Пензенский государственный университет

Кафедра «Вычислительная техника»

ОТЧЕТ

по лабораторной работе № 5

по дисциплине: "Логика и основы алгоритмизации в инженерных задачах"

## на тему: "Определение характеристик графов"

Выполнили студенты гр. 22ВВП2:

Корнилов В.М.

Самофалова А.В.

Горбунов Д.А.

Приняли:

Акифьев И.В.

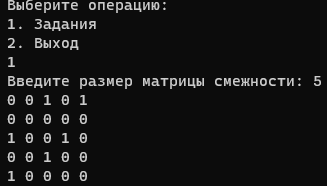
Юрова О.В.

Пенза, 2023

**Цель:** научиться генерировать матрицу смежности, и с помощью неё создавать матрицу инцидентности, а также считать размерность графа и искать изолированные, концевые и доминирующие вершины.

**Задание 1**

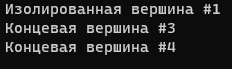
1. Сгенерировали (используя генератор случайных чисел) матрицу смежности для неориентированного графа *G*. Вывели матрицу на экран.



1. Определили размер графа *G*, используя матрицу смежности графа.

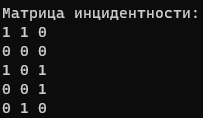


1. Нашли изолированные, концевые и доминирующие вершины.

****

**Задание 2\***

1. Постройте для графа G матрицу инцидентности.



1. Определили размер графа *G*, используя матрицу инцидентности графа.



1. Нашли изолированные, концевые и доминирующие вершины.



**Вывод:** научились генерировать матрицу смежности, и с помощью неё создавать матрицу инцидентности, а также считать размерность графа и искать изолированные, концевые и доминирующие вершины.

**Приложение А (Листинг)**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <windows.h>

#include <locale.h>

#include <time.h>

int menu() {

int n = 0;

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice;

while (1) {

printf("\nВыберите операцию:\n");

printf("1. Задания\n");

printf("2. Выход\n");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("Введите размер матрицы смежности: ");

scanf("%d", &n);

smej(n);

break;

case 2:

exit(0);

break;

default:

printf("Неверный выбор.\n");

}

}

return 0;

}

int smej(int n) {

// Создание и заполнение матрицы смежности

int\*\* A;

int count = 0;

srand(time(NULL));

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

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

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

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

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

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

A[j][i] = A[i][j];

if (i == j) A[i][j] = 0;

if (A[i][j] == 1) count++;

}

// Вывод матрицы смежности

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

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

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

}

printf("\n");

}

//Поиск Доминирующей, изолированной и концевой вершин

printf("Размер графа: %d\n", count);

int\* st;

st = (int\*)calloc(n,sizeof(int));

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

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

if (A[i][j] == 1) st[i]++;

}

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

if (st[i] == (n - 1))

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

if (st[i] == 0)

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

if (st[i] == 1)

printf("Концевая вершина #%d\n", i);

}

// Создание и заполнение матрицы инцидентности

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

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

B[i] = (int\*)calloc(count, sizeof(int));

}

int reb = 0;

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

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

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

B[i][reb] = 1;

B[j][reb] = 1;

reb++;

}

}

}

// Вывод матрицы инцидентности

printf("\nМатрица инцидентности:\n");

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

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

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

}

printf("\n");

}

int gs = 0;

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

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

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

gs++;

break;

}

}

}

printf("Размер графа по матрице инцидентности: %d\n", gs);

//Поиск Доминирующей, изолированной и концевой вершин

int\* s;

s = (int\*)calloc(n, sizeof(int));

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

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

if (B[i][j] == 1) s[i]++;

}

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

if (s[i] == gs)

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

if (s[i] == 0)

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

if (s[i] == 1)

printf("Концевая вершина #%d\n", i);

}

// Освобождение памяти

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

free(B[i]);

free(B);

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

free(A[i]);

free(A);

}

int main() {

menu();

}