Министерство высшего образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Пермский национальный исследовательский политехнический университет» (ПНИПУ)
Электротехнический факультет
Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ
Дискретная математика
«Раскраска графа»

Выполнил
Студент группы РИС-22-2б
Прядеин И.А.
Проверил доцент кафедры
ИТАС
Рустамханова Г. И.

Пермь 2024

Постановка задачи:

Дана матрица смежности неориентированного графа, состоящего из 10 вершин.

Найти раскраску исходного графа.

Алгоритм работы:

Считывание файла (рис. 1) реализовано с помощью функции “load_graph”

```
bool load_graph(char *filename) {
    FILE *file;
    file = fopen(filename, "r");
    int c, num, row, column;

    row = column = 0;
    if (file) {
        while ((c = fgetc(file)) != EOF) {
            num = c - '0';
            if (num >= 0 && num <= 100)
                graph[row][column++] = num;
            if (column == 10) {
                column = 0;
                ++row;
            }
        }
    } else {
        printf("Can't open the file.\n");
        return false;
    }

    return true;
}
```

Рис.1 – Считывание матрицы из файла

Основная функция раскраски графа называется “graph_coloring” (рис. 2). Функция заполняет массив цветов и вызывает функцию “graph_coloring_util” (рис. 3), которая рекурсивно проходит по каждой вершине, проверяет можно ли вершине присвоить конкретный цвет и возвращает “true” в случае успеха.

```
bool graph_coloring(int m)
{
    int color[VERTEX_COUNT];
    for (int i = 0; i < VERTEX_COUNT; i++)
        color[i] = 0;

    if (!graph_coloring_util(m, color, 0))
        return false;

    print_solution(color);
    return true;
}
```

Рис. 2 – Функция “graph_coloring”

```
bool graph_coloring_util(int m, int color[], int v) {
    if (v == VERTEX_COUNT)
        return true;

    for (int c = 1; c <= m; c++) {
        if (is_safe(v, color, c)) {
            color[v] = c;

            if (graph_coloring_util(m, color, v + 1) == true)
                return true;

            color[v] = 0;
        }
    }

    return false;
}
```

Рис. 3 – Функция “graph_coloring_util”

Вывод результата происходит с помощью функции “print_solution” (рис. 4).

```
void print_solution(int color[])
{
    printf("Solution Exists:\n");
    for (int i = 0; i < VERTEX_COUNT; i++)
        printf("%d: %d\n", i + 1, color[i]);
    printf("\n");
}
```

Рис. 4 – Вывод результата

Результат тестирования:

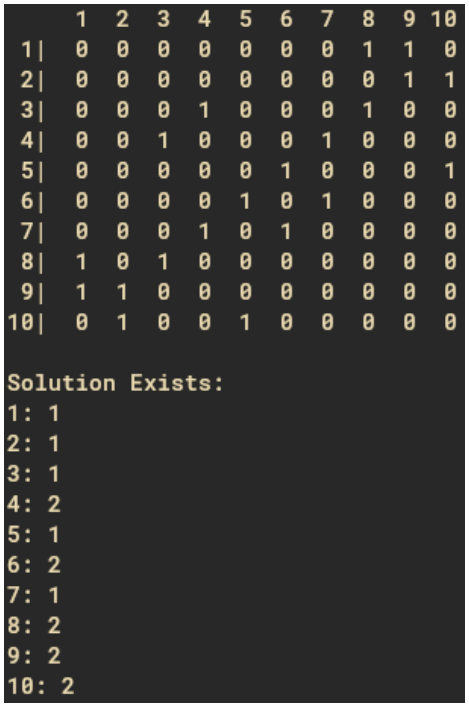

Рис. 5 – Файл “g31”

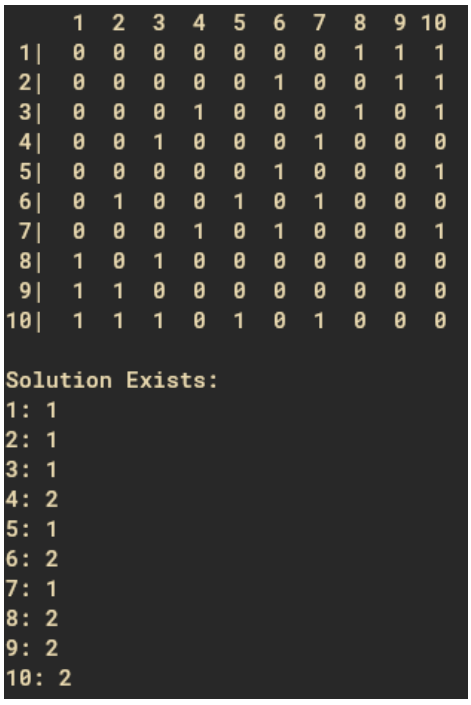

Рис. 6 – Файл “g32”

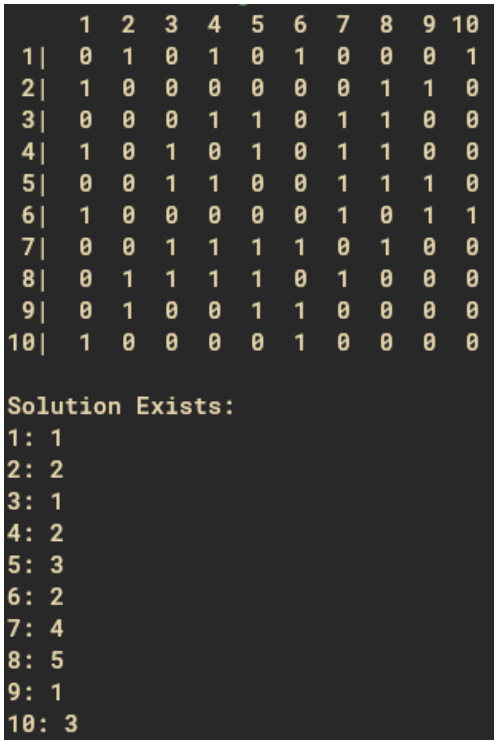

Рис. 7 – Файл “g33”

Исходный код:

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

#define VERTEX_COUNT 10

int graph[VERTEX_COUNT][VERTEX_COUNT] = {0};

bool load_graph(char *filename) {
    FILE *file;
    file = fopen(filename, "r");
    int c, num, row, column;

    row = column = 0;
    if (file) {
        while ((c = fgetc(file)) != EOF) {
            num = c - '0';
            if (num >= 0 && num <= 100)
                graph[row][column++] = num;
            if (column == 10) {
                column = 0;
                ++row;
            }
        }
    } else {
        printf("Can't open the file.\n");
        return false;
    }
```

```
    return true;
}

bool is_safe(int v, int color[], int c) {
    for (int i = 0; i < VERTEX_COUNT; i++)
        if (graph[v][i] && c == color[i])
            return false;
    return true;
}

bool graph_coloring_util(int m, int color[], int v) {
    if (v == VERTEX_COUNT)
        return true;

    for (int c = 1; c <= m; c++) {
        if (is_safe(v, color, c)) {
            color[v] = c;

            if (graph_coloring_util(m, color, v + 1) == true)
                return true;

            color[v] = 0;
        }
    }

    return false;
}

void print_solution(int color[])
{
```

```
    printf("Solution Exists:\n");
    for (int i = 0; i < VERTEX_COUNT; i++)
        printf("%d: %d\n", i + 1, color[i]);
    printf("\n");
}

bool graph_coloring(int m)
{
    int color[VERTEX_COUNT];
    for (int i = 0; i < VERTEX_COUNT; i++)
        color[i] = 0;

    if (!graph_coloring_util(m, color, 0))
        return false;

    print_solution(color);
    return true;
}

int main(int argc, char *argv[]) {
    if (argc == 2) {
        if (!load_graph(argv[1]))
            return 1;
    } else {
        if (!load_graph("g31.txt"))
            return 1;
    }

    printf("    ");
    for (int col = 0; col < VERTEX_COUNT; ++col)
```

```
        printf("%2d ", col + 1);
    printf("\n");
    for (int row = 0; row < VERTEX_COUNT; ++row) {
        printf("%2d| ", row + 1);
        for (int column = 0; column < VERTEX_COUNT; ++column)
            printf("%2d ", graph[row][column]);
        printf("\n");
    }
    printf("\n");

    int colors_number = 1;
    while (!graph_coloring(colors_number++) && colors_number <= 100);
    if (colors_number == 100)
        printf("Solution doesn't exists\n");

    return 0;
}
```