**Practice Problems on Graphs**

**Problem 1:**

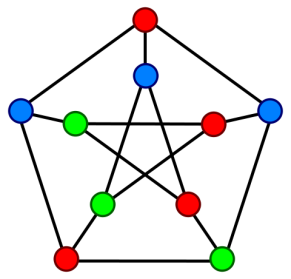
The ***square*** of a directed graph G(V, E) is the graph G2 (V, E2) such that (u, v) Є E2 if and only G contains a path with at most two edges between u and v. Write an efficient function for computing G2 from G for both the adjacency list and adjacency-matrix representations of G. Analyze the running times of your code.

**Problem 2:**

Let Nsum[v] be the sum of degrees of all the neighbors of a vertex v in an undirected graph G(V, E). Devise an O(|V|+|E|) function to compute the value of Nsum for each vertex in G.

**Problem 3:**

A graph G(V, E) is k-colorable if k different colors are enough to color the vertices of the graph such that no two adjacent vertices share a common color. Below is an example of 3-colorable graph.



Develop an O(|V| + |E|) time function which determines whether a given undirected graph G is 2-colorable or not.

**Problem 4**

Implement a function HamiltonianCycle() that takes an unweighted directed graph in form of adjacency list as parameter and returns true if there exists a cycle in the graph that visits each vertex of the graph exactly once (Hamiltonian cycle) and false otherwise. It should also return the sequence of vertices that must be traversed to form a hamiltonian cycle.

**Problem 5**

Implement a function MinHamiltonianCycle() that takes a weighted directed graph in form of adjacency list as parameter and returns true if there exists a cycle in the graph that visits each vertex of the graph exactly once (Hamiltonian cycle) and false otherwise. It should also find the Hamiltonian cycle of minimum weight i.e sum of edge weights of all the edges on the cycle is minimum. It should also return the sequence of vertices that must be traversed to form a minimum weight hamiltonian cycle.