Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №5

**по курсу «Логика и основы алгоритмизации в инженерных задачах»**

на тему «Обход графа в ширину»

Выполнили:

студент группы 20ВВ2

Ермолаев А.Д.

Мартынов Н.В.

Принял:

д.т.н., профессор

Митрохин М.А.

к.т.н., доцент

Юрова О.В.

**Пенза 2021**

**Цель работы:**  составить программу, которая будет осуществлять процедуру обхода графа в ширину и оценить сложность работы.

**Лабораторные задания:**

1.Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного графа G. Выведите матрицу на экран.

2. Для сгенерированного графа осуществите процедуру обхода в ширину, реализованную в соответствии с приведенным выше описанием.

**Листинг:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <queue>

#include <iostream>

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

#include <locale.h>

using namespace std;

queue <int> q; //Объявляем очередь

void BFS(int v, int\*\* M, int\* Vis, int Size)

{

int i;

q.push(v); Vis[v] = 1;

while (!q.empty()) //Выполняем код, пока очередь не пуста

{

v = q.front(); //Устанавливаем текущую вершину

q.pop(); //Извлекаем номер веришны из очереди

printf("%2d", v + 1);

for (i = 0; i < Size; i++)

{

if ((Vis[i] == 0) && (M[v][i] == 1))

{

q.push(i);

Vis[i] = 1;

}

}

}

}

void main()

{

int\*\* G = NULL, i = 0, j = 0, S = 0, \* Visited = NULL, N;

char ReVis = 0; //Индикатор ранее посещённой вершины

setlocale(LC\_ALL, "Rus");

srand(time(NULL));

printf("Введите размерность матрицы: ");

scanf("%d", &S);

while (S <= 0) //Проверка корректности ввода размера матрицы смежности

{

printf("Ошибка при выполнении запроса! Введите целое положительное число : ");

scanf("%d", &S);

}

G = (int\*\*)malloc(S \* sizeof(int)); //Выделяем память

Visited = (int\*)malloc(S \* sizeof(int));

for (i = 0;i < S;i++) //Помечаем все вершины как непосещённые

Visited[i] = 0;

for (i = 0;i < S;i++) //Генерируем матрицу смежности графа

{

G[i] = (int\*)malloc(S \* sizeof(int));

for (j = 0;j < S;j++)

{

G[i][j] = rand() % 2;

if (G[i][i] != 0)

G[i][j] = 0;

}

}

for (i = 0;i < S;i++) //Выводим матрицу смежности на экран

{

printf("\n");

for (j = 0;j < S;j++)

{

G[j][i] = G[i][j];

printf("%2d", G[i][j]);

}

}

printf("\n\nВведите вершину, с которой следует начать обход в ширину: ");

scanf("%d", &N);

while (N<1 || N>S) //Проверяем корректность ввода вершины, с которой нужно начать обход

{

printf("\nОшибка при выполнении запроса!\nВведите целое положительное число в пределах размера матрицы: ");

scanf("%d", &N);

}

printf("\nОбход графа в ширину: ");

BFS(N - 1, G, Visited, S); //Вызываем функцию обхода графа в ширину

printf("\n");

free(Visited); //Очистка памяти

free(G);

}

**Псевдокод:**

Вход: G – матрица смежности графа.

Выход: номера вершин в порядке их прохождения на экране.

Алгоритм ПОШ

1.1. для всех i положим NUM[i] = False пометим как “не посещенную”;

1.2. ПОКА существует “новая” вершина v

1.3. ВЫПОЛНЯТЬ BFS (v).

Алгоритм BFS(v):

2.1. Создать пустую очередь Q = {};

2.2. Поместить v в очередь Q.push(v);

2.3. пометить v как “посещенную” NUM[v] = True;

2.4. ПОКА Q != Æ очередь не пуста ВЫПОЛНЯТЬ

2.5. v = Q.front() установить текущую вершину;

2.6. Удалить первый элемент из очереди Q.pop();

2.7. вывести на экран v;

2.8. ДЛЯ i = 1 ДО size\_G ВЫПОЛНЯТЬ

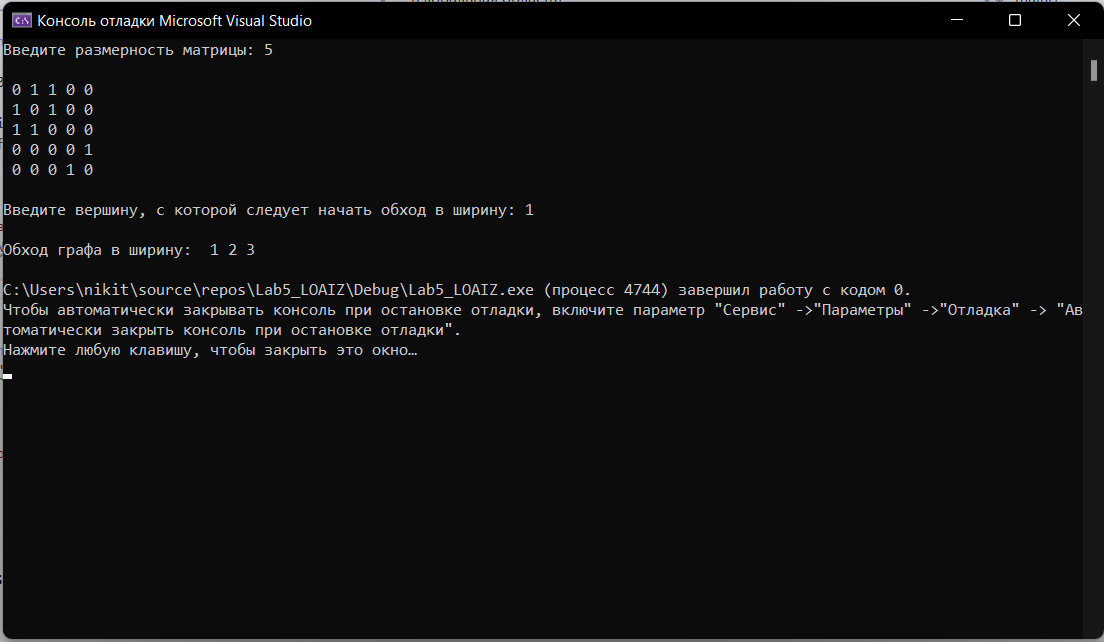
2.9. ЕСЛИ G(v,i) = = 1И NUM[i] = = False

2.10. ТО

2.11. Поместить i в очередь Q.push(i);

2.12. пометить v как “посещенную” NUM[v] = True;

**Результаты программы:**



**Вывод:** составили программу, осуществляющую обход графа в ширину.