

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования

**АДЫГЕЙСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ**

Инженерно-физический факультет

Кафедра автоматизированных систем обработки информации
и управления

Отчет по практике

Обход графа в глубину и ширину.

2 курс, группа ИВТ АСОИУ

Выполнил:

_____ В. А. Сапунов

«05» 06. 2025 г.

Руководитель:

_____ С. В. Теплоухов

«05» 06. 2025 г.

Майкоп, 2025 г.

1. Введение

- 1) Вариант 8 - Обход графа в глубину и ширину.
- 2) Пример кода, решающего данную задачу
- 3) Скриншот программы

2. Ход работы

```
#include <iostream>
#include <vector>
#include <queue>
#include <stack>
```

```
using namespace std;
```

```
void BFS(const vector<vector<int>>& graph, int startNode) {
    int n = graph.size();
    vector<bool> visited(n, false);
    queue<int> q;

    q.push(startNode);
    visited[startNode] = true;

    cout << "Обход BFS, начиная с узла " << startNode << ": ";

    while (!q.empty()) {
        int currentNode = q.front();
        q.pop();
        cout << currentNode << " ";

        for (int neighbor : graph[currentNode]) {
            if (!visited[neighbor]) {
                visited[neighbor] = true;
                q.push(neighbor);
            }
        }
    }
    cout << endl;
}
```

```
void DFS_recursive(const vector<vector<int>>& graph, vector<bool>& visited, int currentNode) {
    visited[currentNode] = true;
    cout << currentNode << " ";
}
```

```

        for (int neighbor : graph[currentNode]) {
            if (!visited[neighbor]) {
                DFS_recursive(graph, visited, neighbor);
            }
        }
    }

void DFS_iterative(const vector<vector<int>>& graph, int startNode) {
    int n = graph.size();
    vector<bool> visited(n, false);
    stack<int> s;

    s.push(startNode);
    visited[startNode] = true;

    cout << "DFS (итеративный) обход, начинающийся с узла " << startNode << ": ";

    while (!s.empty()) {
        int currentNode = s.top();
        s.pop();
        cout << currentNode << " ";

        for (auto it = graph[currentNode].rbegin(); it != graph[currentNode].rend(); ++it)
            if (!visited[*it]) {
                visited[*it] = true;
                s.push(*it);
            }
    }
    cout << endl;
}

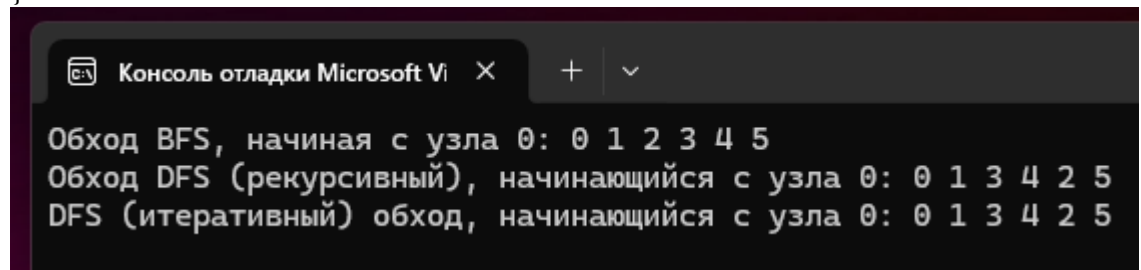
int main() {
    setlocale(0, "ru");
    vector<vector<int>> graph = {
        {1, 2},
        {0, 3, 4},
        {0, 5},
        {1},
        {1},
        {2}
    };

    BFS(graph, 0);

    vector<bool> visited(graph.size(), false);

```

```
cout << "Обход DFS (рекурсивный), начинающийся с узла 0: ";  
DFS_recursive(graph, visited, 0);  
cout << endl;  
  
DFS_iterative(graph, 0);  
}
```



The screenshot shows a debug console window titled "Консоль отладки Microsoft Vi" with a close button. It displays the output of three traversal algorithms starting from node 0: BFS (0 1 2 3 4 5), recursive DFS (0 1 3 4 2 5), and iterative DFS (0 1 3 4 2 5).

```
Обход BFS, начиная с узла 0: 0 1 2 3 4 5  
Обход DFS (рекурсивный), начинающийся с узла 0: 0 1 3 4 2 5  
DFS (итеративный) обход, начинающийся с узла 0: 0 1 3 4 2 5
```