Hamiltonian Cycle

A hamiltonian path in a graph is a path that includes each vertex exactly once. A hamiltonian cycle is a cycle that includes each vertex exactly once. A hamiltonian graph is a graph that has a hamiltonian cycle

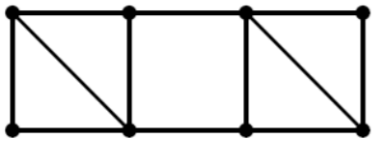


Figure : A graph with Hamitonian Cycle (Hamitonian Graph)

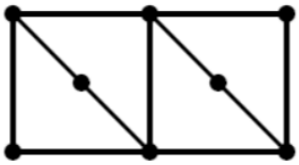


Figure : A graph with no Hamiltonian Cycle

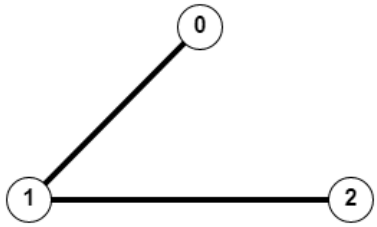
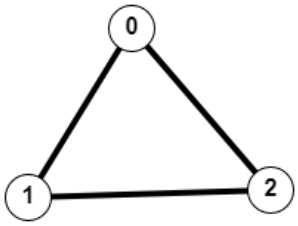


Figure : A graph with Hamiltonian Path but no Hamiltonian Cycle

Note : If a graph have hamiltonian cycle it also means, it definitely have a hamiltonian path but opposite might not be true.



{0, 1, 2, 0}
{1, 2, 0, 1}
{2, 0, 1, 2}

{0, 2, 1, 0}

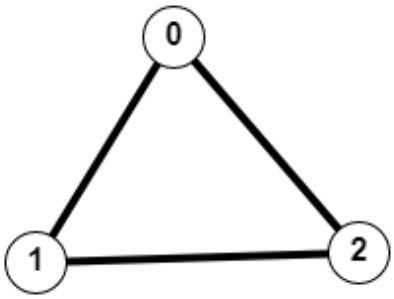
Note :

- A graph can have more than one Hamiltonian Cycle
- In an undirected graph, If there is a Hamiltonian Cycle, then the path can be started from any point(vertex) in the cycle.

Adjacency Matrix

If vertices of a Graph G are ordered e.g. v_1, v_2, v_3, ……, v_n , the *adjacency matrix* A = (a_{ij}) of graph G is n*n matrix is defined as

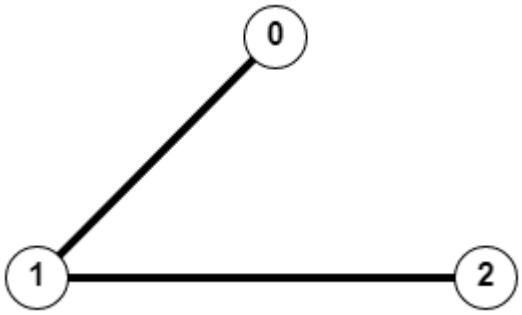
a_{ij} = 1 if there is an edge between v_i and v_j (adjacent nodes)
0 otherwise




```
int graph1[][] = {
                 {0, 1, 1},
                 {1, 0, 1},
                 {1, 1, 0}
           };
```

Key Points :

- Different orderings of the vertex set may result in different adjacency matrices for the same graph.
- In an undirected graph the number of 1s in adjacency matrix is twice the number of edges




```
int graph2[][] = {
                {0, 1, 0},
                {1, 0, 1},
                {0, 1, 0}
           };
```

Approach

We will first add vertex 0 (v_0) to hamiltonian path, and then try to add next vertices to hamiltonian path, while trying to add next vertices to hamiltonian path we will start from vertex 1 (v_1).

We will only add a vertex to hamiltonian path If that vertex satisfies following conditions

1. If the next vertex we are trying to add to hamiltonian path is connected (have an edge) to the last vertex in hamiltonian path
2. If the next vertex we are trying to add to hamiltonian path is not already added to hamiltonian path

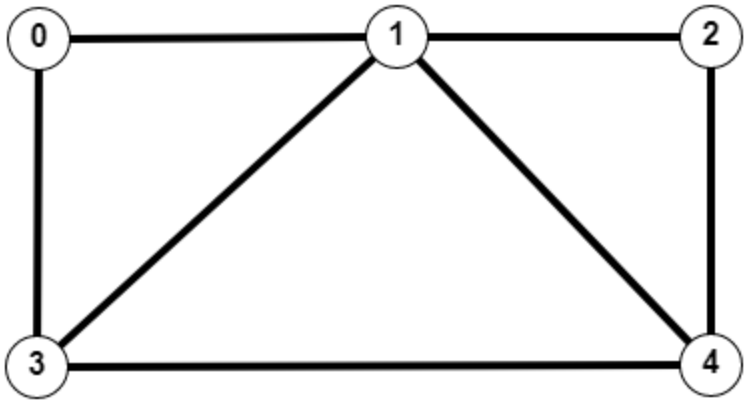
Finally when the hamiltonian path contains the vertices equal to total number of vertices in the graph, we will check If there is an edge from the last vertex in hamiltonian path to vertex 0 (v_0). If there is an edge it means, the graph have a hamiltonian cycle.

```
class App {

   public static int adjacencyMatrix[][] = {
               {0, 1, 1},
                {1, 0, 1},
                {1, 1, 0}
           };

    public static void main(String args[]) {
      HamiltonianCycle.findHamiltonianCycle(adjacencyMatrix);
    }
}
```

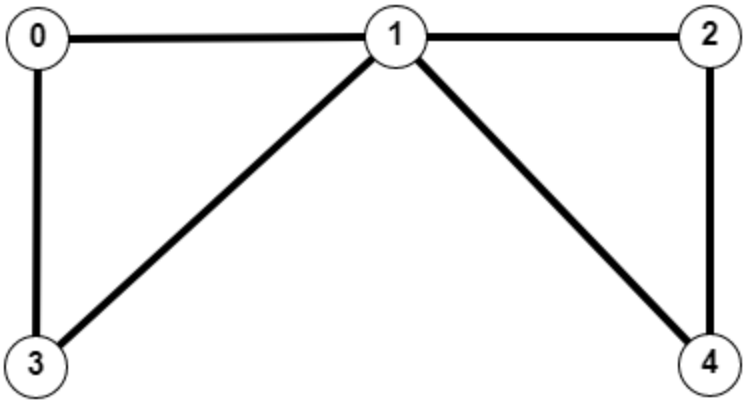
```
public class HamiltonianCycle {

       private static int numOfVertices;
       private static int hamiltonianPath[];
       private static int[][] adjacencyMatrix;

      . . .

}
```

```
public static void findHamiltonianCycle(int graphAdjacencyMatrix[][]) {
              adjacencyMatrix = graphAdjacencyMatrix;
              numOfVertices = adjacencyMatrix[0].length;
              hamiltonianPath = new int[numOfVertices];

              hamiltonianPath[0] = 0;

              if (findFeasibleSolution(1)) {
                    printHamiltonianCycle();

              } else{
               System.out.println("There is no Hamiltonian Cycle in the graph...");
              }
}
```

```
public static boolean findFeasibleSolution(int nextVertex) {

 if (nextVertex == numOfVertices) {
   if (adjacencyMatrix[hamiltonianPath[nextVertex - 1]][hamiltonianPath[0]] == 1) {
            return true;
    } else {
            return false;
    }
 }

for (int vertexIndex = 1; vertexIndex < numOfVertices; ++vertexIndex) {
      if (isFeasible(vertexIndex, nextVertex)) {

         hamiltonianPath[nextVertex] = vertexIndex;
         if (findFeasibleSolution(nextVertex + 1)) {
            return true;
         }
          hamiltonianPath[nextVertex] = 0;
       }

}

 return false;
}




public static boolean isFeasible(int vertex, int nextVertex) {

    if (adjacencyMatrix[hamiltonianPath[nextVertex - 1]][vertex] == 0) {
            return false;
    }

    for (int i = 0; i < nextVertex; ++i) {
      if (hamiltonianPath[i] == vertex) {
            return false;
       }
    }

   return true;
}
```




```
int graph3[][] = {
            {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},
        };
```




```
int graph4[][] = {
            {0, 1, 0, 1, 0},
            {1, 0, 1, 1, 1},
            {0, 1, 0, 0, 1},
            {1, 1, 0, 0, 0},
            {0, 1, 1, 0, 0},
        };
```