Program on uninformed search methods.(BFS)

```
import java.util.Iterator;
import java.util.LinkedList;
public class Main {
 public static void main(String[] args) {
 Graph g = new Graph(8);
 g.AddEdge(1, 2);
 g.AddEdge(1, 5);
 g.AddEdge(2, 3);
 g.AddEdge(2, 5);
 g.AddEdge(3, 4);
 g.AddEdge(4, 5);
 g.AddEdge(4, 6);
 g.AddEdge(5, 4);
 g.BFS(1);
 }
}
class Graph {
 private int NodeNumber;
 private LinkedList<Integer> AdjacentNodes[];
 Graph(int V) {
 AdjacentNodes = new LinkedList[V];
 for (int i = 0; i < AdjacentNodes.length; i++) {
 AdjacentNodes[i] = new LinkedList();
 }
 NodeNumber = V;
 }
 public void AddEdge(int v, int w) {
 AdjacentNodes[v].add(w);
 }
 public void BFS(int s) {
 boolean visited[] = new boolean[NodeNumber];
 for (int i = 0; i < NodeNumber; i++) {
 visited[i] = false;
 }
 LinkedList<Integer> queue = new LinkedList();
 visited[s] = true;
 queue.add(s);
 while (queue.size() != 0) {
 s = queue.poll();
 System.out.println("visiting " + s + " ");
 Iterator<Integer> i = AdjacentNodes[s].listIterator();
 while (i.hasNext()) {
 int n = i.next();
 if (!visited[n]) {
 visited[n] = true;
 queue.add(n);
 }
 }
```

```
 }
 }
}
```

Output:

```
Following is Breadth First Traversal:

visiting 1
visiting 2
visiting 5
visiting 3
visiting 4
visiting 6
```