120. Hamilton Cycle Problem

AIM: To find the hamilton cycle by using backtacking

PROGRAM:

```
class Graph:
    def __init__(self, vertices):
        self.V = vertices
        self.graph = [[0 for column in range(vertices)]
                      for row in range(vertices)]
    def is_safe(self, v, pos, path):
        if self.graph[path[pos - 1]][v] == 0:
            return False
        if v in path:
            return False
        return True
    def hamiltonian_cycle_util(self, path, pos):
        if pos == self.V:
            if self.graph[path[pos - 1]][path[0]] == 1:
                return True
            else:
                return False
        for v in range(1, self.V):
            if self.is_safe(v, pos, path):
                path[pos] = v
                if self.hamiltonian_cycle_util(path, pos + 1) == True:
                    return True
                path[pos] = -1
        return False
    def hamiltonian_cycle(self):
        path = [-1] * self.V
        path[0] = 0
        if self.hamiltonian_cycle_util(path, 1) == False:
```

```
            print("Solution does not exist\n")
            return False
        self.print_solution(path)
        return True
    def print_solution(self, path):
        print("Solution exists: Following is one Hamiltonian Cycle")
        for vertex in path:
            print(vertex, end=" ")
        print(path[0], "\n")
g1 = Graph(5)
g1.graph = [[0, 1, 0, 1, 0],
            [1, 0, 1, 1, 1],
            [0, 1, 0, 0, 1],
            [1, 1, 0, 0, 1],
            [0, 1, 1, 1, 0]]
g1.hamiltonian_cycle()
```

OUTPUT:

```
Solution exists: Following is one Hamiltonian
    Cycle
0 1 2 4 3 0
```

TIME COMPLEXITY: O(V!)