**Name : Prashant Raghuwanshi**

**Roll No : A4\_B2\_34**

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

**Code :**

def is\_safe(v, graph, path, pos):

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

return False

return True

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

if pos == len(graph):

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

for v in range(1, len(graph)):

if is\_safe(v, graph, path, pos):

path[pos] = v

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

return True

path[pos] = -1

return False

def hamiltonian\_cycle(graph, names):

path = [-1] \* len(graph)

path[0] = 0

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

print("No Hamiltonian Cycle exists.")

return

print("Hamiltonian Cycle:", " -> ".join(names[i] for i in path), "->", names[path[0]])

# Example

names = ['A', 'B', 'C', 'D', 'E']

graph = [

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

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

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

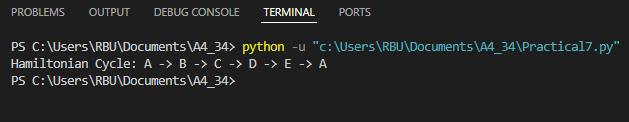
[0, 1, 1, 0, 1],

[1, 0, 0, 1, 0]

]

hamiltonian\_cycle(graph, names)

**Output :**

****

### Competitive Coding Que. Hamiltonian Path

### Code :

### 

### Output :

### 