

PRACTICAL NUMBER – 08

NAME – RANVEER M. BHORTEKAR

SECTION/BATCH – A4/B1

ROLL NO. – 06

SUBJECT – DAA

Aim - Implement Graph Colouring algorithm use Graph colouring concept.

Problem Statement - A GSM is a cellular network with its entire geographical range divided into hexadecimal cells. Each cell has a communication tower which connects with mobile phones within cell. Assume this GSM network operates in different frequency ranges. Allot frequencies to each cell such that no adjacent cells have same frequency range. Consider an undirected graph $G = (V, E)$ shown in fig. Find the colour assigned to each node using Backtracking method. Input is the adjacency matrix of a graph $G(V, E)$, where V is the number of Vertices and E is the number of edges.

CODE IN TEXT FORMAT –

```
def next_value(k,n,m,G,x):
    while True:
        x[k]=(x[k]+1)%(m+1)
        if x[k]==0:return
        for j in range(n):
            if G[k][j]!=0 and x[k]==x[j]:
                break
        else:return

def m_coloring(k,n,m,G,x):
    while True:
```

```
next_value(k,n,m,G,x)
if x[k]==0:return
if k==n-1:print(x)
else:m_coloring(k+1,n,m,G,x)

G=[[0,1,1,1,0],
[1,0,1,0,1],
[1,1,0,1,1],
[1,0,1,0,1],
[0,1,1,1,0]]

n=len(G)

m=3

x=[0]*n

m_coloring(0,n,m,G,x)
```

CODE SCREENSHOT –

The screenshot shows a Google Colab notebook titled "DAA PRAC 08.ipynb". The code in cell [1] defines two functions: `next_value` and `m_coloring`. The `next_value` function takes parameters k, n, m, G, and x. It iterates through a list x, calculating $x[k] = (x[k]+1) \% (m+1)$. If $x[k] \equiv 0 \pmod{m+1}$, it returns. Otherwise, it checks if $G[k][j] \neq 0$ and $x[k] \equiv x[j] \pmod{m+1}$. If true, it breaks the loop; otherwise, it continues. The `m_coloring` function takes parameters k, n, m, G, and x. It calls `next_value` until $x[k] \equiv 0 \pmod{m+1}$. If $k \equiv n-1 \pmod{n}$, it prints x. Otherwise, it calls itself with $k+1$. A global variable G is defined as a 6x6 matrix of lists. The code then calculates the length of G, initializes x as a list of zeros of length n, and calls `m_coloring` with index 0.

```
def next_value(k,n,m,G,x):
    while True:
        x[k]=(x[k]+1)% (m+1)
        if x[k]==0: return
        for j in range(n):
            if G[k][j]!=0 and x[k]==x[j]:
                break
        else: return

def m_coloring(k,n,m,G,x):
    while True:
        next_value(k,n,m,G,x)
        if x[k]==0: return
        if k==n-1: print(x)
        else: m_coloring(k+1,n,m,G,x)

G=[[0,1,1,1,0],
   [1,0,1,0,1],
   [1,1,0,1,1],
   [1,0,1,0,1],
   [0,1,1,1,0]]
n=len(G)
m=3
x=[0]*n
m_coloring(0,n,m,G,x)
```

