PRACTICAL 7

NAME:NEER AWASTHI

CLASS:A4 B2 20

**AIM:** Implement Hamiltonian Cycle using Backtracking.

### **PROBLEM STATEMENT:**

The Smart City Transportation Department is designing a night-patrolroute for security vehicles.  
 Each area is represented as a vertex in a graph, and a road between two areas is represented as an edge.

The goal is to find a route starting from the headquarters, visiting every area exactly once, and returning to the starting point — forming a Hamiltonian Cycle.

If such a route is not possible, display a suitable message.

**Source Code:**

**# Hamiltonian Cycle using Backtracking**

**# Smart City Night Patrol Route Problem**

**def is\_valid(v, pos, path, graph):**

**"""Check if vertex v can be added at position pos"""**

**if graph[path[pos - 1]][v] == 0:**

**return False**

**if v in path:**

**return False**

**return True**

**def hamiltonian\_cycle\_util(graph, path, pos):**

**"""Recursive utility function"""**

**n = len(graph)**

**if pos == n:**

**return graph[path[pos - 1]][path[0]] == 1**

**for v in range(1, n):**

**if is\_valid(v, pos, path, graph):**

**path[pos] = v**

**if hamiltonian\_cycle\_util(graph, path, pos + 1):**

**return True**

**path[pos] = -1**

**return False**

**def hamiltonian\_cycle(graph, vertices):**

**"""Main driver function"""**

**n = len(graph)**

**path = [-1] \* n**

**path[0] = 0**

**if not hamiltonian\_cycle\_util(graph, path, 1):**

**print("No Hamiltonian Cycle exists for the given city layout.\n")**

**else:**

**cycle = [vertices[v] for v in path] + [vertices[path[0]]]**

**print("Hamiltonian Cycle (Night Patrol Route) Found:")**

**print(" → ".join(cycle))**

**print()**

**# -------------------------------**

**# MAIN PROGRAM (Test Cases)**

**# -------------------------------**

**# CASE 1: AREAS (A, B, C, D, E)**

**vertices1 = ['A', 'B', 'C', 'D', 'E']**

**graph1 = [**

**[0, 1, 1, 0, 1],**

**[1, 0, 1, 1, 0],**

**[1, 1, 0, 1, 0],**

**[0, 1, 1, 0, 1],**

**[1, 0, 0, 1, 0]**

**]**

**print("----- CASE 1: AREAS (A, B, C, D, E) -----")**

**hamiltonian\_cycle(graph1, vertices1)**

**# CASE 2: PLACES (T, M, S, H, C)**

**vertices2 = ['T', 'M', 'S', 'H', 'C']**

**graph2 = [**

**[0, 1, 1, 0, 1],**

**[1, 0, 1, 1, 0],**

**[1, 1, 0, 1, 1],**

**[0, 1, 1, 0, 1],**

**[1, 0, 1, 1, 0]**

**]**

**print("----- CASE 2: PLACES (T, M, S, H, C) -----")**

**hamiltonian\_cycle(graph2, vertices2)**

**# CASE 3: NO HAMILTONIAN CYCLE (Negative Test Case)**

**vertices3 = ['P', 'Q', 'R', 'S']**

**graph3 = [**

**[0, 1, 0, 0],**

**[1, 0, 1, 0],**

**[0, 1, 0, 0],**

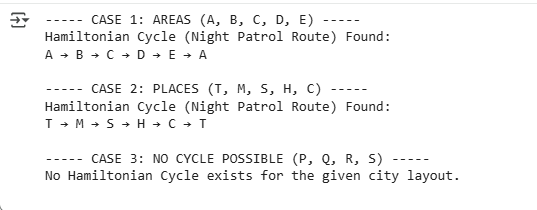
**[0, 0, 0, 0]**

**]**

**print("----- CASE 3: NO CYCLE POSSIBLE (P, Q, R, S) -----")**

**hamiltonian\_cycle(graph3, vertices3)**

**Output:**

****