Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Программная инженерия
кафедра

ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №1

Конечные автоматы
тема

Преподаватель _____ А. С. Кузнецов
подпись, дата — инициалы, фамилия

Студент КИ23-16/1Б, 032321684 _____ К. М. Дорошев
номер группы, зачетной книжки — подпись, дата — инициалы, фамилия

Красноярск 2025

1 Цель

Реализация и исследование детерминированных и недетерминированных конечных автоматов.

2 Задание

Вариант – 10.

Для выполнения практической работы необходимо разработать в системе JFLAP конечные автоматы и произвести программную реализацию на языке Java для следующих автоматов:

1) Построить ДКА, допускающий в алфавите {0, 1} все цепочки нулей и единиц, где самый левый символ отличается от самого правого символа.

2) Построить НКА, допускающий язык из цепочек, состоящих либо из повторяющихся 1 или более раз подцепочек 01, либо из повторяющихся 1 или более раз подцепочек 010.

3 Ход выполнения

Для начала была установлена программа JFLAP, в которой были построены конечные автоматы из условия задания. Каждый КА был сначала протестирован в JFLAP тестовыми цепочками, затем была написана программная реализация на Java, которая также была протестирована на корректность работы теми же самыми тестовыми цепочками.

3.1 Построение ДКА

Необходимо было реализовать детерминированный конечный автомат (ДКА). Исходя из формулировки задачи было выдвинуто предположение, что на вход автомату может поступать строка неограниченной длины и необходимо определить, является ли пятый символ с конца единицей. Для решения данной задачи был разработан ДКА, который запоминает первый символ цепочки, отслеживает текущий символ (последний) символ по мере обработки целочки и в последнюю очередь принимает решение в конце обработки цепочки.

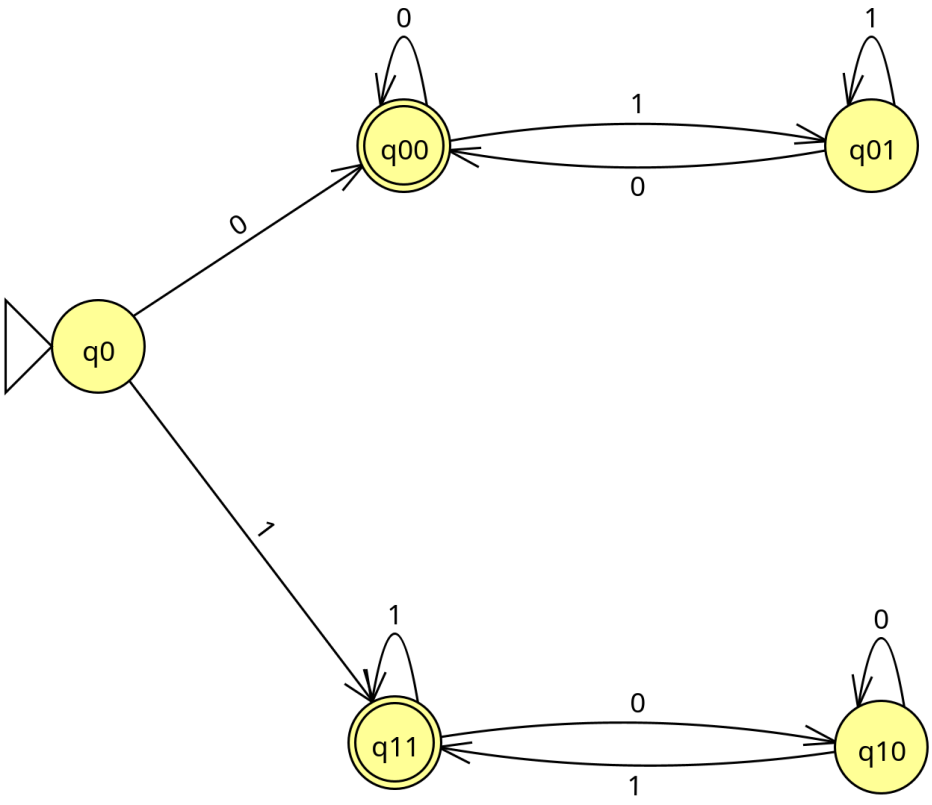


Рисунок 1 – ДКА в JFLAP

На рисунке 2 показан тест ДКА для цепочки «101».

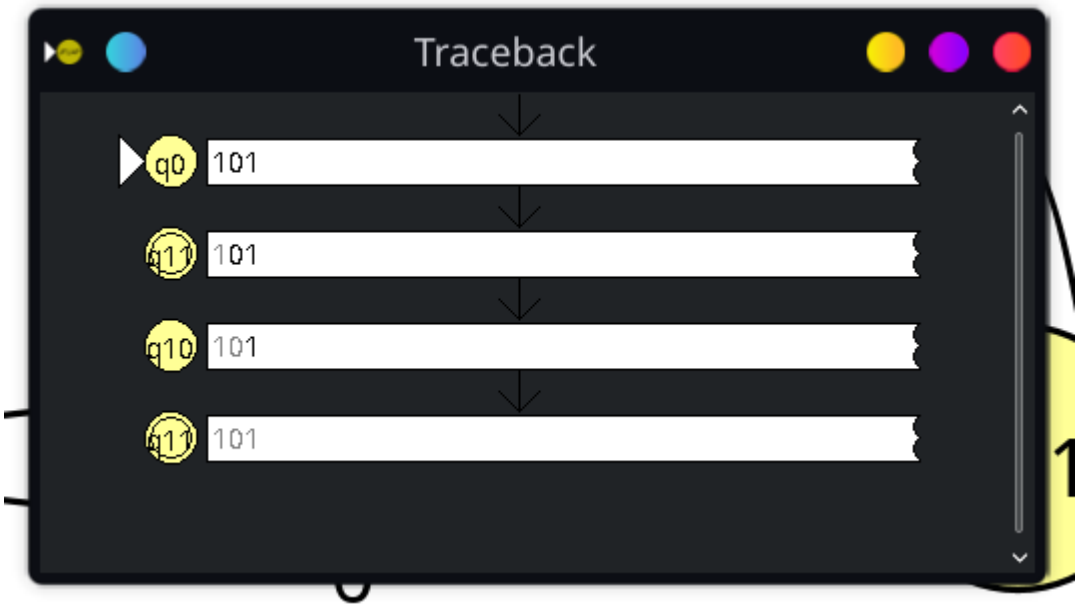


Рисунок 2 – Тест для цепочки «101»

Также была проверка на отклонение строки, которая не удовлетворяет условию. На рисунке 3 показан тест ДКА для цепочки «110».

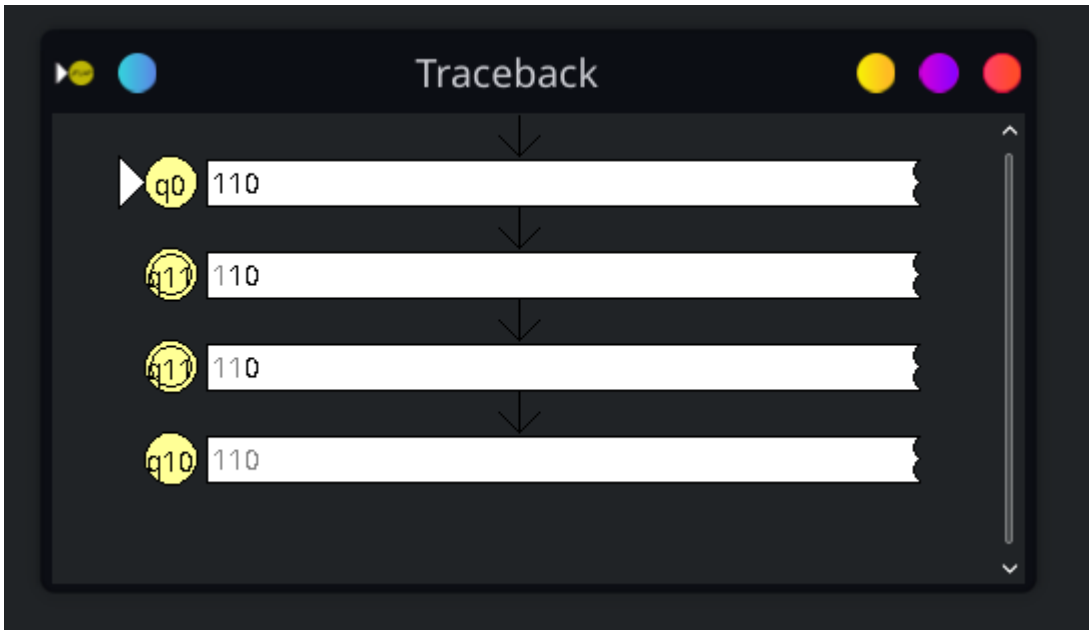


Рисунок 3 – Тест для цепочки «110»

Дальше был написан код на Java для реализации данного ДКА, он показан на рисунке 4.

```java
import java.util.*;

class DFA {
    private final Map<String, Map<String, String[]>> transitions;

    public DFA() {
        transitions = new HashMap<>();

        addTransition(null, null, "0", new String[]{"0", "0"});
        addTransition(null, null, "1", new String[]{"1", "1"});

        addTransition("0", "0", "0", new String[]{"0", "0"});
        addTransition("0", "0", "1", new String[]{"0", "1"});
        addTransition("1", "1", "0", new String[]{"1", "0"});
        addTransition("1", "1", "1", new String[]{"1", "1"});

        addTransition("0", "1", "0", new String[]{"0", "0"});
        addTransition("0", "1", "1", new String[]{"0", "1"});
        addTransition("1", "0", "0", new String[]{"1", "0"});
        addTransition("1", "0", "1", new String[]{"1", "1"});
    }

    private void addTransition(String firstChar, String lastChar, String input, String[] nextState) {
        String key = firstChar + "," + lastChar;
        transitions.computeIfAbsent(key, k -> new HashMap<>()).put(input, nextState);
    }

    public boolean accepts(String inputChain) {
        if (inputChain.isEmpty()) {
            return true;
        }

        String firstChar = null;
        String lastChar = null;

        for (char c : inputChain.toCharArray()) {
            String symbol = String.valueOf(c);
            String key = firstChar + "," + lastChar;

            if (!transitions.containsKey(key) || !transitions.get(key).containsKey(symbol)) {
                return false;
            }

            String[] nextState = transitions.get(key).get(symbol);
            firstChar = nextState[0];
            lastChar = nextState[1];
        }

        return Objects.equals(firstChar, lastChar);
    }
}
```

Рисунок 4 – Код для ДКА на Java

После компиляции, сборки и запуска программы были повторно проведены тесты, показанные на рисунке 5.

```
DFA results:
 0 -> ACCEPT
 00 -> ACCEPT
 01 -> REJECT
 101 -> ACCEPT
 110 -> REJECT
 111 -> ACCEPT
```

Рисунок 5 – Тесты для ДКА на Java

3.2 Построение НКА

Также необходимо было реализовать недетерминированный конечный автомат (НКА) допускающий язык из цепочек из 0 и 1, в которых хотя бы на одной из последних пяти позиций стоит 1. На рисунке 6 показана реализация ДКА первого задания в системе JFLAP.

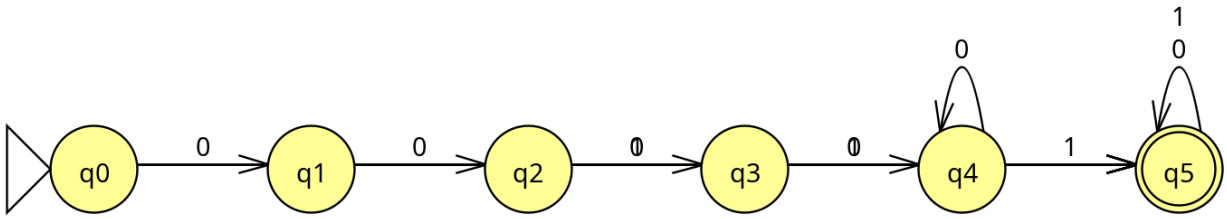


Рисунок 6 - НКА в JFLAP.

Дальше необходимо было провести все самые главные тесты для этой цепочки, они показаны на рисунке 7.

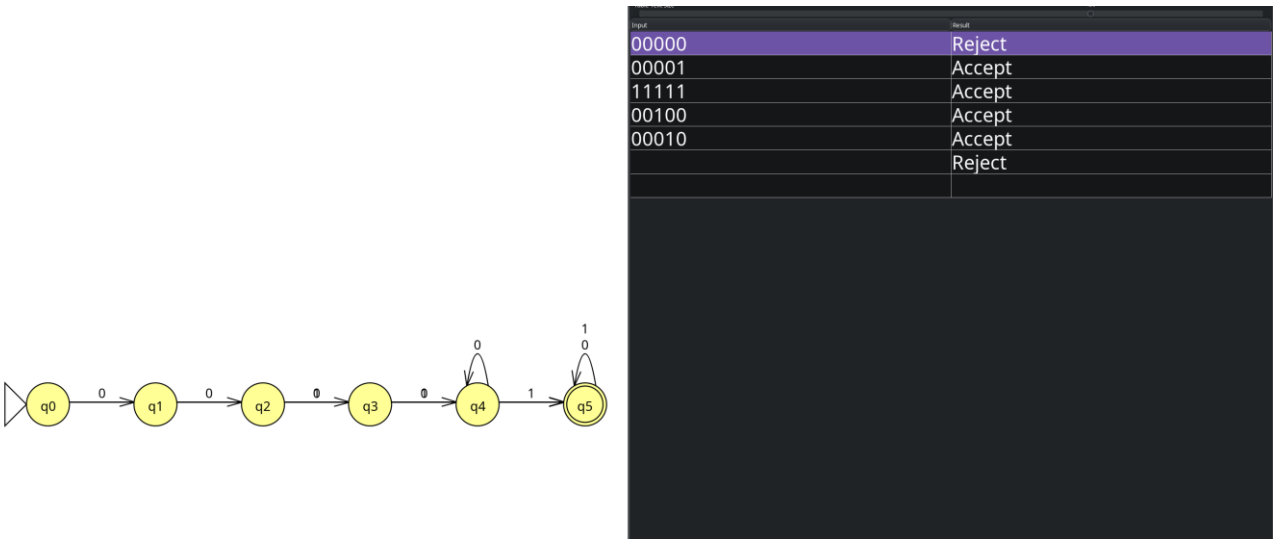
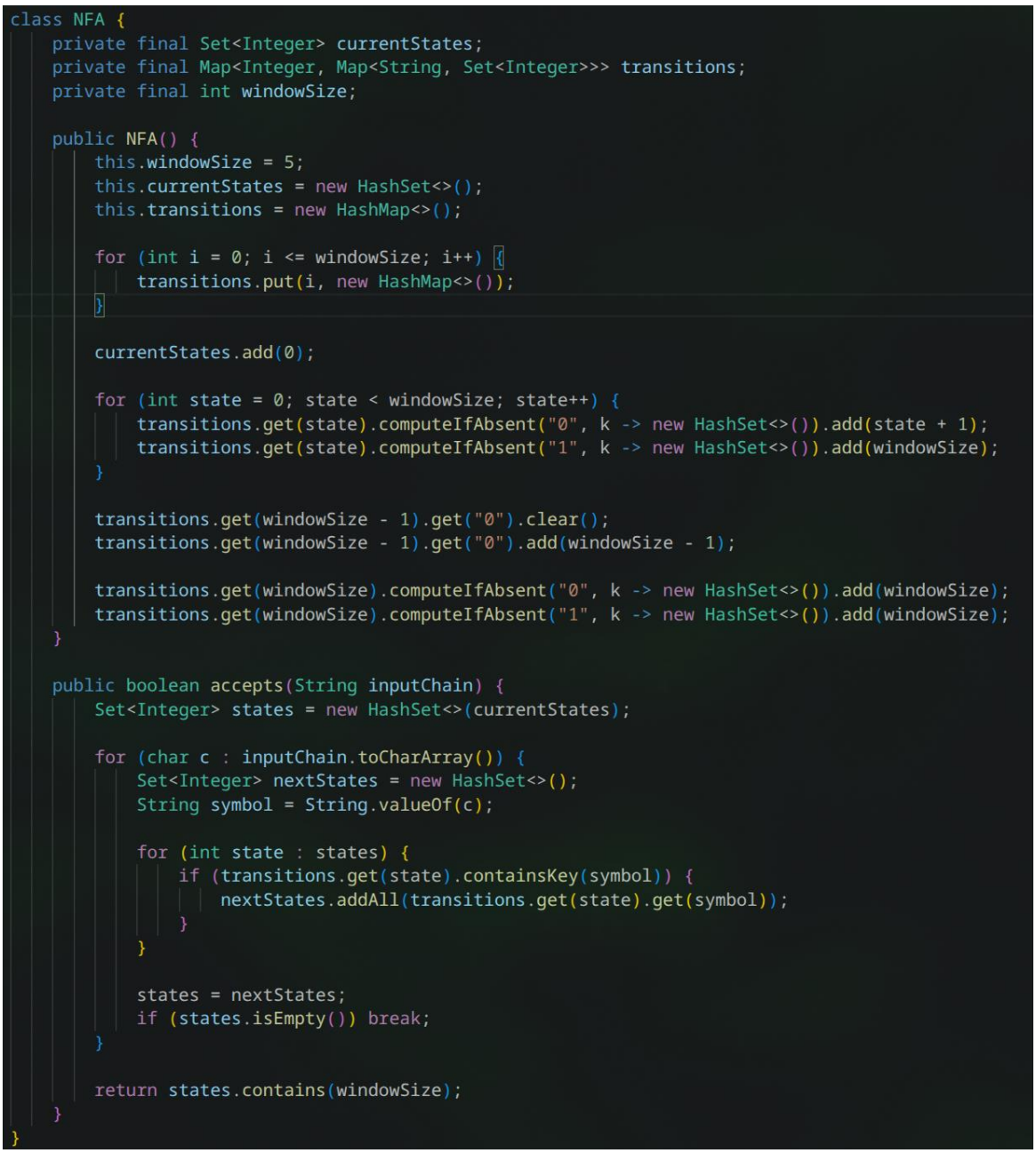


Рисунок 7 – Тесты для НКА.

Все тесты были пройдены успешно. На рисунке 8 показан код на Java для реализации данного НКА.



```
class NFA {
    private final Set<Integer> currentStates;
    private final Map<Integer, Map<String, Set<Integer>>> transitions;
    private final int windowSize;

    public NFA() {
        this.windowSize = 5;
        this.currentStates = new HashSet<>();
        this.transitions = new HashMap<>();

        for (int i = 0; i <= windowSize; i++) {
            transitions.put(i, new HashMap<>());
        }

        currentStates.add(0);

        for (int state = 0; state < windowSize; state++) {
            transitions.get(state).computeIfAbsent("0", k -> new HashSet<>()).add(state + 1);
            transitions.get(state).computeIfAbsent("1", k -> new HashSet<>()).add(windowSize);
        }

        transitions.get(windowSize - 1).get("0").clear();
        transitions.get(windowSize - 1).get("0").add(windowSize - 1);

        transitions.get(windowSize).computeIfAbsent("0", k -> new HashSet<>()).add(windowSize);
        transitions.get(windowSize).computeIfAbsent("1", k -> new HashSet<>()).add(windowSize);
    }

    public boolean accepts(String inputChain) {
        Set<Integer> states = new HashSet<>(currentStates);

        for (char c : inputChain.toCharArray()) {
            Set<Integer> nextStates = new HashSet<>();
            String symbol = String.valueOf(c);

            for (int state : states) {
                if (transitions.get(state).containsKey(symbol)) {
                    nextStates.addAll(transitions.get(state).get(symbol));
                }
            }

            states = nextStates;
            if (states.isEmpty()) break;
        }

        return states.contains(windowSize);
    }
}
```

Рисунок 8 – Код для НКА на C++

После компиляции, сборки и запуска программы были повторно проведены тесты, показанные на рисунке 9.

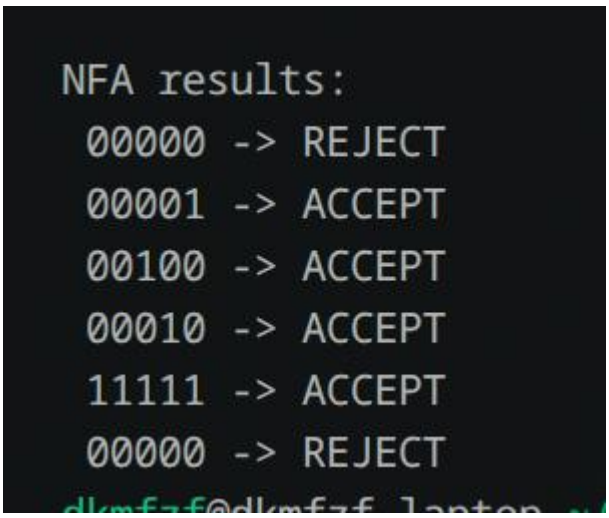


Рисунок 9 – Тесты для НКА на Java

4 Выводы

В ходе данной практической работы был изучен материал о детерминированных и недетерминированных конечных автоматах, были выполнены все задания, построены ДКА и НКА в системе JFLAP и реализован программный код на Java.