Министерство науки и высшего образования Российской федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЁТ**

по лабораторной работе №5

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Определение характеристик графов»

Выполнили ст. группы 22ВВВ1:

Лёвин А.Д.

Колобов И.О.

Приняли:

К.э.н., доцент Акифьев И. В.

К.т.н., доцент Юрова О. В.

Пенза 2023

**Цель работы:**

Цель данной лабораторной работы заключается в изучении определений характеристик графов.

**Лабораторное задание:**

**Задание 1:**

1. Сгенерируйте 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.

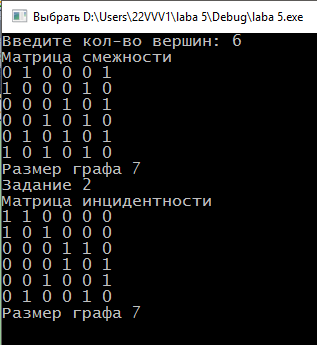
3. Найдите изолированные, концевые и доминирующие вершины.

**Ход работы:**

Создаем первый двумерный динамический массив. Обнуляем его диагональ. Заполняем верхнюю половину рандомно единицами и нулями. Отражаем на нижнюю часть. Считаем количество единиц в верхней части, для того чтобы определить размер графа. Проверяем количество единиц в строчке, чтобы определить какие вершины. Создает второй динамический массив. Строим его на основе первой матрицы. Так же выводим его размер и типы вершин.

**Результаты работы программы:**

Задание 1 и 2:



**Вывод:**

В ходе лабораторной работы научились создавать матрицы смежности и инцидентности, реализовали функцию определения типа вершины.

**Листинг:**

#include "stdafx.h"

#include <iostream>

#include <stdio.h>

#include <stdlib.h>

#include <Windows.h>

#include <time.h>

int main() {

SetConsoleCP(1251);//Установка языка

SetConsoleOutputCP(1251);

srand(time(NULL));

int n;

int raz;

printf("Введите кол-во вершин: ");

scanf\_s("%d", &n);

int\*\* Matr1 = (int\*\*)malloc(n \* sizeof(int\*));

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

Matr1[i] = (int\*)malloc(n \* sizeof(int));

}

raz = 0;

printf("Матрица смежности\n");

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

for (int j = i; j < n; j++) {

if (i == j) {

Matr1[i][j] = 0;

}

else {

Matr1[i][j] = rand() % 2;

Matr1[j][i] = Matr1[i][j];

}

}

}

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

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

Matr1[i][j] = Matr1[j][i];

}

}

//вывод матрицы

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

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

printf("%d ", Matr1[i][j]);

}

printf("\n");

}

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

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

if (Matr1[i][j] == 1)

{

raz++;

}

}

}

int\*\* Matr2 = (int\*\*)malloc(n \* sizeof(int\*));

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

Matr2[i] = (int\*)malloc(raz \* sizeof(int));

}

printf("Размер графа %d\n", raz);

int\* Str = (int\*)malloc(n \* sizeof(int));

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

Str[i] = 0;

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

if (Matr1[i][j] == 1) {

Str[i]++;

}

}

if (Str[i] == 0) {

printf("Строка %d: Изолированная вершина\n", i);

}

else if (Str[i] == 1) {

printf("Строка %d: Концевая вершина\n", i);

}

else if (Str[i] == n - 1) {

printf("Строка %d: Доминирующая вершина\n", i);

}

}

free(Str);

printf("Задание 2\n");

printf("Матрица инцидентности\n");

raz = 0;

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

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

if (Matr1[i][j] == 1) {

for (int k = 0; k < n; k++) {

if (k == j || k == i) {

Matr2[k][raz] = 1;

}

else {

Matr2[k][raz] = 0;

}

}

raz++;

}

}

}

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

for (int j = 0; j < raz-1; j++) {

printf("%d ", Matr2[i][j]);

}

printf("\n");

}

printf("Размер графа %d\n", raz);

int\* Str1 = (int\*)malloc(n \* sizeof(int));

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

Str1[i] = 0;

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

if (Matr2[i][j] == 1) {

Str1[i]++;

}

}

if (Str1[i] == 0) {

printf("Строка %d: Изолированная вершина\n", i);

}

else if (Str1[i] == 1) {

printf("Строка %d: Концевая вершина\n", i);

}

else if (Str1[i] == n - 1) {

printf("Строка %d: Доминирующая вершина\n", i);

}

}

free(Str1);

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

Matr1[i] = (int\*)malloc(n \* sizeof(int));

Matr2[i] = (int\*)malloc(raz \* sizeof(int));

}

free(Matr1);

free(Matr2);

getchar();

getchar();

//return 0;

}