**MINI PROJECT**

**PROBLEM STATEMENT**

Given a set of cities and distances between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and returns to the starting point.

**THEORY**

 It is an NP-hard problem in combinatorial optimization, important in theoretical computer science and operations research. The travelling purchaser problem and the vehicle routing problem are both generalizations of TSP.

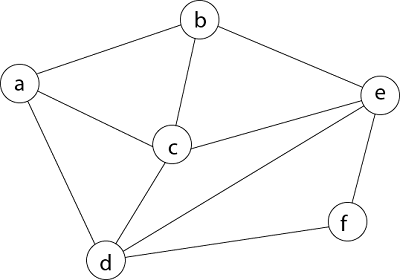
In the theory of computational complexity, the decision version of the TSP (where given a length *L*, the task is to decide whether the graph has a tour of at most *L*) belongs to the class of NP-complete problems. Thus, it is possible that the worst-case running time for any algorithm for the TSP increases superpolynomially (but no more than exponentially) with the number of cities.

Note the difference between Hamiltonian Cycle and TSP. The Hamiltonian cycle problem is to find if there exists a tour that visits every city exactly once. Here we know that Hamiltonian tour exists (because the graph is complete) and in fact, many such tours exist, the problem is to find a minimum weight Hamiltonian cycle.

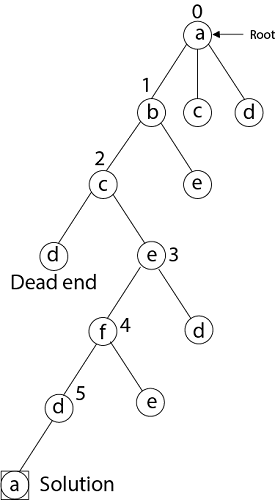
**Hamiltonian Cycle:**

A Hamiltonian cycle, also called a Hamiltonian circuit, Hamilton cycle, or Hamilton circuit, is a graph cycle (i.e., closed loop) through a graph that visits each node exactly once. A graph possessing a Hamiltonian cycle is said to be a Hamiltonian graph.

**Example:**



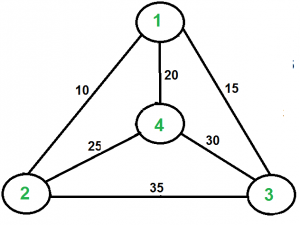
We can start from any node from the given graph but for this case we are starting from the vertex a.



Here we have generated one Hamiltonian circuit, but another Hamiltonian circuit can also be obtained by considering another vertex.

**TSP Example:**

For example, consider the graph shown in the figure on the right side. A TSP tour in the graph is 1-2-4-3-1. The cost of the tour is 10+25+30+15 which is 80.



**ALGORITHM**

1. Consider city 1 as the starting and ending point. Since the route is cyclic, we can consider any point as a starting point.
2. Generate all (n-1)! Permutation with minimum cost.
3. Return the permutation with minimum cost.

**HARDWARE REQUIREMENTS**

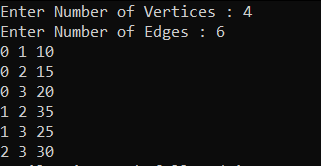
* + 1. Local System

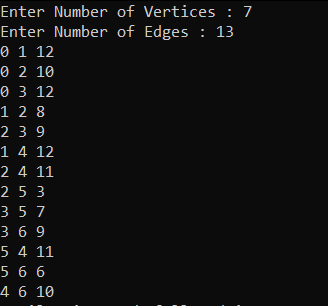
1. i3/i5/i7 processor
2. 4GB RAM

**SOFTWARE REQUIREMENTS**

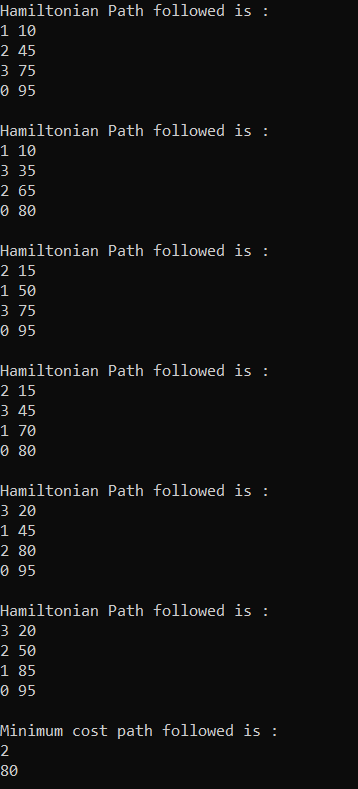
1. Jupyter - Notebook
2. Environment (VS Code/Code Blocks)

**INPUT**





**OUTPUT**



**APPLICATIONS**

1. Railway can analyse the best route available for a long distance train with many stops. They can come up with the best route which has more travellers and takes minimal time and distance between first and last destination.
2. Airlines flights consist of multiple legs. A flight leg is a unit of flight consisting of one take-off and one landing. Each flight leg is unique with its source, destination, arrival and departure time.
3. Control of the stacker cranes in a Herlitz warehouse.
4. Production of ICs and PCBs.