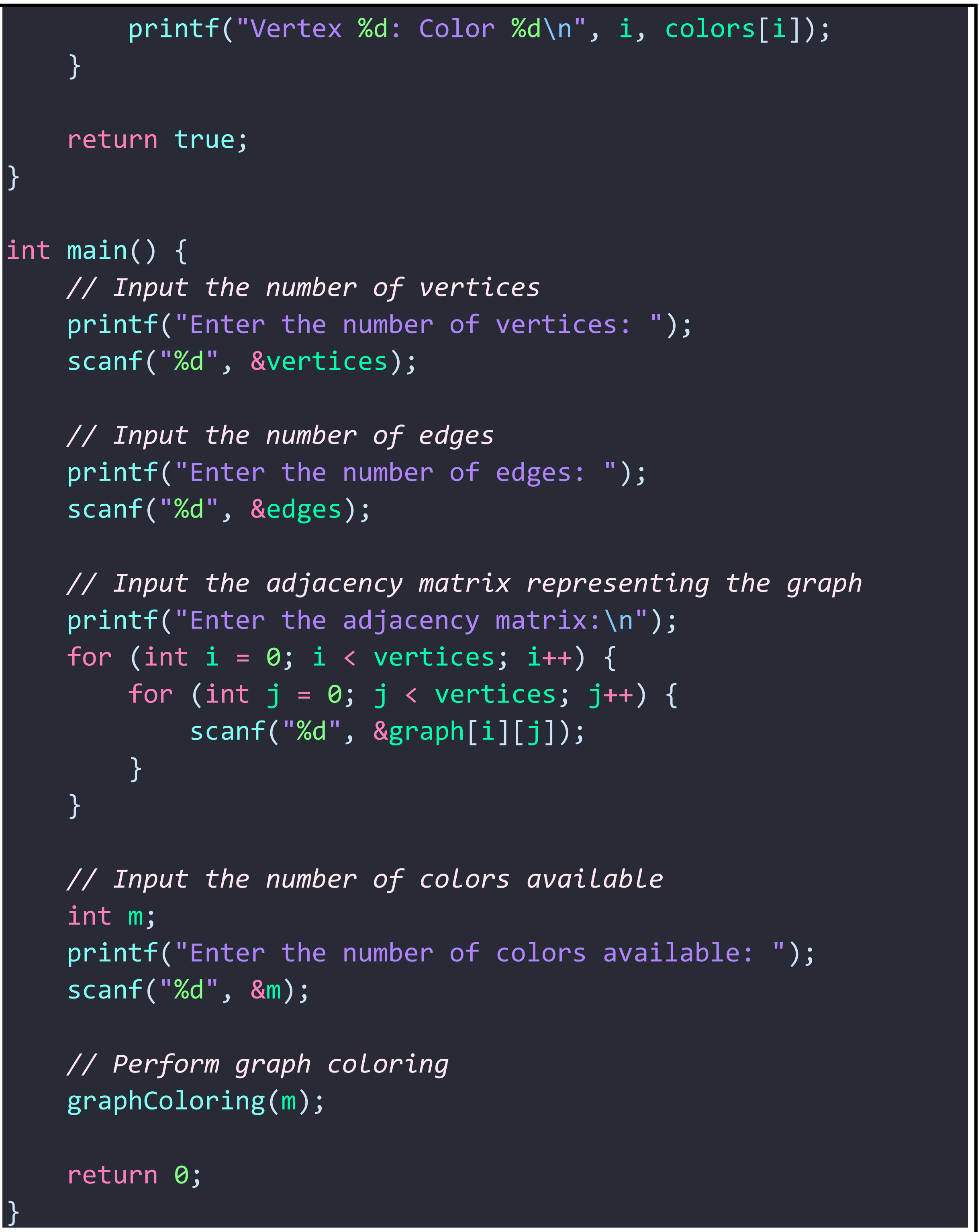
BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
(Empowered Autonomous Institute Affiliated to Mumbai University)
Department Of Computer Engineering

Name	Manish Shashikant Jadhav
UID	2023301005
Subject	Design and Analysis of Algorithms (DAA)
Experiment No.	8
Aim	To implement Backtracking (Graph Coloring).

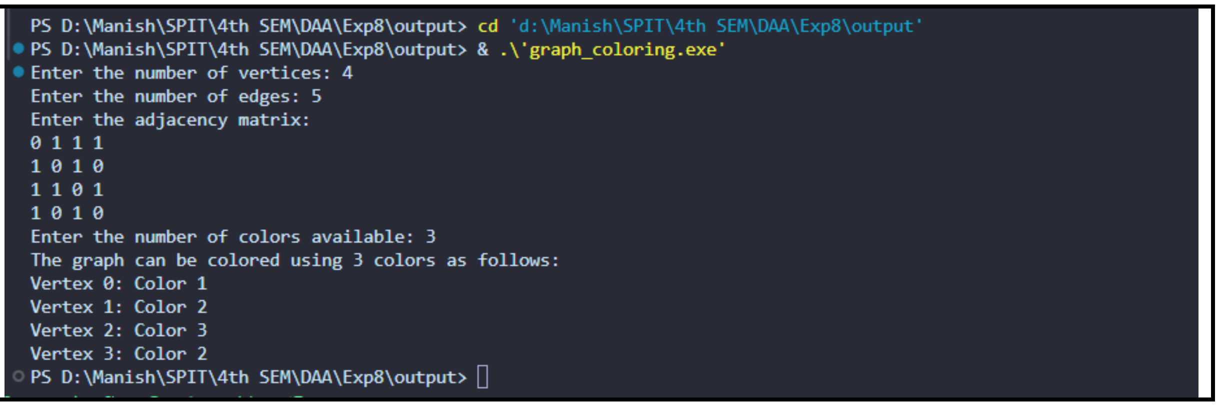
Code:

```c
#include <stdio.h>
#include <stdbool.h>

#define MAX_VERTICES 100

int vertices, edges;
int graph[MAX_VERTICES][MAX_VERTICES];
int colors[MAX_VERTICES];

// Function to check if it's safe to assign color 'c' to vertex 'v'
bool isSafe(int v, int c) {
    for (int i = 0; i < vertices; i++) {
        // Check if there is an edge between 'v' and 'i' and if the color of 'i' is 'c'
        if (graph[v][i] && c == colors[i]) {
            return false;  // If the condition is true, it's not safe to assign 'c' to 'v'
        }
    }
    return true;  // It's safe to assign 'c' to 'v'
}

// Recursive function to perform graph coloring using backtracking
bool graphColoringUtil(int v, int m) {
    // If all vertices are assigned colors, return true
    if (v == vertices) {
        return true;
    }

    // Try assigning colors to 'v'
    for (int c = 1; c <= m; c++) {
        // Check if it's safe to assign color 'c' to vertex
'v'
        if (isSafe(v, c)) {
            colors[v] = c;  // Assign color 'c' to vertex 'v'

            // Recur to assign colors to the rest of the
vertices
            if (graphColoringUtil(v + 1, m)) {
                return true;  // If coloring is possible,
return true
            }

            colors[v] = 0;  // Backtrack: Remove color
assignment if coloring is not possible
        }
    }

    return false;  // If no color can be assigned to this
vertex, return false
}

// Main function to perform graph coloring
bool graphColoring(int m) {
    // Call the utility function with vertex 0 and m colors
    if (!graphColoringUtil(0, m)) {
        printf("No solution exists\n");
        return false;
    }

    printf("The graph can be colored using %d colors as
follows:\n", m);
    for (int i = 0; i < vertices; i++) {
```



```
        printf("Vertex %d: Color %d\n", i, colors[i]);
    }

    return true;
}

int main() {
    // Input the number of vertices
    printf("Enter the number of vertices: ");
    scanf("%d", &vertices);

    // Input the number of edges
    printf("Enter the number of edges: ");
    scanf("%d", &edges);

    // Input the adjacency matrix representing the graph
    printf("Enter the adjacency matrix:\n");
    for (int i = 0; i < vertices; i++) {
        for (int j = 0; j < vertices; j++) {
            scanf("%d", &graph[i][j]);
        }
    }

    // Input the number of colors available
    int m;
    printf("Enter the number of colors available: ");
    scanf("%d", &m);

    // Perform graph coloring
    graphColoring(m);

    return 0;
}
```

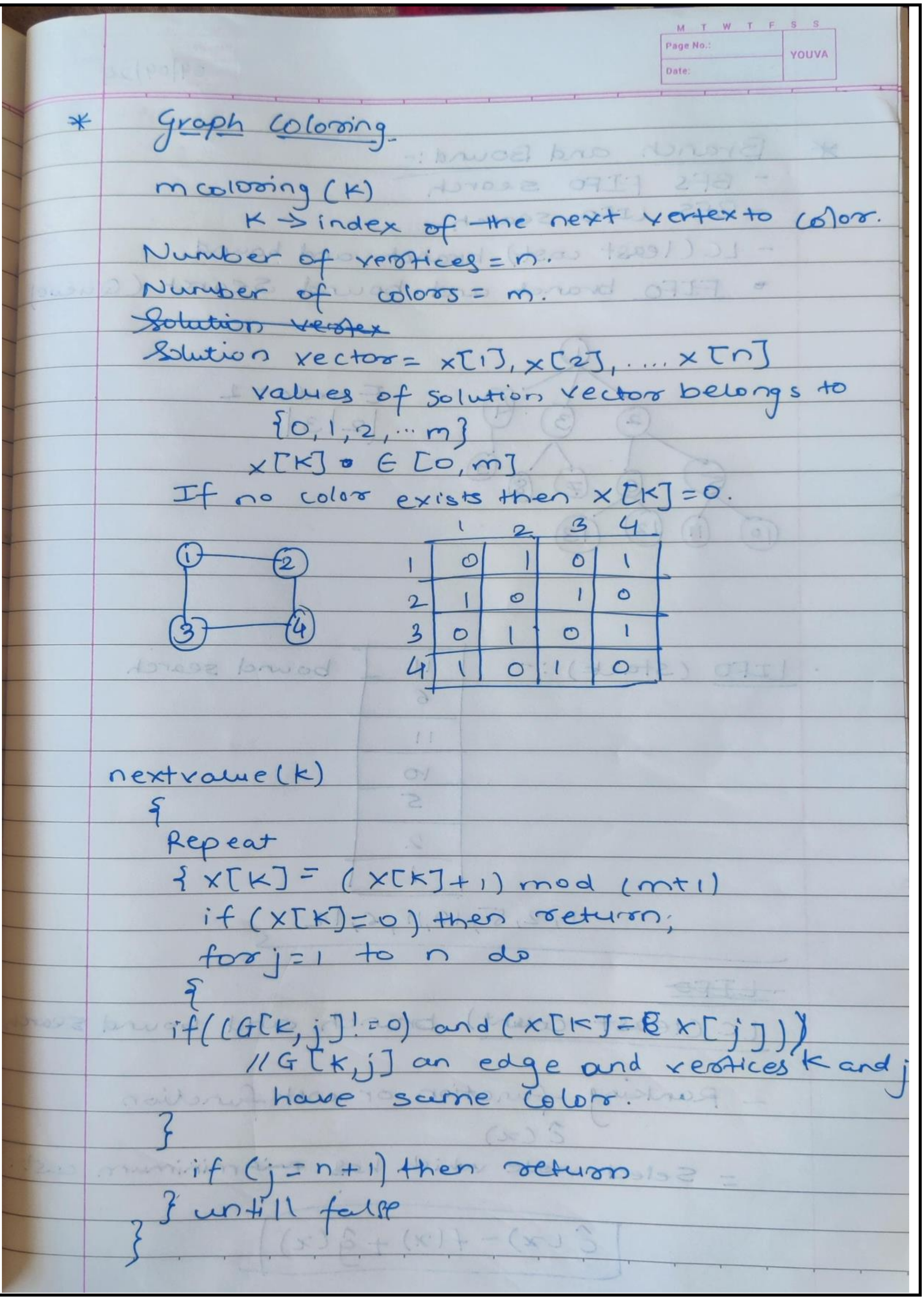
Output



```
PS D:\Manish\SPIT\4th SEM\DAA\Exp8\output> cd 'd:\Manish\SPIT\4th SEM\DAA\Exp8\output'
PS D:\Manish\SPIT\4th SEM\DAA\Exp8\output> & .\'graph_coloring.exe'
Enter the number of vertices: 4
Enter the number of edges: 5
Enter the adjacency matrix:
0 1 1 1
1 0 1 0
1 1 0 1
1 0 1 0
Enter the number of colors available: 3
The graph can be colored using 3 colors as follows:
Vertex 0: Color 1
Vertex 1: Color 2
Vertex 2: Color 3
Vertex 3: Color 2
PS D:\Manish\SPIT\4th SEM\DAA\Exp8\output> 
```

BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
(Empowered Autonomous Institute Affiliated to Mumbai University)
Department Of Computer Engineering

Pseudo Code


M T W T F S S
Page No.:
Date:
YOUVA

* Graph coloring

mcoloring (K)
K -> index of the next vertex to color.
Number of vertices = n.
Number of colors = m.
~~Solution vertex~~
Solution vector = x[1], x[2], ... x[n]
values of solution vector belongs to {0, 1, 2, ... m}
x[K] ∈ [0, m]
If no color exists then x[K] = 0.

	1	2	3	4
1	0	1	0	1
2	1	0	1	0
3	0	1	0	1
4	1	0	1	0

```
nextvalue(K)
{
  Repeat
  { x[K] = (x[K]+1) mod (m+1)
    if (x[K]=0) then return;
    for j=1 to n do
    {
    if((G[k,j]!=0) and (x[K]=B x[j]))
        // G[k,j] an edge and vertices k and j
           have same color.
    }
    if (j=n+1) then return
  } untill false
}
```

Conclusion	Hence, by completing this experiment I came to know about implementation of Graph Coloring using Backtracking.