Министерство науки и высшего образования Российской Федерации  
Пензенский государственный университет  
Кафедра вычислительная техника

**ОТЧËТ**  
по лабораторной работе №5  
по дисциплине: «Определение характеристик графов»

Выполнили студенты группы 22ВВВ2:  
Ипполитов И. Д.

Сергунов М.Р.

Амиров И. Р.

Приняли:  
Акифьев И. В.

Митрохин М. А.

Пенза 2023

**Название**

Определение характеристик графов

**Цель работы**

Разобраться в графах и матрицах. Научиться определять характеристики графов.

**Лабораторное задание**

**Задание 1**

1. Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного графа *G*. Выведите матрицу на экран.
2. Определите размер графа *G*, используя матрицу смежности графа.
3. Найдите изолированные, концевые и доминирующие вершины.

**Задание 2\***

1. Постройте для графа G матрицу инцидентности.
2. Определите размер графа *G*, используя матрицу инцидентности графа.
3. Найдите изолированные, концевые и доминирующие вершины.

**Листинг**

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#define SIZE 6 // Размер матрицы смежности

void generateMatrix(int matrix[SIZE][SIZE]) {

srand(time(NULL)); // Инициализация генератора случайных чисел

for (int i = 0; i < SIZE; i++) {

for (int j = i + 1; j < SIZE; j++) {

int random = rand() % 2; // Генерация случайного числа 0 или 1

matrix[i][j] = random;

matrix[j][i] = random; // Заполнение элементов и их симметричных пар

}

}

}

int calc(int matrix[SIZE][SIZE]) {

int N = 0;

for (int i = 0; i < SIZE; i++) {

for (int j = 0; j < SIZE; j++) {

if (matrix[i][j] == 1) {

N += 1;

}

}

}

N = N / 2;

return N;

}

int grafCount(int arr[SIZE][SIZE]) {

int sum;

for (int i = 0; i < SIZE; i++)

{

sum = 0;

for (int j = 0; j < SIZE; j++)

{

sum += arr[i][j];

}

if (sum == 1) {

printf("\n%d string: %d vershina - koncevaya\n", i + 1, sum);

}

if (sum == 0) {

printf("\n%d string: %d vershina - izolirovanaya\n", i + 1, sum);

}

if (sum == SIZE - 1) {

printf("\n%d string: %d vershina - dominiruyshaya\n", i + 1, sum);

}

}

}

void printMatrix(int matrix[SIZE][SIZE]) {

for (int i = 0; i < SIZE; i++) {

for (int j = 0; j < SIZE; j++) {

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

}

printf("\n");

}

}

//int GenerateIncidenceMatrix(int matrix[SIZE][SIZE], int countOfEdges)

//{

// int\*\* incMatrix = (int\*\*)malloc(SIZE \* sizeof(int\*));

// for (int i = 0; i < SIZE; i++)

// {

// incMatrix[i] = (int\*)malloc(countOfEdges \* sizeof(int));

// }

//

// int curEdge = 0;

//

// for (int i = 0; i < SIZE; i++)

// {

// for (int j = 0; j < countOfEdges; j++)

// {

// incMatrix[i][j] = 0;

// }

// }

//

// for (int i = 0; i < SIZE; i++)

// {

// for (int j = i; j < SIZE; j++)

// {

// if (matrix[i][j] == 1)

// {

// incMatrix[i][curEdge] = 1;

// incMatrix[j][curEdge] = 1;

// curEdge++;

// }

// }

// }

//

// for (int i = 0; i < SIZE; i++) {

// for (int j = 0; j < countOfEdges; j++) {

// printf("%d ", incMatrix[i][j]);

// }

// printf("\n");

// }

// return \*\*incMatrix;

//}

int sizeOfIncGraph(int\*\* incMatrix, int countOfEdges) {

int N = 0;

int count = 0;

for (int i = 0; i < SIZE; i++) {

for (int j = 0; j < countOfEdges; j++) {

if (incMatrix[i][j] == 1) {

N++;

}

}

if (N == 2) {

count++;

}

N = 0;

}

return count;

}

int incGrafCount(int\*\* arr, int count) {

int sum;

for (int i = 0; i < SIZE; i++)

{

sum = 0;

for (int j = 0; j < count; j++)

{

sum += arr[i][j];

}

if (sum == 1) {

printf("\n%d string: %d vershina - koncevaya\n", i + 1, sum);

}

if (sum == 0) {

printf("\n%d string: %d vershina - izolirovanaya\n", i + 1, sum);

}

if (sum == SIZE - 1) {

printf("\n%d string: %d vershina - dominiruyshaya\n", i + 1, sum);

}

}

}

int main() {

int matrix[SIZE][SIZE] = { 0 };

//Часть 1

generateMatrix(matrix);

printMatrix(matrix);

grafCount(matrix);

int count = calc(matrix);

printf("Size of graf = %d\n\n", count);

//Часть 2

//int incMatrix = GenerateIncidenceMatrix(matrix, count);

int\*\* incMatrix = (int\*\*)malloc(SIZE \* sizeof(int\*));

for (int i = 0; i < SIZE; i++)

{

incMatrix[i] = (int\*)malloc(count \* sizeof(int));

}

int curEdge = 0;

for (int i = 0; i < SIZE; i++)

{

for (int j = 0; j < count; j++)

{

incMatrix[i][j] = 0;

}

}

for (int i = 0; i < SIZE; i++)

{

for (int j = i; j < SIZE; j++)

{

if (matrix[i][j] == 1)

{

incMatrix[i][curEdge] = 1;

incMatrix[j][curEdge] = 1;

curEdge++;

}

}

}

for (int i = 0; i < SIZE; i++) {

for (int j = 0; j < count; j++) {

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

}

printf("\n");

}

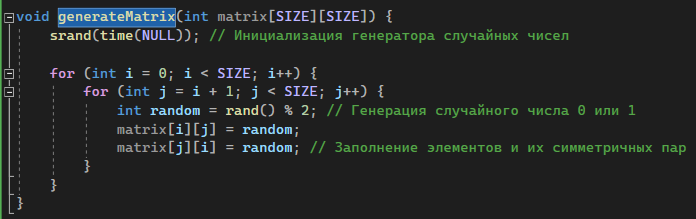
printf("Size of graf = %d\n\n", sizeOfIncGraph(incMatrix, count));

incGrafCount(incMatrix, count);

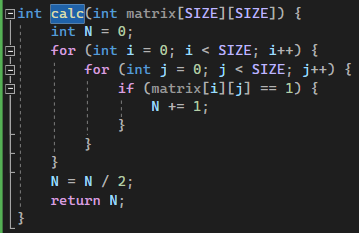
return 0;

} **Задания**

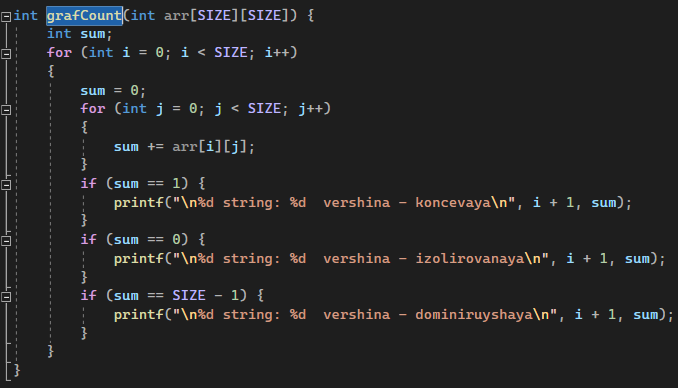
* 1. Матрица смежности



* 1. Размер графа



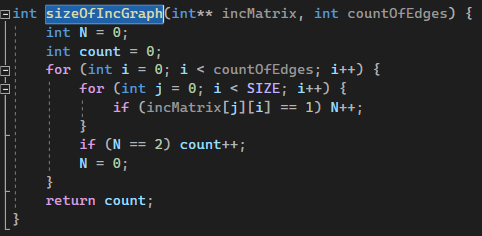
* 1. Характеристики графа



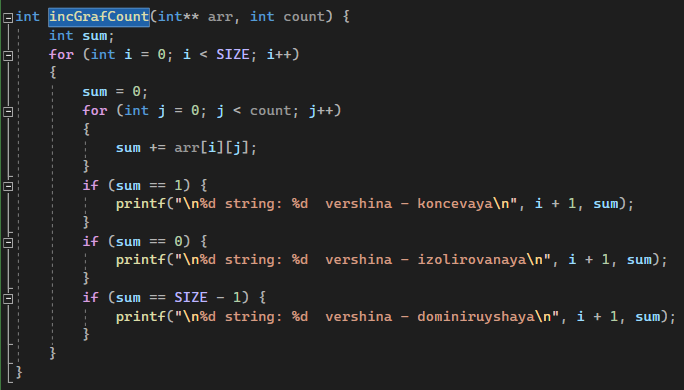
2.1 Матрица инцидентности



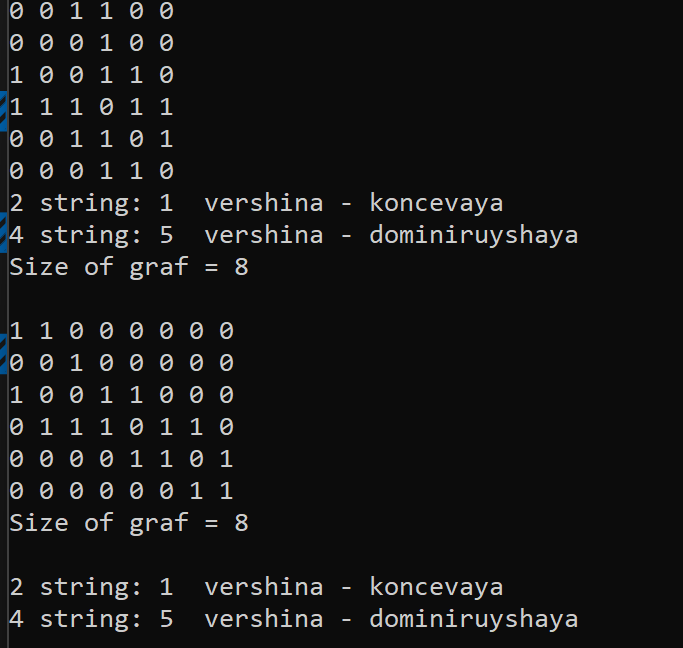
2.2 Размер графа



2.3 Характеристики графа



**Результат работы программы**

****

**Вывод**

Разобрались с графами и матрицами, научились определять характеристики графов, создавать графы, строить матрицы смежности и инцидентности.