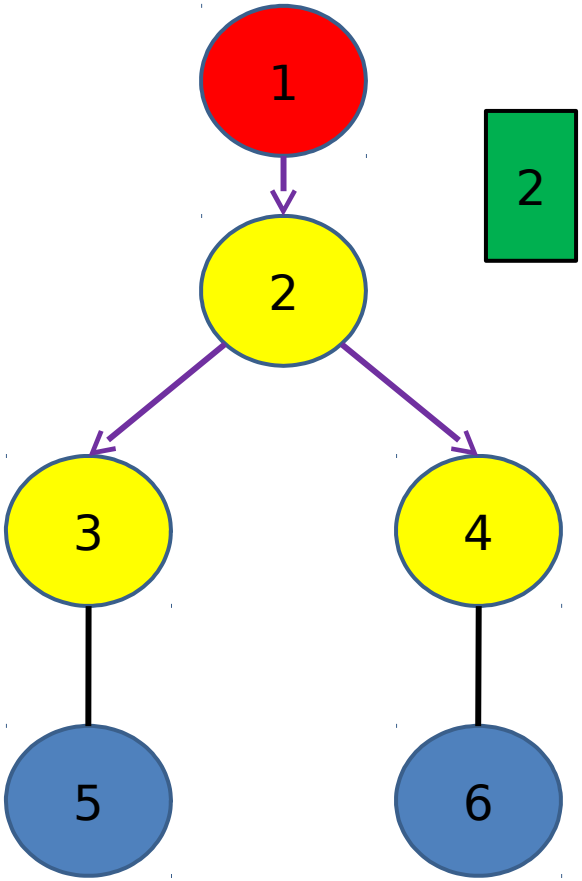
Queue

BFS



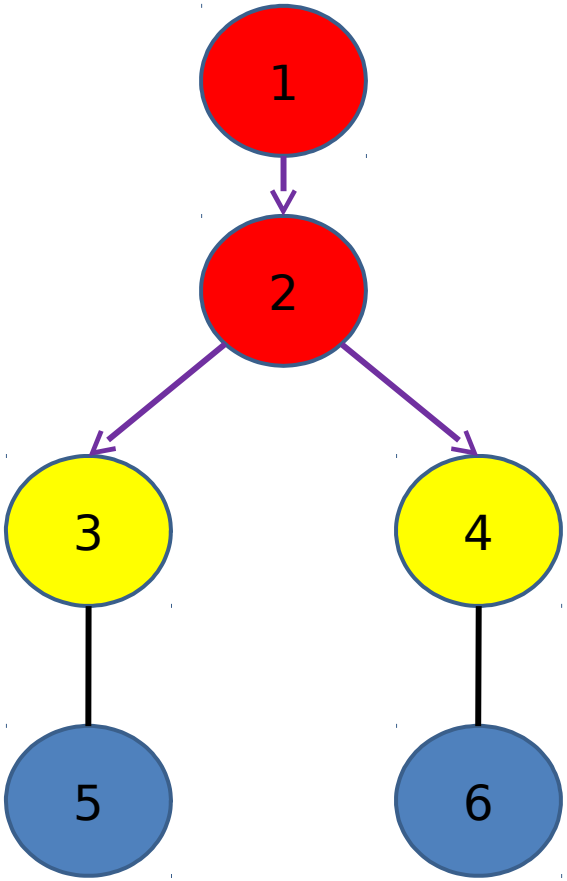
Queue

BFS



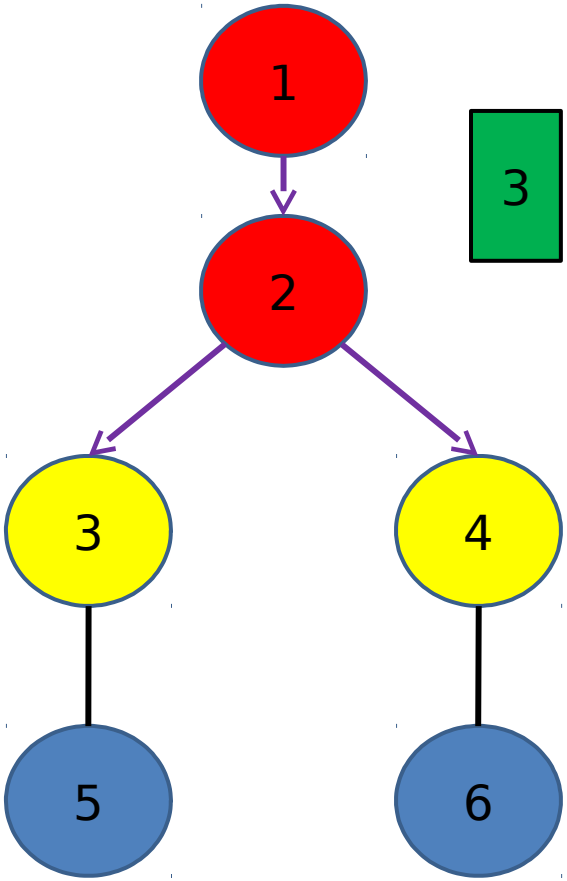
Queue

BFS



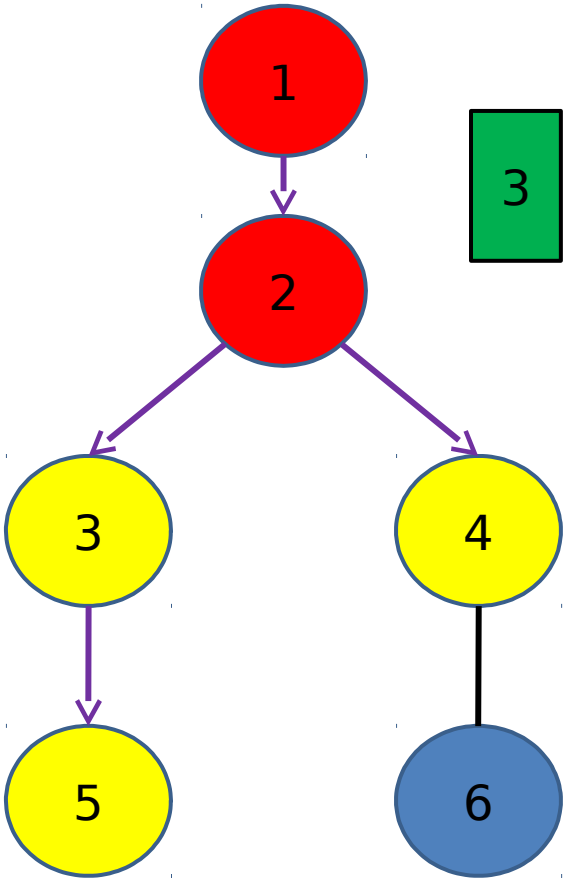
Queue

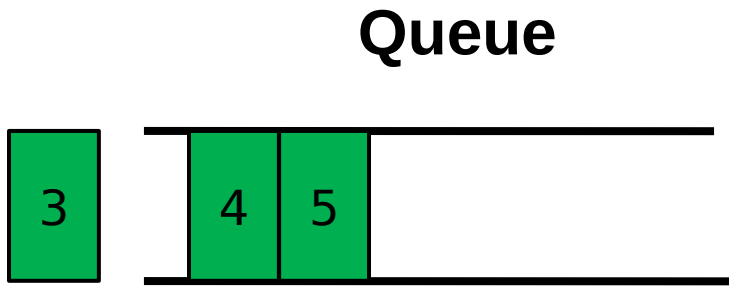
BFS



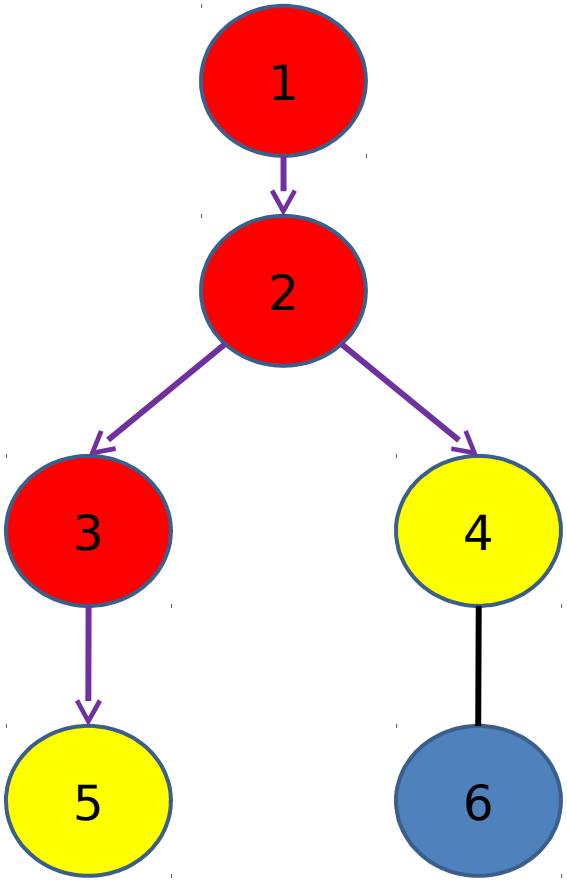
Queue

BFS

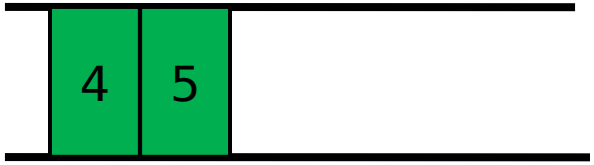




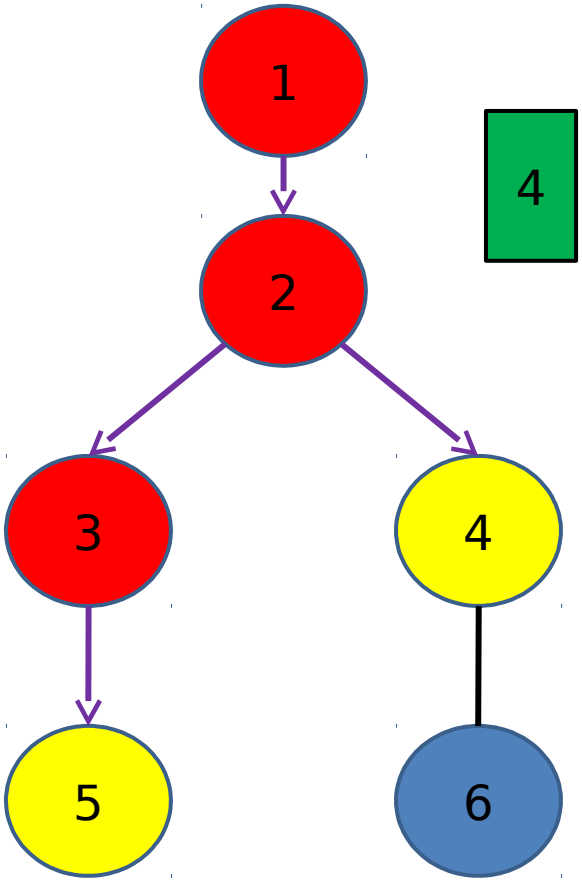
BFS



Queue

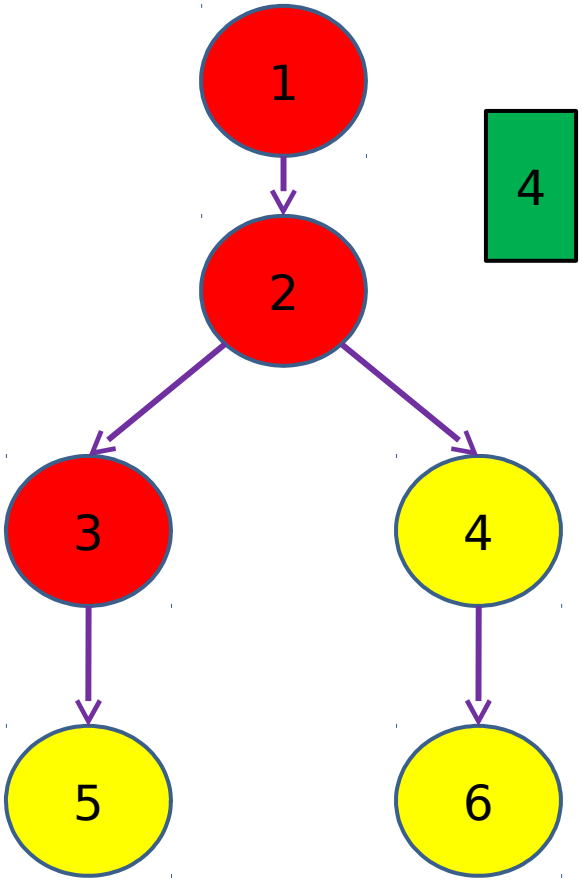


BFS

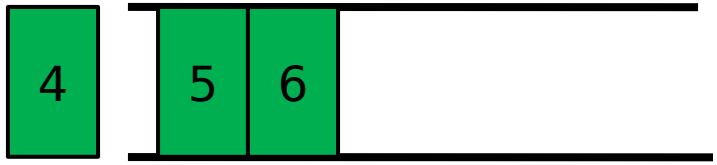


Queue

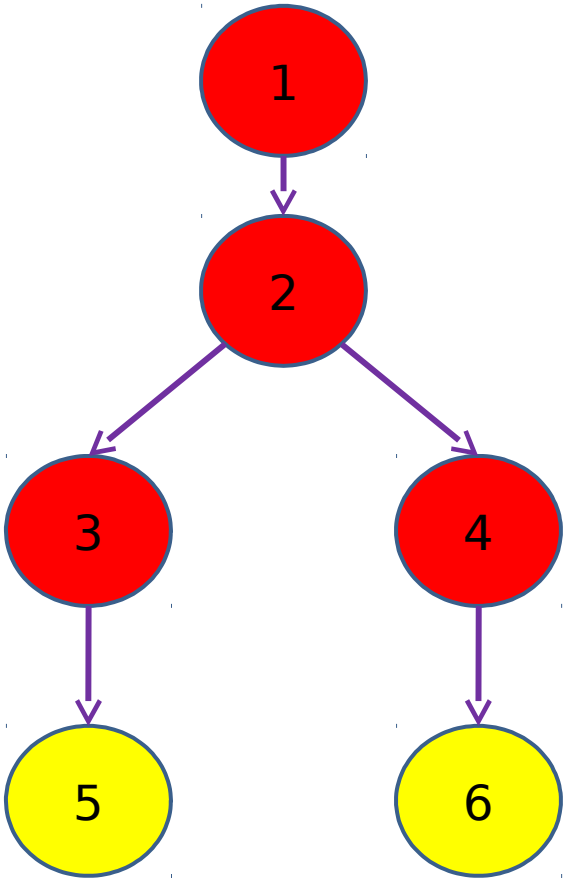
BFS



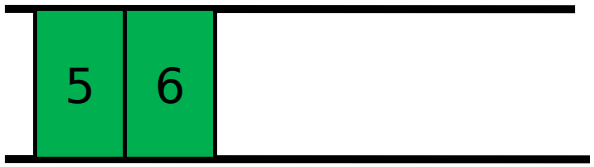
Queue



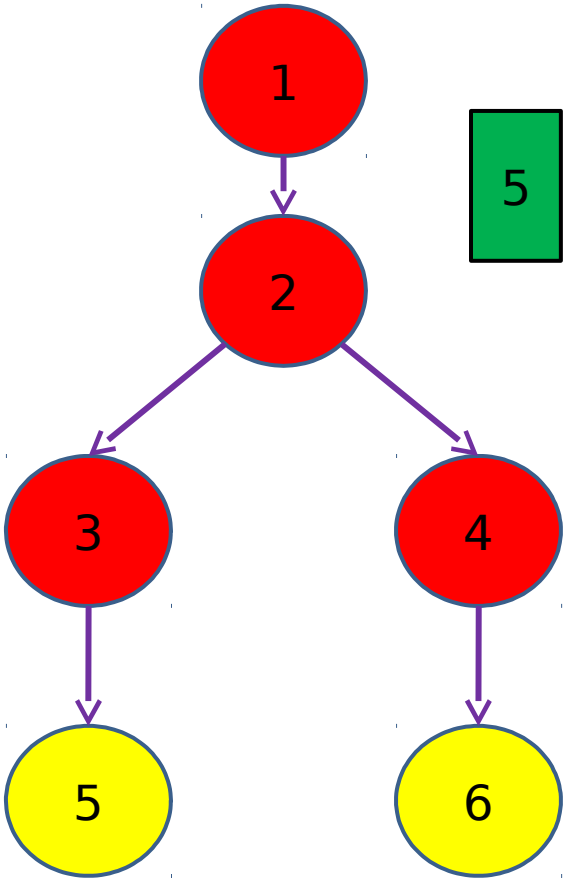
BFS



Queue

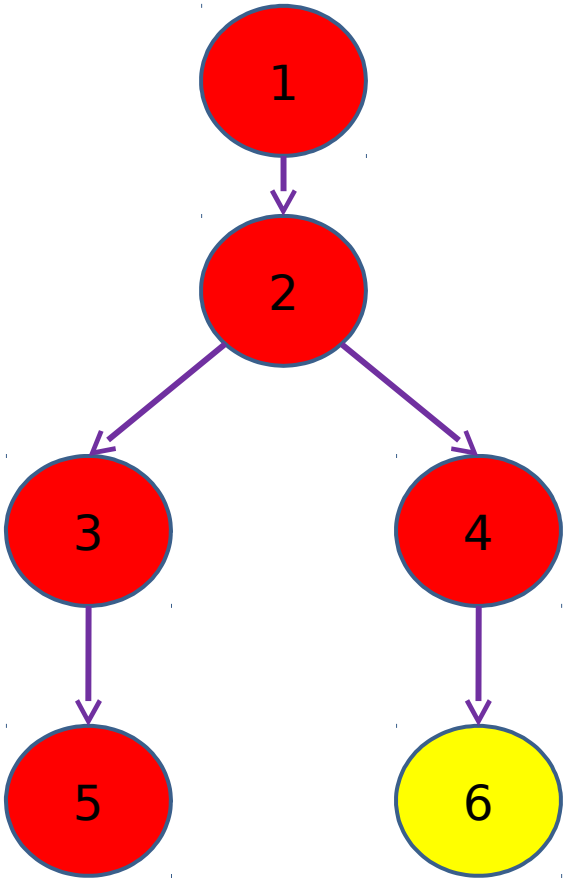


BFS



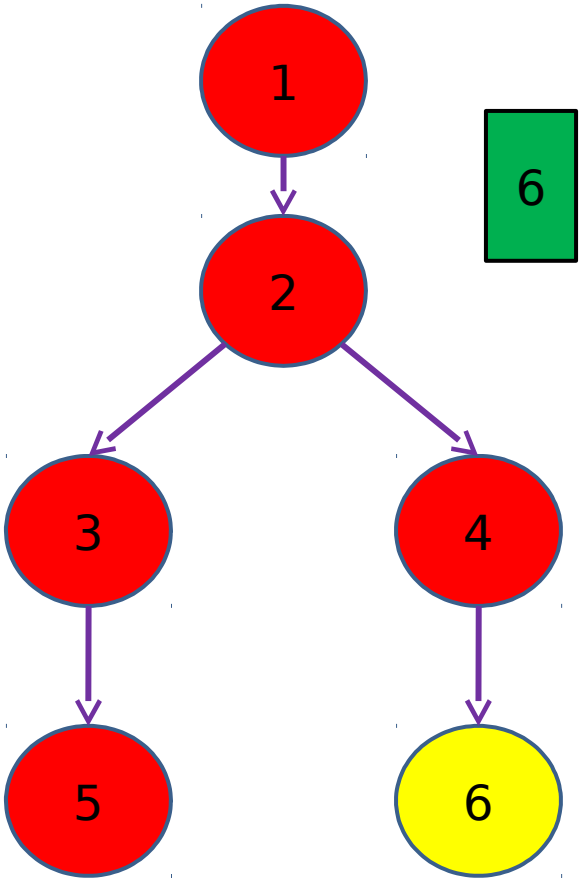
Queue

BFS



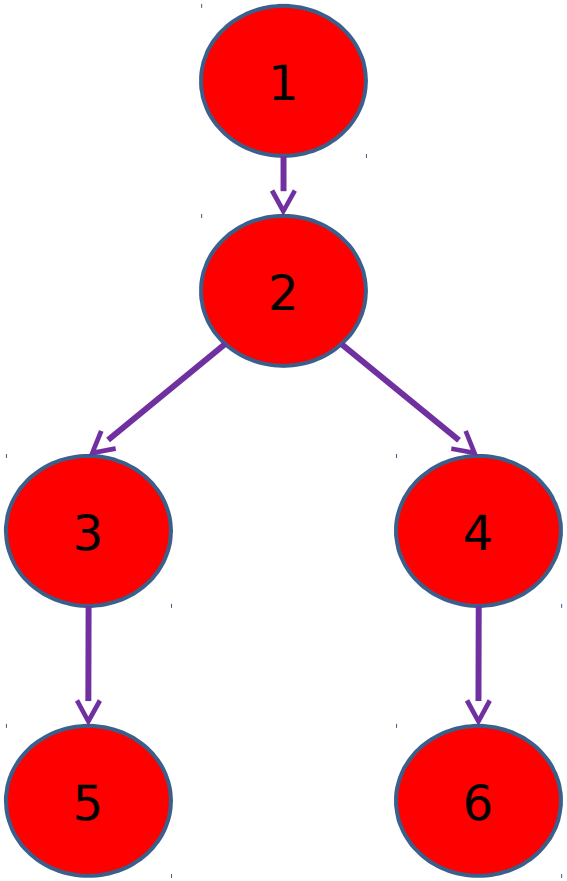
Queue

BFS



Queue

BFS



Queue

BFS

- **Source Code** (adjacency list)

```
void BFS(int root)
{
    queue<int> que;
    que.push(root);
    visited[root]=true;

    while(!que.empty())
    {
        int cur=que.front();
        que.pop();

        for(int i=0;i<adj[cur].size();i++)
        {
            int next=adj[cur][i];
            if(!visited[next])
            {
                visited[next]=true;
                que.push(next);
            }
        }
    }
    return;
}
```

BFS

- **Practice**

[UVA-532]　Dungeon Master

- **Skill:**

  int dir[4][2]={ {1,0},{-1,0},{0,1},{0,-1} }

  ☒ search four directions

- **Skill:**

```
int dir[4][2]={ {1,0},{-1,0},{0,1},{0,-1} };

void DFS(int cur_x,int cur_y)
{
    vis[cur_x][cur_y]=1;

    for(int i=0;i<4;i++)
    {
        int nx=cur_x+dir[i][0];
        int ny=cur_y+dir[i][1];

        //watch for boundary
        if( !vis[nx][ny])
            DFS(nx,ny);
    }
}
```

Time Complexity

DFS O(V + E)

BFS O(V + E)

V: the number of nodes

E: the number of edges

(adjacency list)

Time Complexity

DFS O(V^2)

BFS O(V^2)

V: the number of nodes

E: the number of edges

(adjacency matrix)

HW3

Totally 30 problems

UVa:

260,336,352,383,439,532,539,567,571,601,705,

762,10004,10009,10474, 10505,10592,10603, 10946, 11624, 532, 572

POJ:

1129,1154,1416,1606,1753,1915,1979,2243

acm International Collegiate Programming Contest

IBM event sponsor

Thank for Your Attention

Made By louis7340 & Edit by