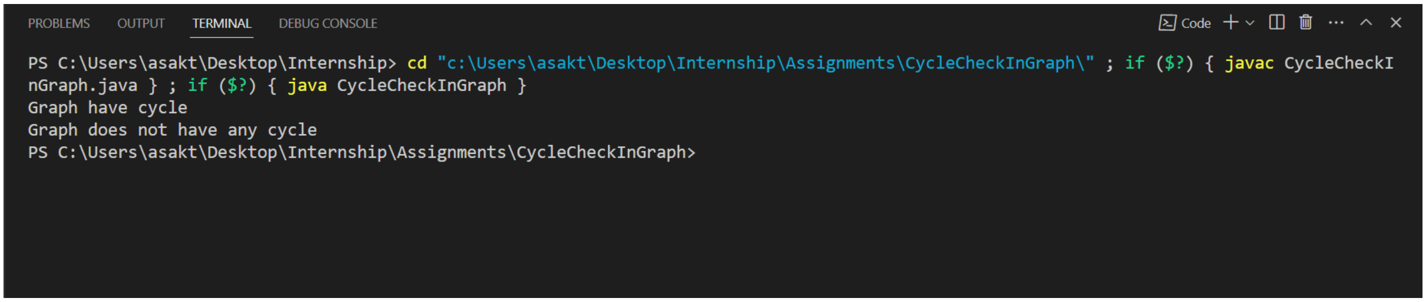
```
/*
 * Given a Directed Graph with  vertices (Numbered from  to ) and  edges,
 * check whether it contains any cycle or not.
 */

import java.util.*;
public class CycleCheckInDirectedGraph {
    public static boolean searchForCycle(int nodeIndex,
    ArrayList<int[]> graph,int visited[] ,int pathVisited[]){
        visited[nodeIndex] = 1;
        pathVisited[nodeIndex] = 1;
        for(int adjNode : graph.get(nodeIndex)){
            if(visited[adjNode] == 0){
                if(searchForCycle(adjNode, graph, visited, pathVisited)){
                    return true;
                }
            }
            else if(pathVisited[adjNode] == 1)
                return true;
        }
        pathVisited[nodeIndex] = 0;
        return false;
    }
    public static boolean checkCycle(int vertices, ArrayList<int[]> graph){
        int visited[] = new int[vertices];
        int pathVisited[] = new int[vertices];
        for (int i = 0; i < vertices; i++) {
            if(visited[i] == 0){
                if(searchForCycle(i,graph,visited,pathVisited))
                   return true;
            }
        }
        return false;
    }
    public static void main(String[] args) {
        // Here i represented the graph by Adjacency List (ArrayList in java)
        // and i used an ArrayList of Integer arrays which are representing
        // the nodes which are adjacent to that node and connected by a directed
//edge

        // Creating the graph 1
        ArrayList<int[]> graph1 = new ArrayList<>();

        // Inserting the nodes of the graph 1
        graph1.add(0, new int[]{1});
        graph1.add(1, new int[]{2});
        graph1.add(2, new int[]{3});
        graph1.add(3, new int[]{3});

        // Checking wheather the graph contains cycle or not
        if(checkCycle(4, graph1))
            System.out.println("Graph have cycle");
```

```java
        else
            System.out.println("Graph does not have any cycle");



        // Creating the graph 2
        ArrayList<int[]> graph2 = new ArrayList<>();

        // Inserting the nodes of the graph 2
        graph2.add(0, new int[]{1});
        graph2.add(1, new int[]{2});
        graph2.add(2, new int[]{});

        // Checking wheather the graph contains cycle or not
        if(checkCycle(3, graph2))
            System.out.println("Graph have cycle");
        else
            System.out.println("Graph does not have any cycle");
    }
}
```

// Output:



```
PS C:\Users\asakt\Desktop\Internship> cd "c:\Users\asakt\Desktop\Internship\Assignments\CycleCheckInGraph\" ; if ($?) { javac CycleCheckI
nGraph.java } ; if ($?) { java CycleCheckInGraph }
Graph have cycle
Graph does not have any cycle
PS C:\Users\asakt\Desktop\Internship\Assignments\CycleCheckInGraph>
```