МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Институт математики и информационных систем
Факультет автоматики и вычислительной техники
Кафедра электронных вычислительных машин

Дата сдачи на проверку:

«__»_____ 2025 г.

Проверено:

«__»_____ 2025 г.

ГРАФЫ. ОРГРАФЫ. МАТРИЦА СМЕЖНОСТИ.

Отчёт по лабораторной работе №2

по дисциплине

«Дискретная математика»

Разработал студент гр. ИВТб-1301-05-00 _____/Черкасов А. А./

(подпись)

Проверила преподаватель _____/Пахарева И. В./

(подпись)

Работа защищена «__»_____ 2025 г.

Киров

2025

Цель

Цель работы: Разработать программу, генерирующую случайную матрицу смежности и анализировать полустепень захода вершин графа по заданию.

Задание

– Орграф задается матрицей смежности, которая формируется случайным образом. Размерность $4 \leq n \leq 10$ вводится с клавиатуры.

– Найти кол-во вершин с заданной полустепенью захода, которая вводится с клавиатуры, вывести множества дуг соответственно найденным вершинам (имя множества - номер вершины).

Решение

Схема алгоритма решения представлена на рисунке 1. Исходный код решений представлен в Приложении А1.

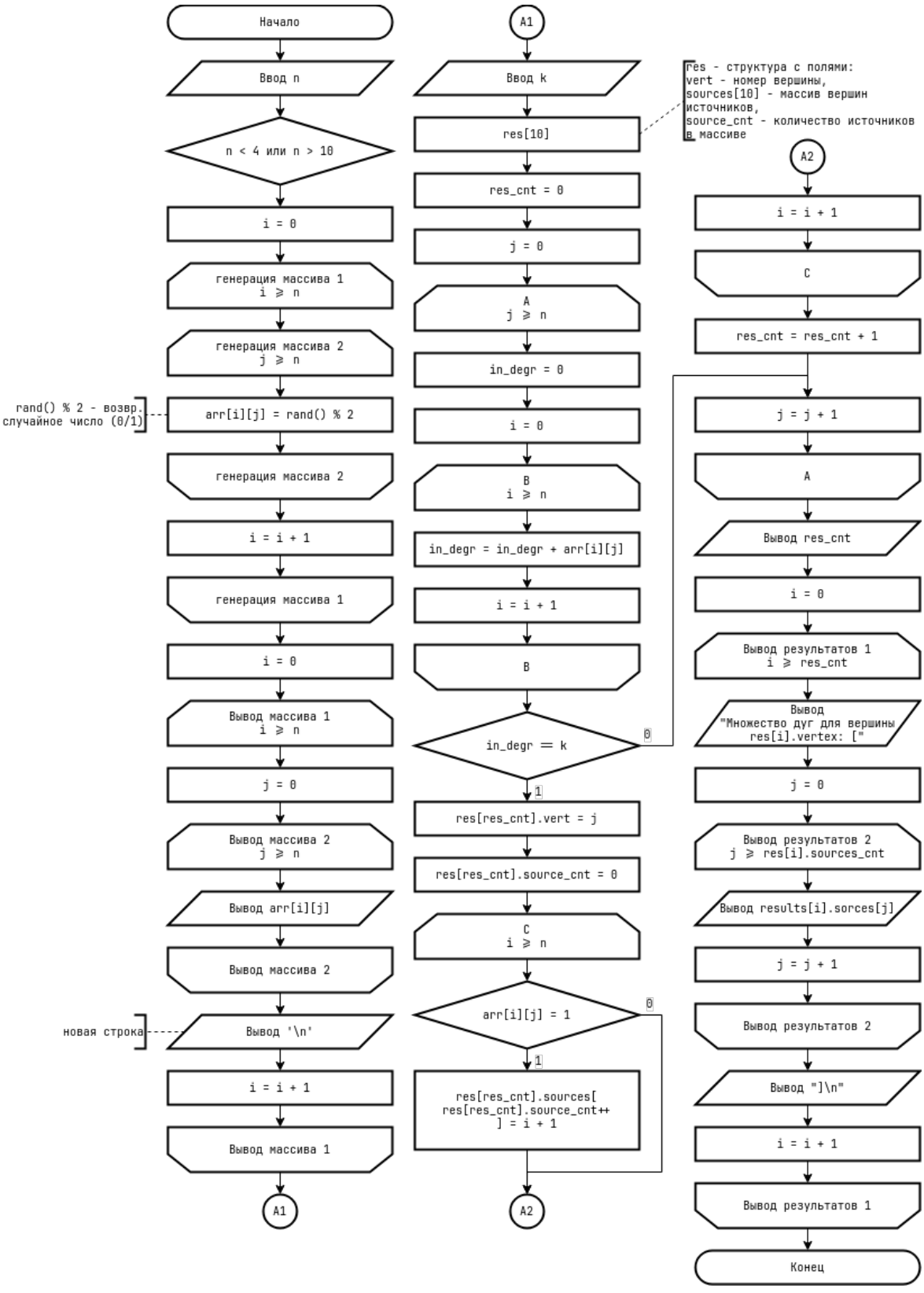

Рисунок 1 - Схема алгоритма Задания.

Вывод

В ходе выполнения лабораторной работы была реализована программа, генерирующая случайную матрицу смежности для орграфа заданного размера. На основе полученной матрицы произведён анализ полустепеней захода всех вершин графа. Программа позволила определить количество вершин, имеющих заданную пользователем полустепень захода, а также отобразить множества входящих дуг для каждой из таких вершин.

Приложение А1. Исходный код

```rust
mod graphviz;

use rand::Rng;
use std::env;
use std::io;

fn main() {
    // Обработка аргументов командной строки
    let enable_visualization = {
        let args: Vec<String> = env::args().collect();
        args.contains(&"-viz".to_string()) || args.contains(&"--visualize".to_string())
    };

    // Ввод размерности матрицы
    let n = loop {
        println!("Введите размерность матрицы (4-10):");
        let mut input = String::new();
        io::stdin().read_line(&mut input).expect("Ошибка чтения");

        match input.trim().parse::<usize>() {
            Ok(num) if (4..=10).contains(&num) => break num,
            _ => println!("Некорректный ввод! Введите число от 4 до 10"),
        }
    };

    // Генерация случайной матрицы смежности
    let mut matrix = vec![vec![0; n]; n];
    let mut rng = rand::rng();

    for i in 0..n {
        for j in 0..n {
            matrix[i][j] = if rng.random_bool(0.5) { 1 } else { 0 };
```

```
    }
}

// Сохранение графа только при необходимости
if enable_visualization {
    if let Err(e) = graphviz::save_dot(&matrix, n, "graph.dot") {
        eprintln!("Ошибка сохранения графа: {}", e);
        return;
    }
} else {
    println!("Визуализация отключена");
}

// Вывод матрицы
println!("\nСгенерированная матрица смежности:");
for row in &matrix {
    println!("{:?}", row);
}

// Ввод искомой полустепени захода (оставлено без изменений)
let k = loop {
    println!("\nВведите искомую полустепень захода:");
    let mut input = String::new();
    io::stdin().read_line(&mut input).expect("Ошибка чтения");

    match input.trim().parse::<usize>() {
        Ok(num) => break num,
        _ => println!("Некорректный ввод! Введите целое число"),
    }
};

// Поиск вершин (оставлено без изменений)
let mut results = Vec::new();
```

```
    for j in 0..n {
        let in_degree: usize = matrix.iter().map(|row| row[j]).sum();

        if in_degree == k {
            let sources: Vec<usize> = matrix
                .iter()
                .enumerate()
                .filter(|(i, _)| matrix[*i][j] == 1)
                .map(|(i, _)| i + 1)
                .collect();

            results.push((j + 1, sources));
        }
    }

    // Вывод результатов (оставлено без изменений)
    println!("\nРезультаты:");
    println!("Найдено вершин: {}", results.len());

    for (vertex, arcs) in results {
        println!("Множество дуг для вершины {}: {:?}", vertex, arcs);
    }
}
```