Name: Aditya.Rajesh.Wanwade

Roll no: 27

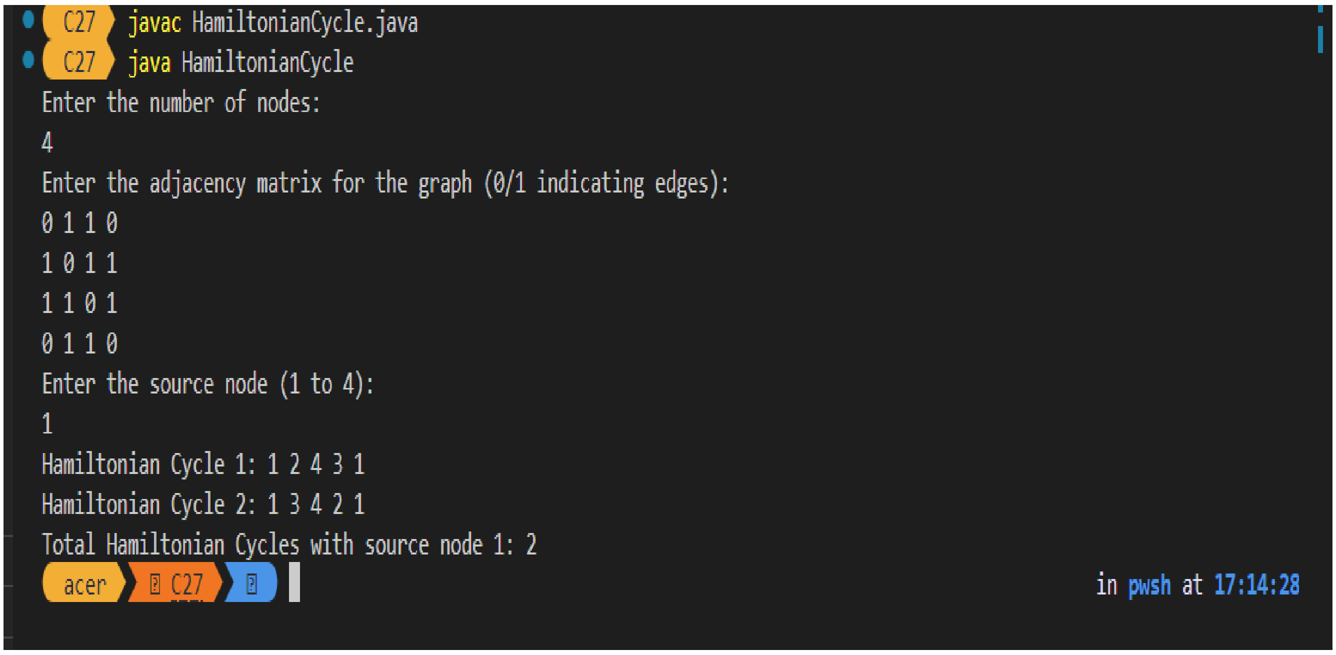
Branch: Cyber Security

DAA Practical no:7

Code of Hamiltonian Cycle:

```java
import java.util.*;

public class HamiltonianCycle{
    private static int N;
    private static int[][] graph;
    private static int[] path;
    private static boolean[] visited;
    private static Map<List<Integer>, Boolean> visitedPaths;
    private static int hamiltonianCycleCount = 0;

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number of nodes:");
        N = sc.nextInt();

        graph = new int[N][N];
        path = new int[N];
        visited = new boolean[N];
        visitedPaths = new HashMap<>();

        System.out.println("Enter the adjacency matrix for the graph (0/1
indicating edges):");
        for (int i = 0; i < N; i++) {
            for (int j = 0; j < N; j++) {
                graph[i][j] = sc.nextInt();
            }
        }

        System.out.println("Enter the source node (1 to " + N + "):");
        int sourceNode = sc.nextInt();

        findHamiltonianCycle(0, sourceNode);

        if (hamiltonianCycleCount > 0) {
            System.out.println("Total Hamiltonian Cycles with source node " +
sourceNode + ": " + hamiltonianCycleCount);
        } else {
            System.out.println("No Hamiltonian Cycle found with source node "
+ sourceNode);
        }

        sc.close();
    }

    private static void findHamiltonianCycle(int pos, int sourceNode) {
        if (pos == N) {
            if (graph[path[N - 1] - 1][path[0] - 1] == 1 && path[0] ==
sourceNode) {
                // Check if last node connects back to the source node to form
a cycle
                List<Integer> currentPath = new ArrayList<>();
                for (int i = 0; i < N; i++) {
                    currentPath.add(path[i]);
                }
                if (!visitedPaths.containsKey(currentPath)) {
                    hamiltonianCycleCount++;
                    visitedPaths.put(currentPath, true);
                    System.out.print("Hamiltonian Cycle " +
hamiltonianCycleCount + ": ");
                    for (int node : path) {
                        System.out.print(node + " ");
                    }
                    System.out.println(path[0]);
                }
            }
            return;
        }

        for (int v = 1; v <= N; v++) {
            if (isSafe(pos, v)) {
                path[pos] = v;
                visited[v - 1] = true;
                findHamiltonianCycle(pos + 1, sourceNode);
                visited[v - 1] = false;
                path[pos] = 0;
            }
        }
    }

    private static boolean isSafe(int pos, int v) {
        if (pos == 0) {
            return true;
        }
        if (!visited[v - 1] && graph[path[pos - 1] - 1][v - 1] == 1) {
            return true;
        }
        return false;
    }
}
```

Output:



```
C27  javac HamiltonianCycle.java
C27  java HamiltonianCycle
Enter the number of nodes:
4
Enter the adjacency matrix for the graph (0/1 indicating edges):
0 1 1 0
1 0 1 1
1 1 0 1
0 1 1 0
Enter the source node (1 to 4):
1
Hamiltonian Cycle 1: 1 2 4 3 1
Hamiltonian Cycle 2: 1 3 4 2 1
Total Hamiltonian Cycles with source node 1: 2
acer  C27                                              in pwsh at 17:14:28
```