PRACTICAL NO 7

NAME:Tanmay Kulkarni
CLASS:A3 B4 60

Aim: Implement Hamiltonian Cycle using Backtracking.

Problem Statement:

The Smart City Transportation Department is designing a night-patrol route for security vehicles. Each area of the city 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 that starts from the main headquarters (Area A), visits each area exactly once, and returns back to the headquarters — forming a Hamiltonian Cycle.
If such a route is not possible, display a suitable message.

1) Adjacency Matrix
A B C D E
A 0 1 1 0 1
B 1 0 1 1 0
C 1 1 0 1 0
D 0 1 1 0 1
E 0 1 1 0 1

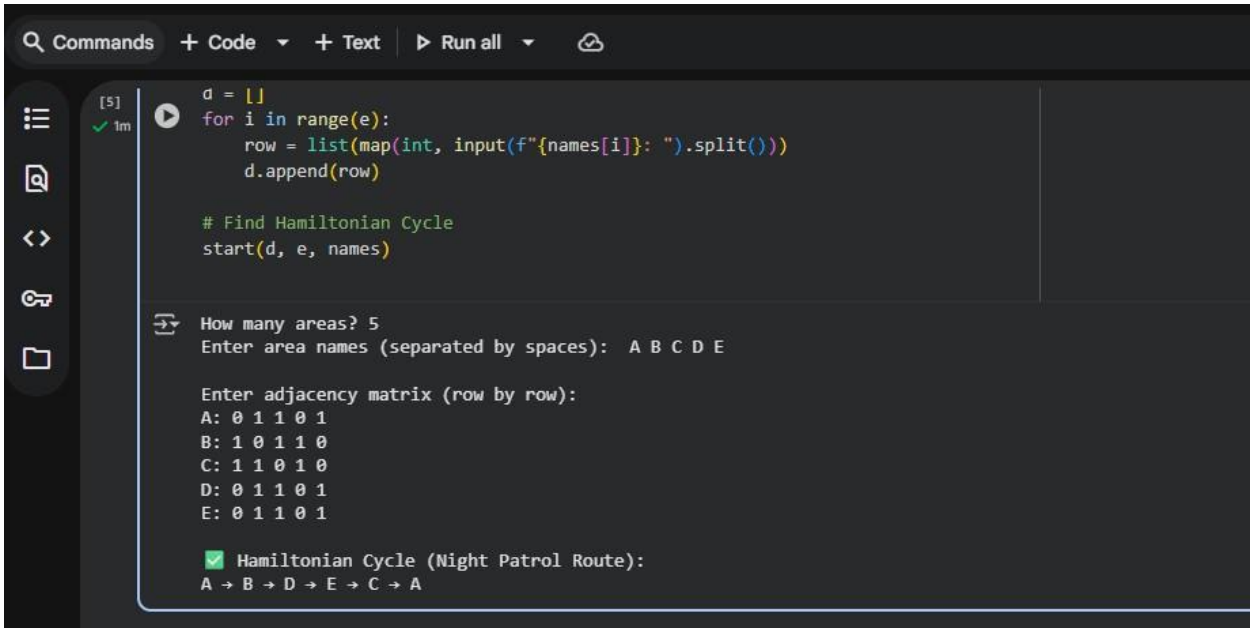
2) Adjacency Matrix
T M S H C
T 0 1 1 0 1
M 1 0 1 1 0
S 1 1 0 1 1
H 0 1 1 0 1
C 1 0 1 1 0

CODE:

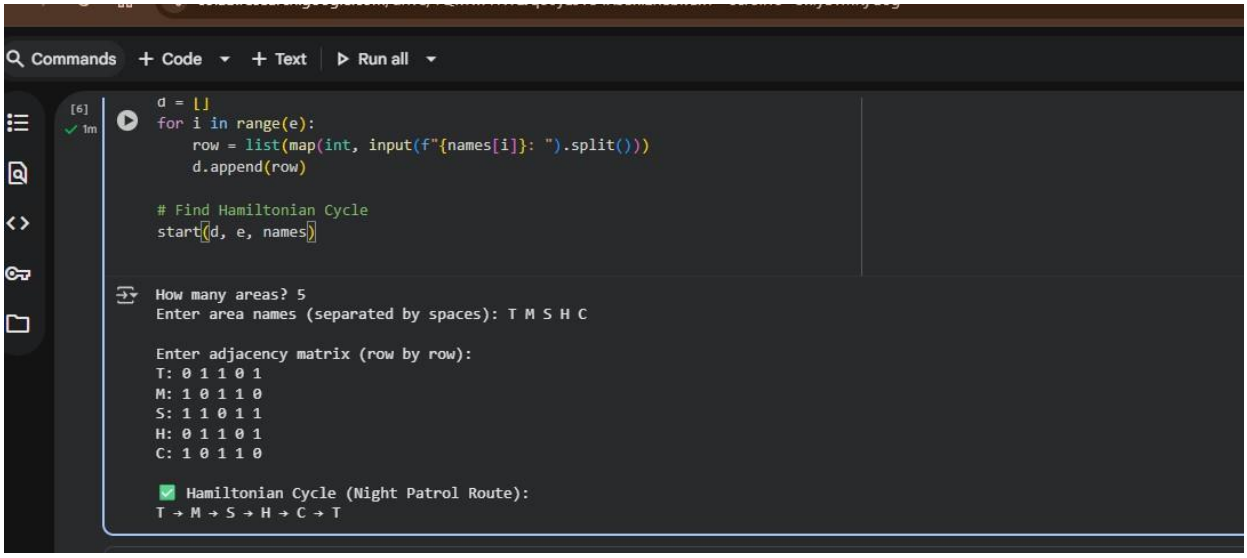
```
def check(b, a, c, d):
    if d[c[a - 1]][b] == 0:
        return False
    if b in c:
        return False
```

```
    return True

def go(a, c, d, e):
    if a == e:
        return d[c[a - 1]][c[0]] == 1

    for b in range(1, e):  # start from 1 since 0 is fixed (start area)
        if check(b, a, c, d):
            c[a] = b
            if go(a + 1, c, d, e):
                return True
            c[a] = -1
    return False

def start(d, e, names):
    c = [-1] * e
    c[0] = 0  # start from the first area

    if go(1, c, d, e):
        print("\n Hamiltonian Cycle (Night Patrol Route):")
        for x in c:
            print(names[x], end=" → ")
        print(names[c[0]])
    else:
        print("\n No Hamiltonian Cycle found for the given city network.")

# ---- MAIN PROGRAM ----
e = int(input("How many areas? "))

# Input area names (e.g., A B C D E)
names = input("Enter area names (separated by spaces): ").split()

# Input adjacency matrix
print("\nEnter adjacency matrix (row by row):")
d = []
for i in range(e):
    row = list(map(int, input(f"{names[i]}: ").split()))
    d.append(row)

# Find Hamiltonian Cycle
start(d, e, names)
```

OUTPUT:


Commands
+ Code
+ Text
Run all
[5]
1m
d = []
for i in range(e):
row = list(map(int, input(f"{names[i]}: ").split()))
d.append(row)
# Find Hamiltonian Cycle
start(d, e, names)
How many areas? 5
Enter area names (separated by spaces): A B C D E
Enter adjacency matrix (row by row):
A: 0 1 1 0 1
B: 1 0 1 1 0
C: 1 1 0 1 0
D: 0 1 1 0 1
E: 0 1 1 0 1
Hamiltonian Cycle (Night Patrol Route):
A → B → D → E → C → A

2 output:


Commands
+ Code
+ Text
Run all
[6]
1m
d = []
for i in range(e):
row = list(map(int, input(f"{names[i]}: ").split()))
d.append(row)
# Find Hamiltonian Cycle
start(d, e, names)
How many areas? 5
Enter area names (separated by spaces): T M S H C
Enter adjacency matrix (row by row):
T: 0 1 1 0 1
M: 1 0 1 1 0
S: 1 1 0 1 1
H: 0 1 1 0 1
C: 1 0 1 1 0
Hamiltonian Cycle (Night Patrol Route):
T → M → S → H → C → T