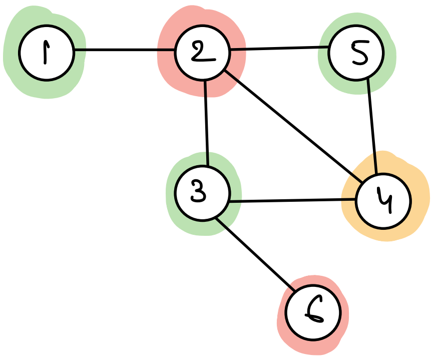
Graph Coloring {Undirected Graphs}

⇒ Minimum colors required to color each node of a graph s.t no two adjacent nodes have same color.

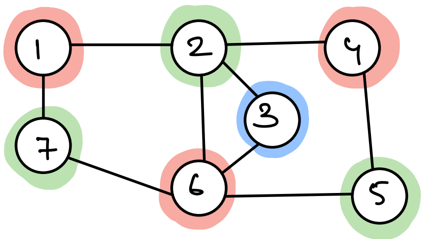
↳ Chromatic Number

Eg



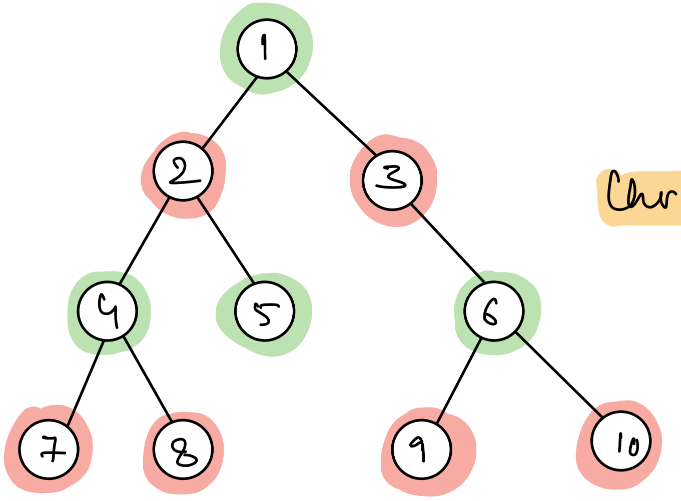
Chromatic Number = 3

Eg



Chromatic Number = 3

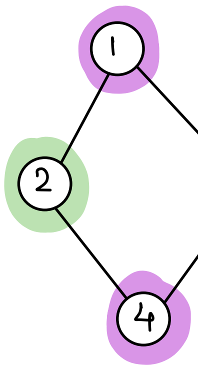
Eg



Chromatic Number = 2

Bipartite Graph

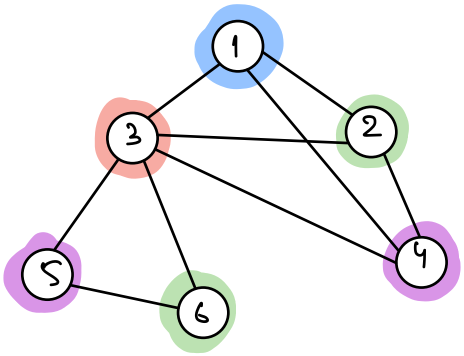
Eg



⇒ Chromatic Number = 2

Bipartite Graph

Eg



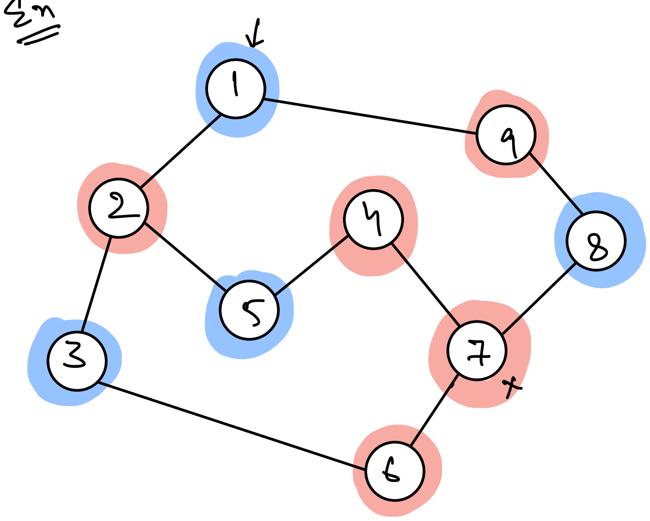
Chromatic Number = 4

Note : Calucating chromatic number of a graph takes exponential time Complexity.

Bipartite Graphs :-

⇒ If chromatic number of a graph is 2, then graph is Bipartite.

Eg



Col[i] → 0: No Color
→ 1: Blue
→ 2: Red

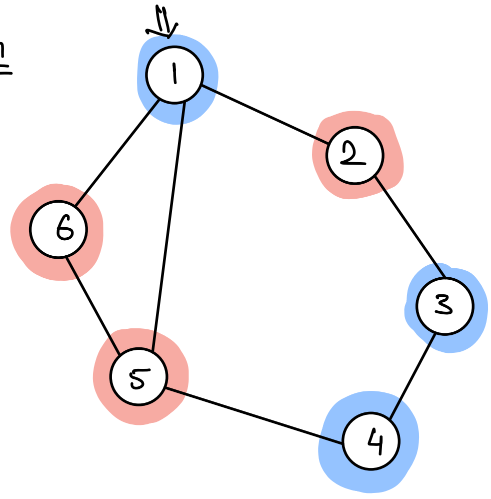
int col[10]:

0	1	2	3	4	5	6	7	8	9
x	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~
	1	2	1	2	1	2	2	1	2

~~1~~ ~~2~~ ~~9~~ ~~3~~ ~~5~~ ~~8~~ 6
↑

⇒ Not a Bipartite graph.

Eg



Adj list

1 → 2, 5, 6

2 → 1, 3

3 → 2, 4

4 → 3, 5

5 → 1, 4, 6

6 → 1, 5

int Color[7]:

0	1	2	3	4	5	6
x	~~0~~1	~~0~~2	~~0~~1	~~0~~1	2~~0~~	~~0~~2

Q ~~1~~ ~~2~~ 5 6 3

1: Ⓑ { 2 : R
5 : R
6 : R }

2: Ⓡ { 1 ✗
3 : B }

5: Ⓡ { 1 : B ✓
4 : B ✓
6 : B ✗ }

→ Not a Bipartite Graph.

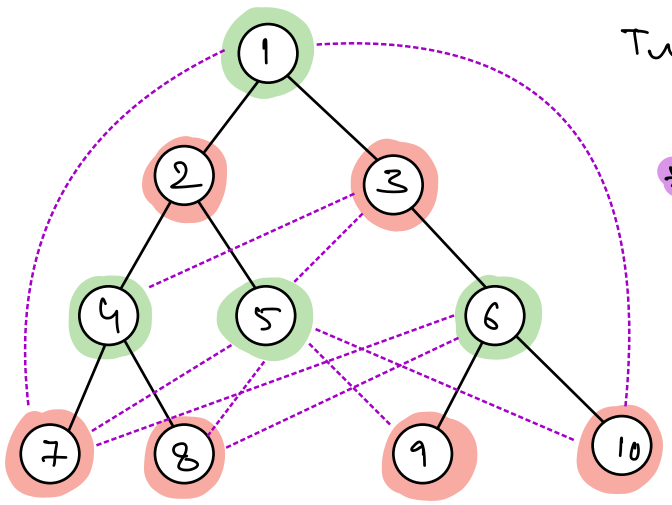
Code

```
bool isBipartite ( List<int> g[], int n) {
    int col[N+1] = {0};
    queue<int> q;
     for(i=1; i<=n; i++) {
        col[i] = 1; q.insert(i);
        while (q.size() > 0) {
             u = q.front();
             q.dequeue();
             //get adjacent nodes of ⓤ
             for(j=0; j< g[u].size(); j++){
                  v= g[u][j]
                  if(col[v] == col[u]){
                      return false;
                  }
                  // Node v is NOT colored.
                  if(col[v] ==0){
                       col[v]= 3- col[u]
                       q.enqueue(v);
                   }
             }
    }
    return true;
}
```

TC: O(N+E) → O(E)

SC: O(N+E) → O(E)

Q: Given a Tree, max no. of edges which can be added such that graph remains Bipartite.



Tree ⇒ Always Bipartite

* We can create an edge b/w 2 nodes with different colors.

nodes : 10

nodes with green : 4

nodes with red : 6

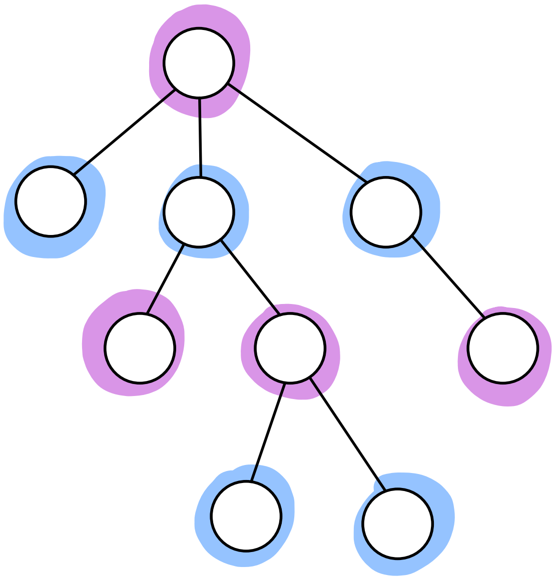
1	2
4	3
5	7
6	8
	9
	10

Edges = 4 × 6 = 24

ans = 24 - Current no. of edges.

= 24 - 9 = 15

Ex



Blue nodes : 5

Purple nodes : 4

Edges = 20

ans = 20 - 8

= 12

⇒

$$\underbrace{\text{\# of nodes with col 1}}_{x} * \underbrace{\text{no. of nodes with col 2}}_{y} - \underbrace{\left\{\text{Edges which are already present}\right\}}_{\textcircled{2}}$$

ans = $x * y - 2$

Ex