Breadth First Search

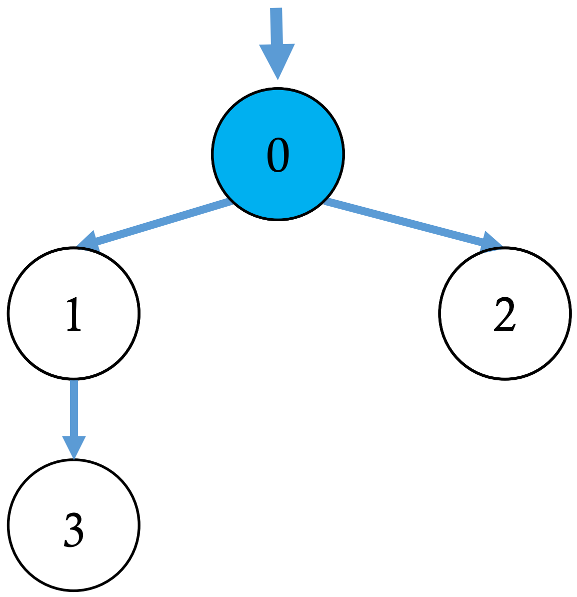
CSE Dept, MIST

Story Behind BFS

- A monarchy maintains a hierarchical plan
- The first king is decided by people
- Then the rule is simple
- When anyone becomes king he nominates all his children for next king
- When any king departs, the person who stands in front of the nomination list becomes the king
- No person can be king for the second time

Story Behind BFS

- Lets follow the hierarchy
- The first king decided by people is **0**
- Now, find the order of the king

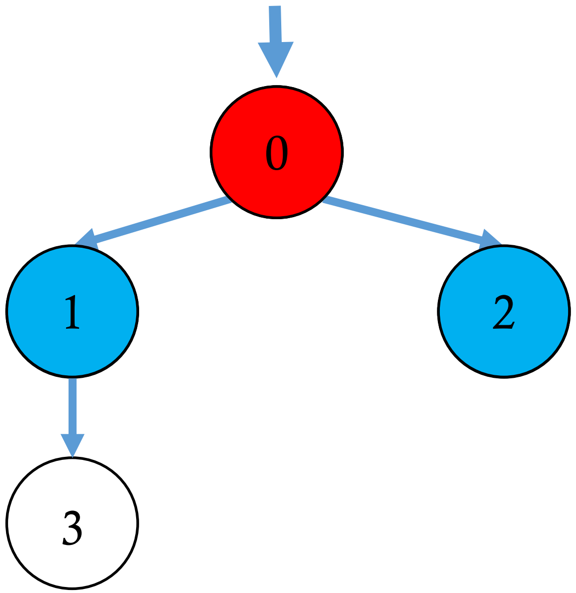


King: 0

Nomination List: 0

Story Behind BFS

- Lets follow the hierarchy
- The first king decided by people is **0**
- Now, find the order of the king

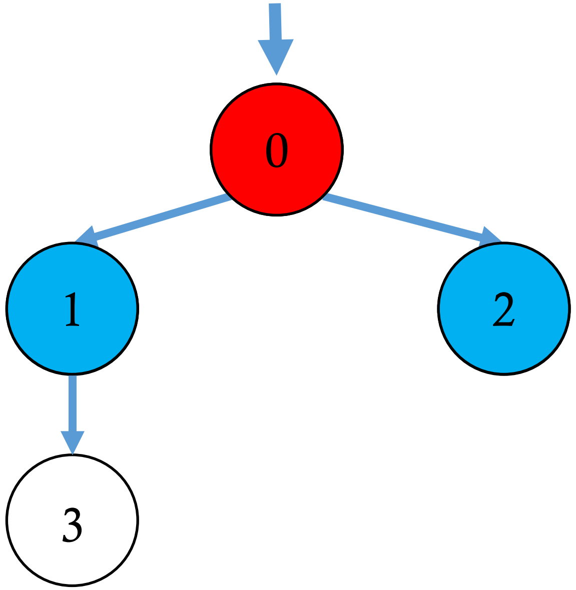


King: 0

Nomination List: 1 2

Story Behind BFS

- Lets follow the hierarchy
- The first king decided by people is **0**
- Now, find the order of the king

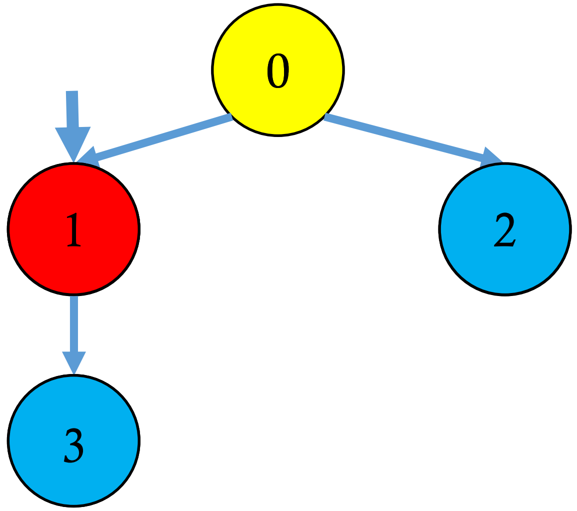


King: 0

Nomination List: 1 2

Story Behind BFS

- Lets follow the hierarchy
- The first king decided by people is **0**
- Now, find the order of the king

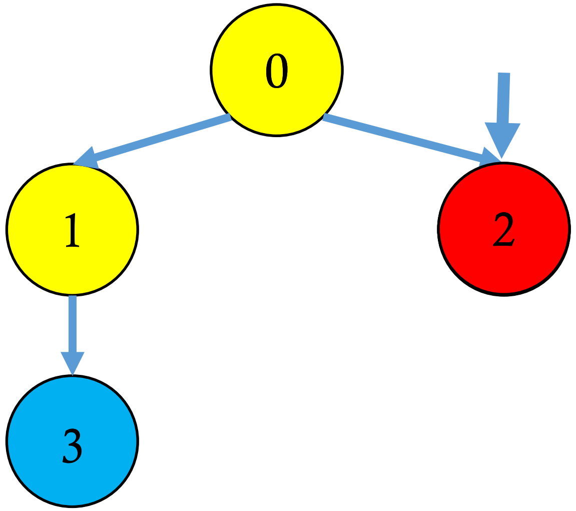


King: 1

Nomination List: 1 3

Story Behind BFS

- Lets follow the hierarchy
- The first king decided by people is **0**
- Now, find the order of the king

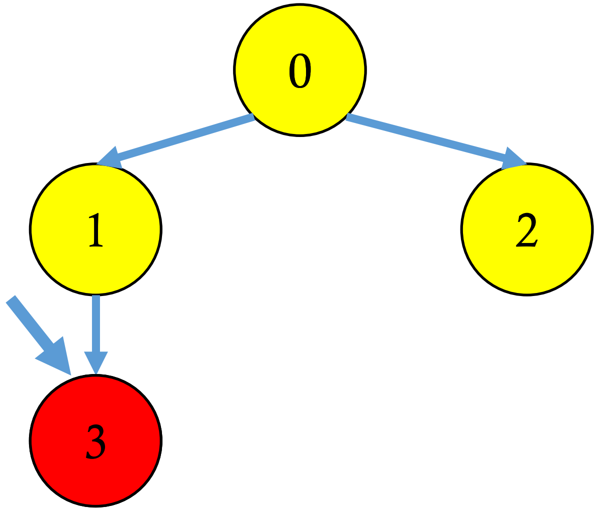


King: 2

Nomination List: 2 3

Story Behind BFS

- Lets follow the hierarchy
- The first king decided by people is **0**
- Now, find the order of the king

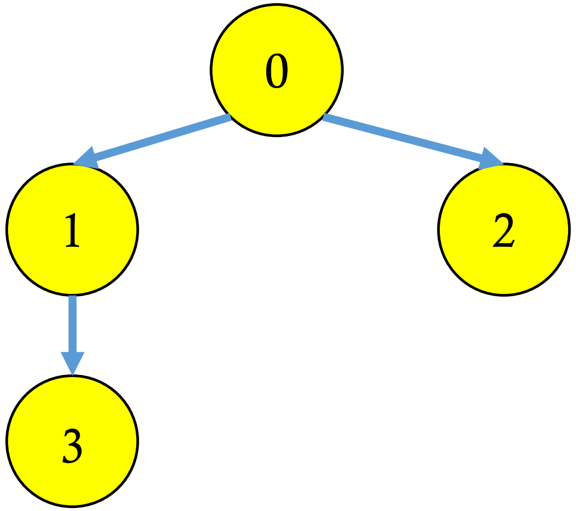


King: 3

Nomination List: 3

Story Behind BFS

- Lets follow the hierarchy
- The first king decided by people is **0**
- Now, find the order of the king



King:

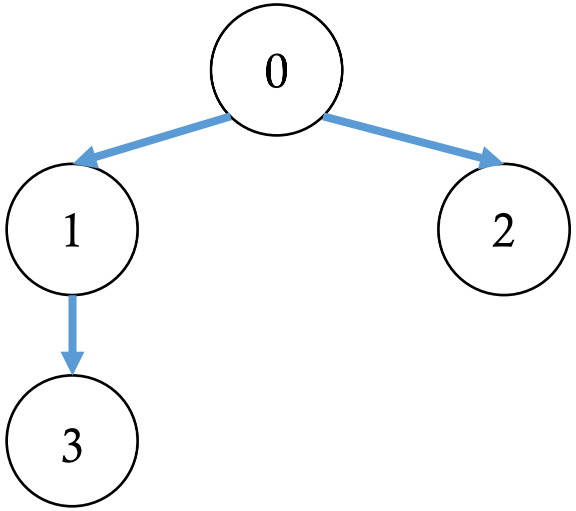
Nomination List:

King Order: 0 1 2 3

Required Data Structures

- 2D array for adjacency matrix
- Queue for nomination list

Adjacency Matrix

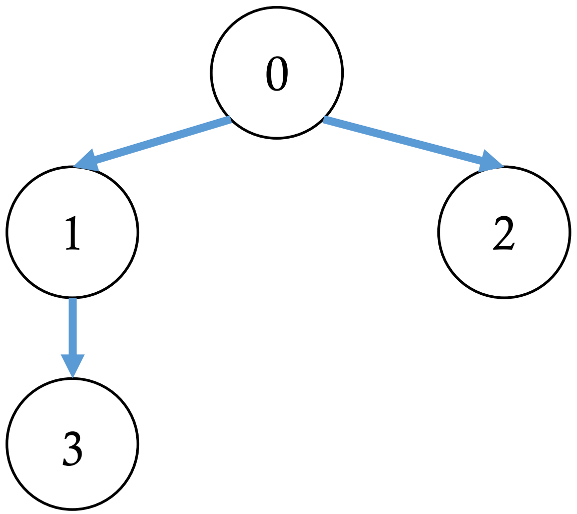


	0	1	2	3
0	0	1	1	0
1	0	0	0	1
2	0	0	0	0
3	0	0	0	0

Adjacency Matrix

- Sometimes it is needed to declare a large size adjacency matrix
- Just bound it at the time of using

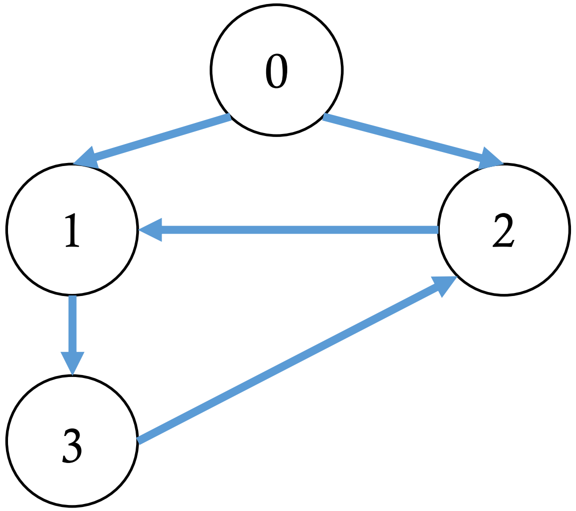
	0	1	2	3	4	5
0	0	1	1	0	0	0
1	0	0	0	1	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0



Lets Do Some Coding

What If

- There is an edge from Vertex-2 to Vertex-1 & Vertex-3 to Vertex-2
- Find the King Order this time (Starting from 0)

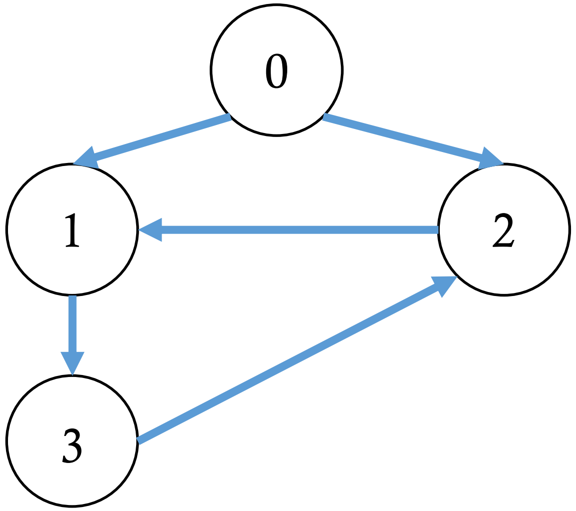


King:

Nomination List: 0

What If

- There is an edge from Vertex-2 to Vertex-1 & Vertex-3 to Vertex-2
- Find the King Order this time (Starting from 0)

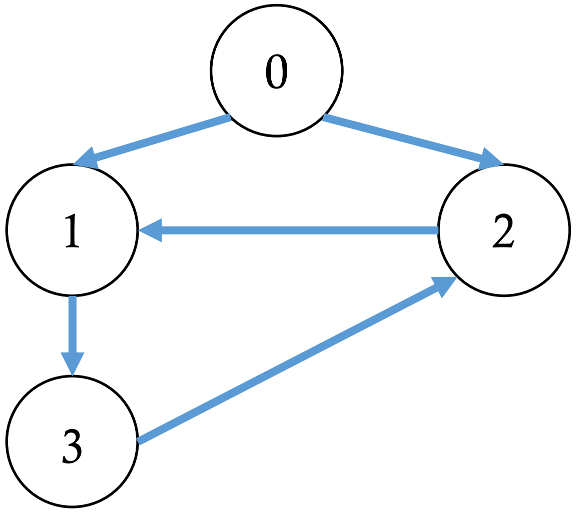


King: 0

Nomination List: 1 2

What If

- There is an edge from Vertex-2 to Vertex-1 & Vertex-3 to Vertex-2
- Find the King Order this time (Starting from 0)

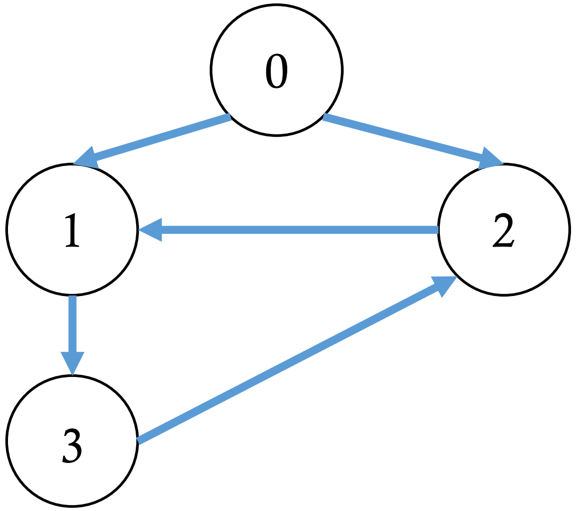


King: 1

Nomination List: 2 3

What If

- There is an edge from Vertex-2 to Vertex-1 & Vertex-3 to Vertex-2
- Find the King Order this time (Starting from 0)

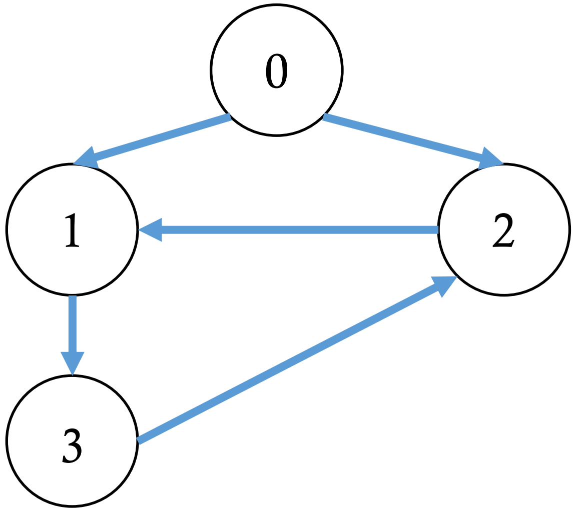


King: 2

Nomination List: 3 1

What If

- There is an edge from Vertex-2 to Vertex-1 & Vertex-3 to Vertex-2
- Find the King Order this time (Starting from 0)

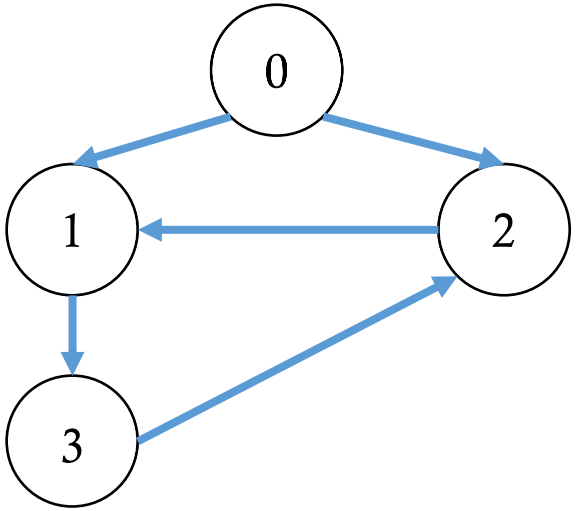


King: 3

Nomination List: 1 2

What If

- There is an edge from Vertex-2 to Vertex-1 & Vertex-3 to Vertex-2
- Find the King Order this time (Starting from 0)

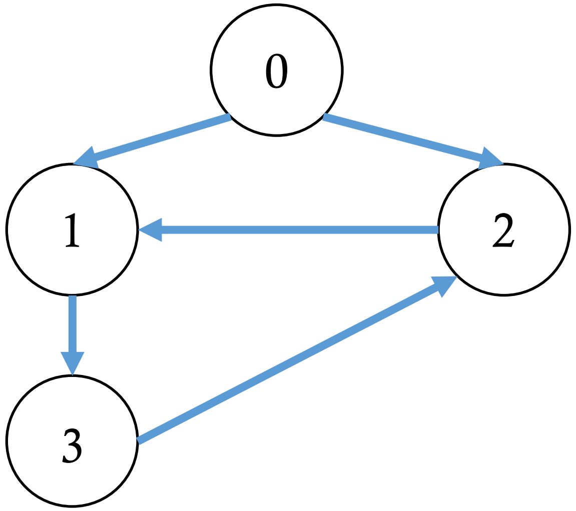


King: 1

Nomination List: 2 3

What If

- There is an edge from Vertex-2 to Vertex-1 & Vertex-3 to Vertex-2
- Find the King Order this time (Starting from 0)



King: 2

Nomination List: 3 1

This becomes an infinite process!

Because same element is entering into the queue for multiple times!

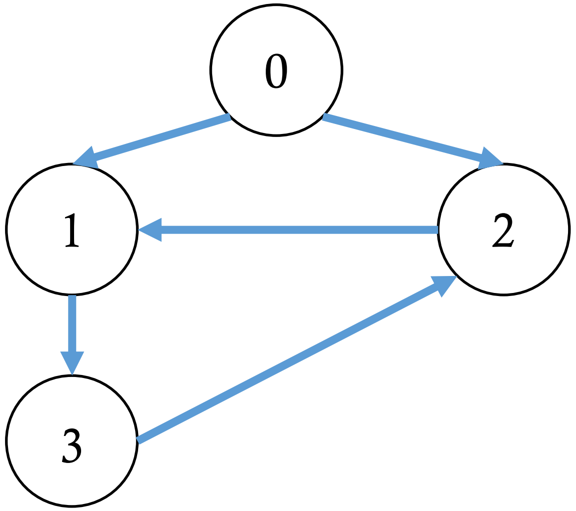
Solution

- Mark a vertex at the time of pushing in the queue
- Checking the mark of a vertex before pushing in the queue
- Real time example: Election Center

- Required Data Structure
    - 1D Array of dimension **n**

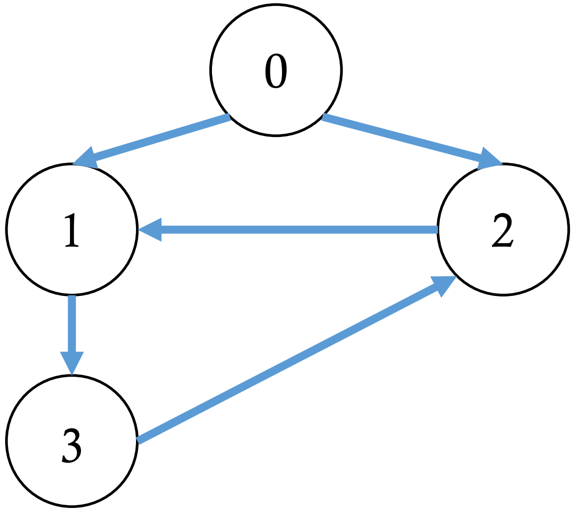
Implementation Idea

- Treat the vertices as the index of the array
- Initially set “No Mark” for all vertices
- A vertex will be pushed in the queue if it has “No Mark”
- Change the mark of a vertex after pushing it to the queue



0	0	0	0		
0	1	2	3	4	5

Implementation Idea



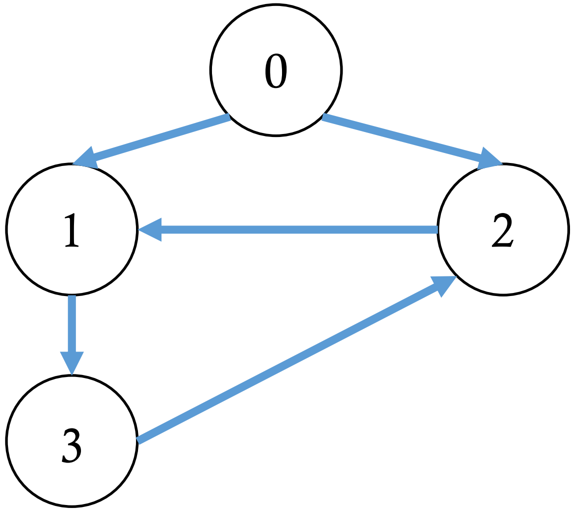
King:

Nomination List: 0

mark	1	0	0	0		
	0	1	2	3	4	5

mark[0] = 1

Implementation Idea



King: 0

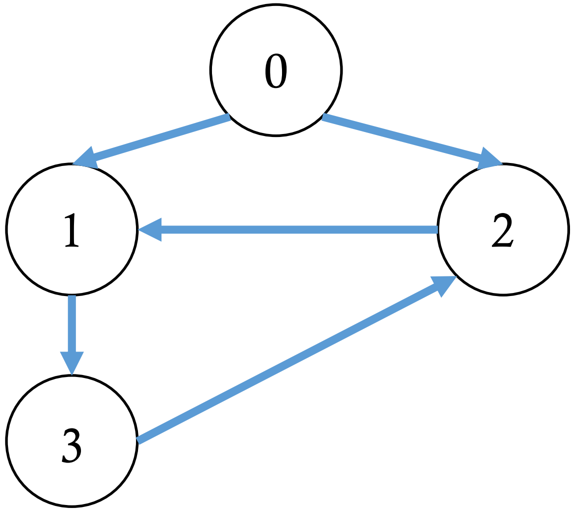
Nomination List: 1 2

mark	1	1	1	0		
	0	1	2	3	4	5

mark[1] = 1

mark[2] = 1

Implementation Idea



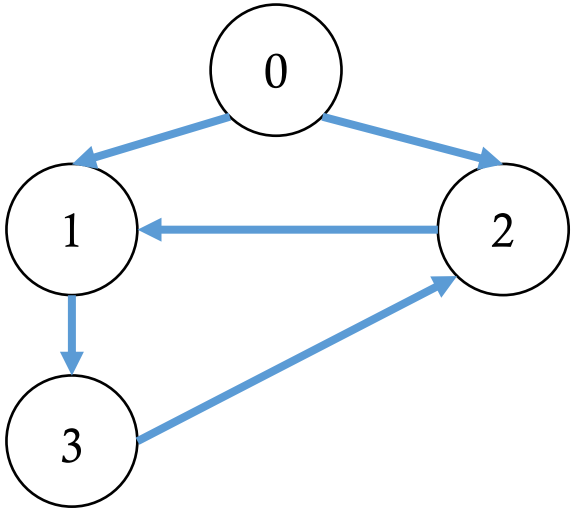
King: 1

Nomination List: 2 3

mark	1	1	1	1		
	0	1	2	3	4	5

mark[3] = 1

Implementation Idea

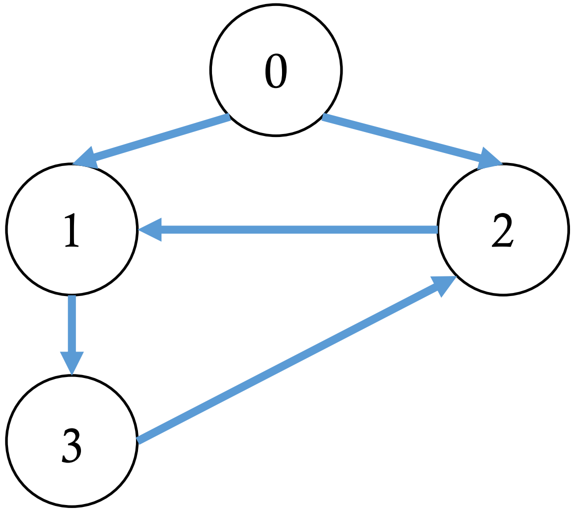


King: 2

Nomination List: 3

	0	1	2	3	4	5
mark	1	1	1	1		

Implementation Idea



King: 3

Nomination List:

mark	1	1	1	1		
	0	1	2	3	4	5

Lets Do Some Coding

QUESTIONS