

AI LAB MANUAL

Exp 2: Agent Problems

Name:	SHAMUNESH P
Reg.no:	RA1911030010122
Problem:	Graph Coloring Problem
Date:	14-01-22

Code: (Python)

```
class Graph:
```

```
    def __init__(self,edges,n):
```

```
        self.adjList = [[] for _ in range(n)]
```

```
        for (src,dest) in edges:
```

```
            self.adjList[src].append(dest)
```

```
            self.adjList[dest].append(src)
```

```
def colorGraph(graph, n):
```

```
    result = {}
```

```
    for u in range(n):
```

```
        assigned = set([result.get(i) for i in graph.adjList[u] if i in result])
```

```
        color = 1
```

```
        for c in assigned:
```

```
            if color != c:
```

```
                break
```

```
            color = color +1
```

```
        result[u] = color
```

```
    for v in range(n):
```

```
        print(f'Color assigned to vertex {v} is {colors[result[v]]}')
```

```
if __name__ == '__main__':
```

```
colors = ['', 'BLUE', 'GREEN', 'RED', 'YELLOW', 'ORANGE', 'PINK', 'BLACK', 'BROWN', 'WHITE',  
, 'PURPLE', 'VIOLET']
```

```
edges = [(0,1), (0,4), (0,5), (4,5), (1,4), (1,3), (2,3), (2,4)]
```

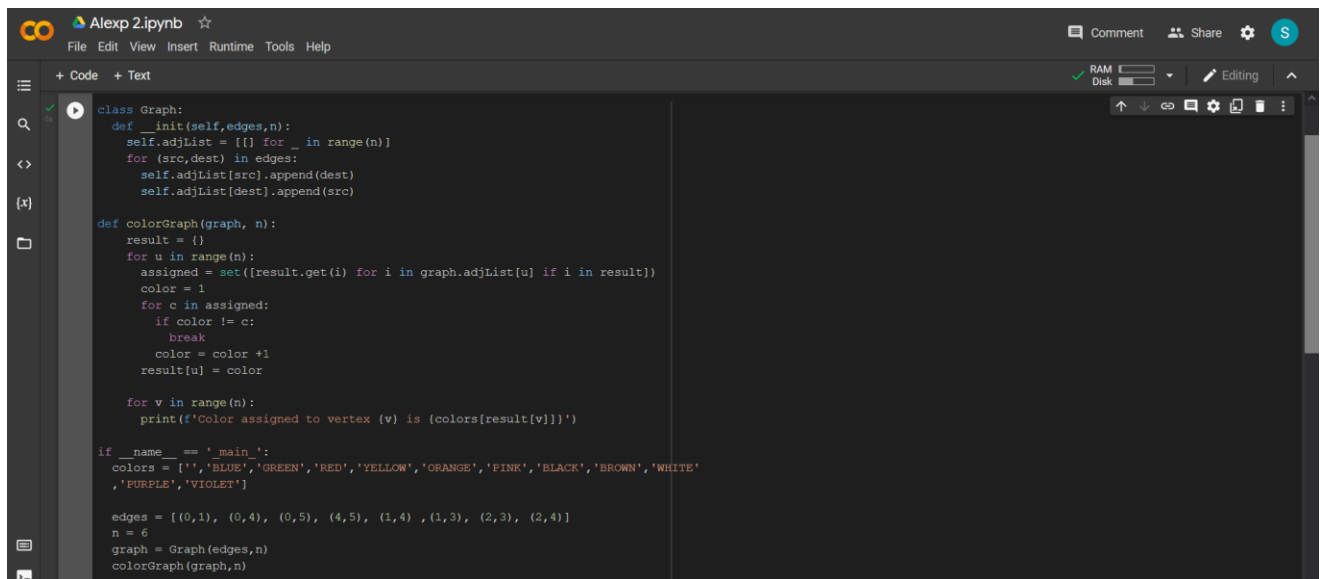
```
n = 6
```

```
graph = Graph(edges,n)
```

```
colorGraph(graph,n)
```

Implementation:

Screenshots

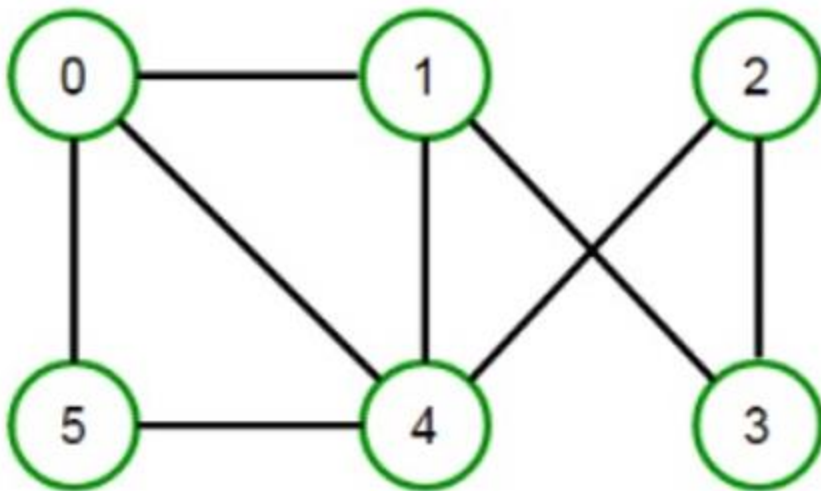


The screenshot shows a Jupyter Notebook interface with a dark theme. The notebook is titled 'Alexp 2.ipynb'. The code is written in Python and defines a 'Graph' class and a 'colorGraph' function. The 'Graph' class has an '__init__' method that takes 'self', 'edges', and 'n' as arguments. It initializes 'self.adjList' as a list of empty lists for each vertex from 0 to n-1. It then iterates over the 'edges' list and appends the source and destination vertices to their respective adjacency lists. The 'colorGraph' function takes 'graph' and 'n' as arguments. It initializes a 'result' list of size 'n' with empty sets. It then iterates over each vertex 'u' from 0 to n-1. For each 'u', it creates a set 'assigned' containing the colors of all vertices 'i' in the adjacency list of 'u' that are already in the 'result' list. It then increments 'color' starting from 1 until it finds a color not in 'assigned'. It assigns this color to 'result[u]' and prints the color assigned to vertex 'u'. Finally, it iterates over all vertices 'v' from 0 to n-1 and prints the color assigned to each vertex. The main block of the notebook defines the 'colors' list, the 'edges' list, sets 'n' to 6, creates a 'Graph' object, and calls the 'colorGraph' function.

```
class Graph:  
    def __init__(self, edges, n):  
        self.adjList = [[] for _ in range(n)]  
        for (src, dest) in edges:  
            self.adjList[src].append(dest)  
            self.adjList[dest].append(src)  
  
    def colorGraph(self, n):  
        result = {}  
        for u in range(n):  
            assigned = set([result.get(i) for i in self.adjList[u] if i in result])  
            color = 1  
            for c in assigned:  
                if color != c:  
                    break  
            color = color + 1  
            result[u] = color  
  
        for v in range(n):  
            print(f'Color assigned to vertex {v} is {colors[result[v]]}')  
  
if __name__ == '__main__':  
    colors = ['', 'BLUE', 'GREEN', 'RED', 'YELLOW', 'ORANGE', 'PINK', 'BLACK', 'BROWN', 'WHITE',  
            , 'PURPLE', 'VIOLET']  
  
    edges = [(0,1), (0,4), (0,5), (4,5), (1,4), (1,3), (2,3), (2,4)]  
    n = 6  
    graph = Graph(edges, n)  
    colorGraph(graph, n)
```

Observation:

Graph Before Vertex Coloring:



Output:

```

class Graph:
    def __init__(self, edges, n):
        self.adjlist = [[] for _ in range(n)]
        for (src, dest) in edges:
            self.adjlist[src].append(dest)
            self.adjlist[dest].append(src)

    def colorGraph(self, n):
        result = {}
        for u in range(n):
            assigned = not([result.get(i) for i in graph.adjlist[u] if i in result])
            color = 1
            for c in assigned:
                if color != c:
                    break
            color = color + 1
            result[u] = color
        for v in range(n):
            print(f'Color assigned to vertex {v} is {color[result[v]]}')

if __name__ == '__main__':
    colors = ['BLUE', 'GREEN', 'RED', 'YELLOW', 'ORANGE', 'PINK',
             'BLACK', 'BROWN', 'WHITE', 'PURPLE', 'GREY']
    edges = [(0, 1), (0, 4), (0, 5), (1, 4), (1, 2), (2, 3), (2, 4), (3, 4), (4, 5)]
    n = 6
    graph = Graph(edges, n)
    colorGraph(graph, n)
  
```

Color assigned to vertex 0 is BLUE
 Color assigned to vertex 1 is GREEN
 Color assigned to vertex 2 is BLUE
 Color assigned to vertex 3 is RED
 Color assigned to vertex 4 is RED
 Color assigned to vertex 5 is GREEN

```

Color assigned to vertex 0 is BLUE
Color assigned to vertex 1 is GREEN
Color assigned to vertex 2 is BLUE
Color assigned to vertex 3 is RED
Color assigned to vertex 4 is RED
Color assigned to vertex 5 is GREEN
  
```

Graph After Vertex Coloring:

