Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**Отчёт**

по дисциплине «Логика и основы алгоритмизации в инженерных задачах»

по лабораторной работе №5

## «Определение характеристик графов»

Выполнили:

студенты группы 22ВВС1

Эрмедов А.Э.

Казаров И.И.

Приняли:

к.т.н, доцент Юрова О.В.

к.э.н, доцент Акифьев И.В.

Пенза 2023

**Название**

Оценка времени выполнения программ

**Практическая часть**

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.

**Листинг labloaviz51.cpp**

#include <iostream>

#include <vector>

#include <cstdlib>

#include <ctime>

#include <locale.h>

// Функция для генерации случайного числа в заданном диапазоне

int generateRandomNumber(int min, int max) {

return min + rand() % (max - min + 1);

}

int main() {

setlocale(LC\_ALL, "ru\_RU.UTF-8");

srand(time(0)); // Инициализация генератора случайных чисел

int n;

std::cout << "Введите размер матрицы: ";

std::cin >> n;

std::cout << "\n";

// Генерация и вывод матрицы смежности

std::vector<std::vector<int>> adjacencyMatrix(n, std::vector<int>(n));

int edgesCount = 0; // Переменная для подсчета количества ребер

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

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

adjacencyMatrix[i][j] = generateRandomNumber(0, 1);

adjacencyMatrix[j][i] = adjacencyMatrix[i][j];

if (i == j) {

adjacencyMatrix[i][j] = 0;

}

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

edgesCount++;

}

}

}

edgesCount = edgesCount / 2;

std::cout << "Матрица смежности:\n";

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

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

std::cout << adjacencyMatrix[i][j] << " ";

}

std::cout << "\n";

}

std::cout << "\n";

// Вывод размера графа (количество ребер)

std::cout << "Размер графа (количество ребер): " << edgesCount << "\n";

std::cout << "\n";

// Поиск изолированных, конечных и доминирующих вершин

std::vector<int> isolatedVertices, terminalVertices, dominatingVertices;

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

int degree = 0;

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

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

degree++;

}

}

if (degree == 0) {

isolatedVertices.push\_back(i);

}

else if (degree == 1) {

terminalVertices.push\_back(i);

}

else if (degree == n - 1) {

dominatingVertices.push\_back(i);

}

}

// Вывод изолированных вершин

std::cout << "Изолированные вершины: ";

if (isolatedVertices.empty()) {

std::cout << "нет";

}

else {

for (const auto& vertex : isolatedVertices) {

std::cout << vertex << " ";

}

}

std::cout << "\n";

// Вывод конечных вершин

std::cout << "Концевые вершины: ";

if (terminalVertices.empty()) {

std::cout << "нет";

}

else {

for (const auto& vertex : terminalVertices) {

std::cout << vertex << " ";

}

}

std::cout << "\n";

// Вывод доминирующих вершин

std::cout << "Доминирующие вершины: ";

if (dominatingVertices.empty()) {

std::cout << "нет";

}

else {

for (const auto& vertex : dominatingVertices) {

std::cout << vertex << " ";

}

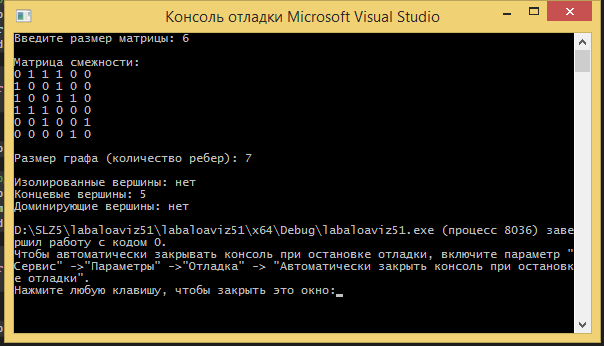
}

std::cout << "\n";

return 0;

}

**Результат**

****

**Вывод**

Задание:

1. Сгенерировали (используя генератор случайных чисел) матрицу смежности для неориентированного графа *G*. Вывели матрицу на экран.
2. Определили размер графа *G*, используя матрицу смежности графа.
3. Нашли изолированные, концевые и доминирующие вершины.