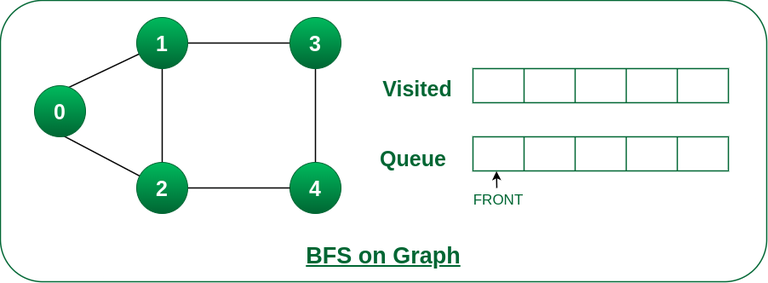
***BFS(Breadth First Search)***

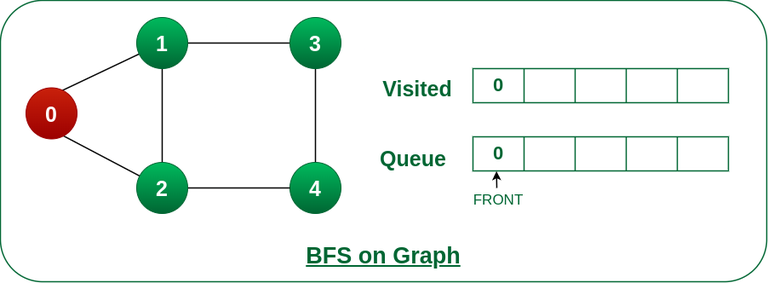
***------------------------------------------***

***Step1:****Initially queue and visited arrays are empty.*

**

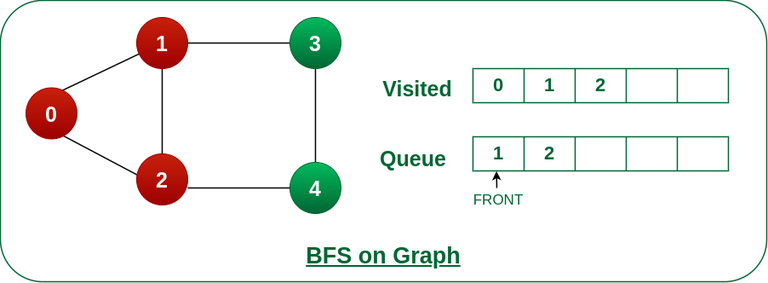
*Queue and visited arrays are empty initially.*

***Step2:****Push node 0 into queue and mark it visited.*

**

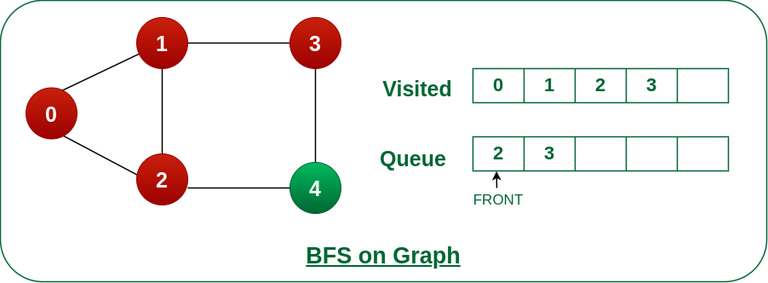
*Push node 0 into queue and mark it visited.*

***Step 3:****Remove node 0 from the front of queue and visit the unvisited neighbours and push them into queue.*

**

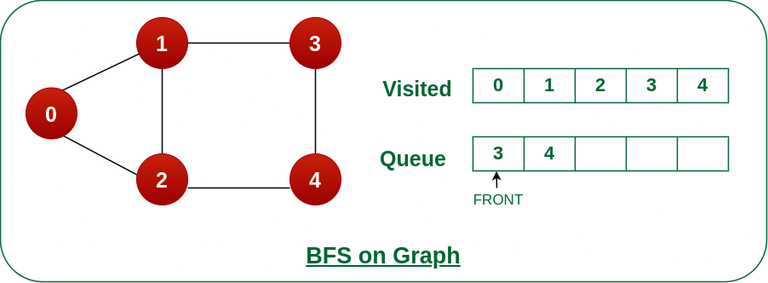
*Remove node 0 from the front of queue and visited the unvisited neighbours and push into queue.*

***Step 4:****Remove node 1 from the front of queue and visit the unvisited neighbours and push them into queue.*

**

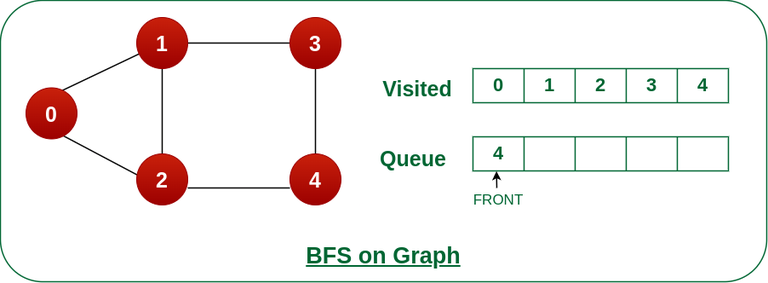
*Remove node 1 from the front of queue and visited the unvisited neighbours and push*

***Step 5:****Remove node 2 from the front of queue and visit the unvisited neighbours and push them into queue.*

**

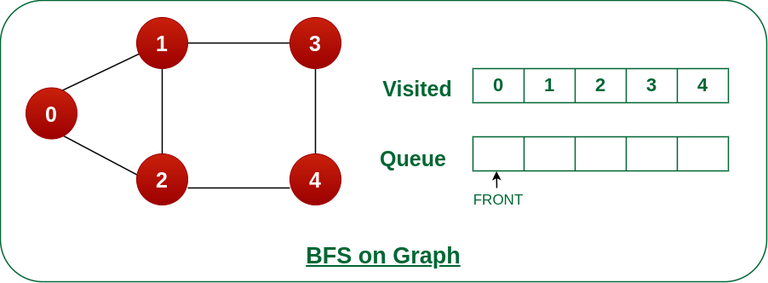
*Remove node 2 from the front of queue and visit the unvisited neighbours and push them into queue.*

***Step 6:****Remove node 3 from the front of queue and visit the unvisited neighbours and push them into queue.   
As we can see that every neighbours of node 3 is visited, so move to the next node that are in the front of the queue.*

**

*Remove node 3 from the front of queue and visit the unvisited neighbours and push them into queue.*

***Steps 7:****Remove node 4 from the front of queue and visit the unvisited neighbours and push them into queue.   
As we can see that every neighbours of node 4 are visited, so move to the next node that is in the front of the queue.*

**

*Remove node 4 from the front of queue and visit the unvisited neighbours and push them into queue.*

*Now, Queue becomes empty, So, term*

*Program of BFS*

*import java.util.LinkedList;*

*import java.util.Queue;*

*class Graph {*

*int vertices;*

*LinkedList<Integer>[] adjList;*

*Graph(int vertices)*

*{*

*this.vertices = vertices;*

*adjList = new LinkedList[vertices];*

*for (int i = 0; i < vertices; ++i)*

*adjList[i] = new LinkedList<>();*

*}*

*void addEdge(int u, int v) { adjList[u].add(v); }*

*void bfs(int startNode)*

*{*

*Queue<Integer> queue = new LinkedList<>();*

*boolean[] visited = new boolean[vertices];*

*visited[startNode] = true;*

*queue.add(startNode);*

*while (!queue.isEmpty()) {*

*int currentNode = queue.poll();*

*System.out.print(currentNode + " ");*

*for (int neighbor : adjList[currentNode]) {*

*if (!visited[neighbor]) {*

*visited[neighbor] = true;*

*queue.add(neighbor);*

*}*

*}*

*}*

*}*

*}*

*public class Main {*

*public static void main(String[] args)*

*{*

*int vertices = 5;*

*Graph graph = new Graph(vertices);*

*graph.addEdge(0, 1);*

*graph.addEdge(0, 2);*

*graph.addEdge(1, 3);*

*graph.addEdge(1, 4);*

*graph.addEdge(2, 4);*

*System.out.print(*

*"Breadth First Traversal starting from vertex 0: ");*

*graph.bfs(0);*

*}*

*}*