ABHINAV RANJAN
RA1911003010003
CSE A1 SECTION
SRMIST , KTR

AI LAB EXP 2 - GRAPH COLOURING

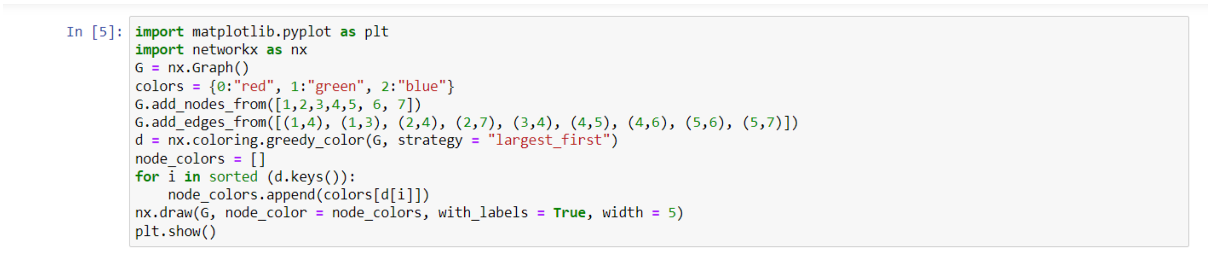
PROBLEM STATEMENT
Graph colouring problem is to assign colours to certain elements of a graph subject to certain constraints.
Vertex colouring is the most common graph colouring problem. The problem is, given m colours, find a way of colouring the vertices of a graph such that no two adjacent vertices are colored using same colour

TOOLS USED - python3 , Jupyter notebook

SOURCE CODE :

```
import matplotlib.pyplot as plt
import networkx as nx
G = nx.Graph()
colors = {0:"red", 1:"green", 2:"blue"}
G.add_nodes_from([1,2,3,4,5, 6, 7])
G.add_edges_from([(1,4), (1,3), (2,4), (2,7), (3,4), (4,5), (4,6), (5,6),
(5,7)])
d = nx.coloring.greedy_color(G, strategy = "largest_first")
node_colors = []
for i in sorted (d.keys()):
   node_colors.append(colors[d[i]])
nx.draw(G, node_color = node_colors, with_labels = True, width = 5)
plt.show()
```

SCREENSHOTS :



```
In [5]: import matplotlib.pyplot as plt
        import networkx as nx
        G = nx.Graph()
        colors = {0:"red", 1:"green", 2:"blue"}
        G.add_nodes_from([1,2,3,4,5, 6, 7])
        G.add_edges_from([(1,4), (1,3), (2,4), (2,7), (3,4), (4,5), (4,6), (5,6), (5,7)])
        d = nx.coloring.greedy_color(G, strategy = "largest_first")
        node_colors = []
        for i in sorted (d.keys()):
            node_colors.append(colors[d[i]])
        nx.draw(G, node_color = node_colors, with_labels = True, width = 5)
        plt.show()
```