Name: Jigar Siddhpura **SAPID:** 60004200155

DIV: C/C2 **Branch:** Computer Engineering

AI - EXPERIMENT 2

Aim: Implement search Algorithm DFID (Depth First Iterative Deepening)

Theory:

The iterative deepening algorithm is a combination of DFS and BFS algorithms. This search algorithm finds out the best depth limit and does it by gradually increasing the limit until a goal is found. This algorithm performs depth-first search up to a certain "depth limit", and it keeps increasing the depth limit after each iteration until the goal node is found. This Search algorithm combines the benefits of Breadth-first search's fast search and depth-first search's memory efficiency. The iterative search algorithm is useful uninformed search when search space is large, and depth of goal node is unknown.

Algorithm:

Step1: INPUT: START and GOAĮ states

Step2: ĮOCAĮ VARIABĮE: Found

Step3: Initialise d = 1 and FOUND = False

Step4: while (FOUND = False) do perform

DFS from start to depth d.

Step5: if goal state is obtained then FOUND = True

else discard the nodes generated in the search of depth d. Step6: d = d

+ 1

Step7: if FOUND = true, then return the depth. Step8:

Stop

Code :

```java
import java.util.*;

public class DFID_JAVA{
    LinkedList<Integer> adj[];
    public void graph(int v){
        adj = new LinkedList[v];
        for(int i=0; i<v; i++){
            adj[i] = new LinkedList();
        }
    }
    void addEdge(int v, int w){
        adj[v].add(w);
    }
    boolean DLS(int v,int target,int limit){
        if(v == target)
            return true;
        if(limit <= 0)
            return false;
        for(int i : adj[v]){
            System.out.print(i+" -> ");
            if(DLS(i,target,limit-1))
                return true;
        }
        return false;
    }
    boolean DFID(int src,int target,int max_depth){
        for(int i=0;i<max_depth;i++){
            System.out.println("\nDepth = "+i);
            if(DLS(src,target,i))
                return true;
            }
        return false;
    }
    public static void main(String[] args){
        DFID_JAVA g = new DFID_JAVA();
        g.graph(7);
        g.addEdge(0,1);
        g.addEdge(0,2);
        g.addEdge(1,3);
        g.addEdge(1,4);
        g.addEdge(2,4);
        g.addEdge(2,5);
        g.addEdge(2,6);
        g.addEdge(3,7);
        g.addEdge(3,8);
        g.addEdge(4,5);
        int src=0,target=5,max_depth=4;
        if(g.DFID(src,target,max_depth)){
            System.out.println("Target found");
        } else {
            System.out.println("Target not found");
        }
    }
}
```

Output :

Conclusion :

Hence DFID search algorithm was performed and optimal path was found