```java
class DetectCycleInUndirectedGraph
{
    public boolean isCycle(int V, ArrayList<ArrayList<Integer>> graph)
    {
     boolean visited[]= new boolean [V];
     for(int i=0;i<visited.length;i++){
         if(!visited[i]){
             boolean stack[] = new boolean[visited.length];
              visited[i]=true;
              stack[i]=true;
             boolean hasCycle = dfs(i,graph,visited,stack,-1);
             stack[i]=false;
             if(hasCycle){
                 return true;
             }
         }
     }
      return false;
    }


    public static boolean dfs(int current,ArrayList<ArrayList<Integer>> graph,boolean[] visited,boolean stack[],int parent){
         int size=graph.get(current).size();
         for(int j=0;j<size;j++){
            int nextNode = graph.get(current).get(j);
            if(nextNode==parent){continue;}
            if(stack[nextNode]==true){
                return true;
            }
            else{
                if(!visited[nextNode]){
                    visited[nextNode]=true;
                    stack[nextNode]=true;
                    if(dfs(nextNode,graph,visited,stack,current)){return true;}
                    stack[nextNode]=false;
                }
            }
         }
         return false;
     }
}
```