Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное учреждение высшего образования  
«Пермский национальный исследовательский политехнический университет»

Электротехнический факультет  
Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

Дискретная математика

Семестр 4

Лабораторная работа 4

«Теория автоматов»

Выполнил  
студент группы РИС-22-2б  
Баяндин К. С.

Проверила  
старший преподаватель кафедры ИТАС  
Рустамханова Г.И.

Пермь, 2024

Постановка задачи

Синтезировать автомат, распознающий заданный язык. Написать программу-анализатор, которая определяет, принадлежит ли слово заданному языку.

Формальный язык задан в алфавите и содержит слова любой длины

*Вариант 4:*

|  |
| --- |
| начинающиеся и заканчивающиеся одинаковым символом |

Анализ алгоритма работы программы

Распознающий автомат – это автомат Мура, в котором фиксируется начальное состояние и подмножество состояний F⊆ Q, называемое множеством заключительных (финальных) состояний. Говорят, что автомат допускает (принимает, распознает, представляет) данное слово, если реакцией на это слово может быть переход автомата в одно из заключительных состояний.

Входные данные: {a, b, c, d}

Состояния: {qS, qA, qB, qC, qD, qAA, qBB, qCC, qDD, qF}

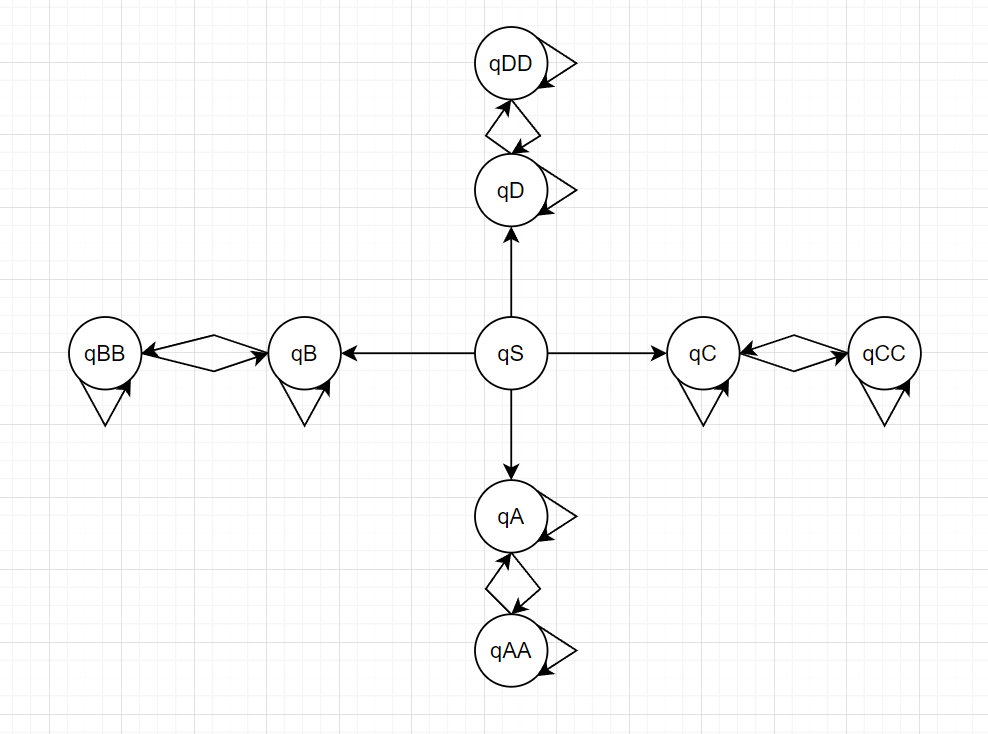
Где qS – стартовое состояние,

qF – финальное состояние,

qA, qAA, qB… – состояния букв (количество), соответственно.

Слово считается недопустимым, если в результате реакции на него автомат не остановится в заключительном состоянии qF.

Из начального состояния автомат переходит в состояния qA, qB, qC, qD. Если конечная буква совпадает с начальной, то переходит в состояния qAA, qBB, qCC, qDD, иначе в одну из букв при первом переходе. Если состояния qS и qF совпадают, значит совпадают начальная и конечная буквы.



*Рисунок 1 – Диаграмма Мура.*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0:S | 1:A | 2:B | 3:C | 4:D | 5:AA | 6:BB | 7:CC | 8:DD | 9:F |
| 0: a | 1 | 5 | 2 | 3 | 4 | 5 | 2 | 3 | 4 | 9 |
| 1: b | 2 | 1 | 6 | 3 | 4 | 1 | 6 | 3 | 4 | 9 |
| 2: c | 3 | 1 | 2 | 7 | 4 | 1 | 2 | 7 | 4 | 9 |
| 3: d | 4 | 1 | 2 | 3 | 8 | 1 | 2 | 3 | 8 | 9 |

*Таблица 1 – таблица переходов.*

Алгоритм работы программы

States - это перечисление всех возможных состояний автомата. Состояние Start представляет начальное состояние, а Final - конечное.

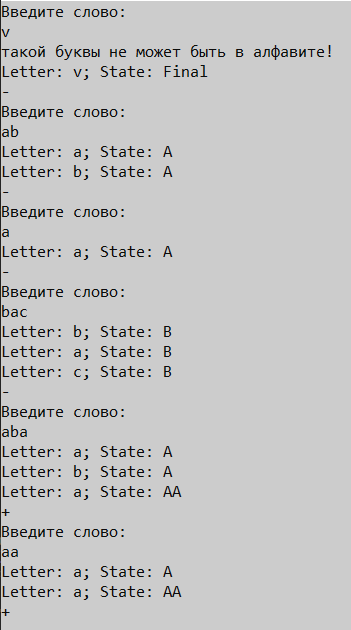
alphabet - массив, содержащий символы, которые могут встречаться во входной строке.

matrix - двумерный массив, представляющий матрицу переходов между состояниями автомата. Каждая строка соответствует символу из alphabet, а каждый столбец соответствует состоянию из States. Элементы матрицы - это номера состояний, в которые переходим из текущего состояния при вводе определенного символа. Например, matrix[0, 1] содержит номер состояния, в которое перейдем, если текущее состояние - Start, а ввод - символ "a".

Transition - это метод, который определяет новое состояние автомата на основе текущего состояния и вводимого символа. Он использует матрицу переходов для выполнения этой операции.

IsWord - это метод, который принимает входную строку и проверяет, является ли она допустимым словом для автомата. Он перебирает каждый символ в строке, последовательно обновляя текущее состояние с помощью метода Transition. Если после обработки всех символов мы оказываемся в состоянии Final, метод возвращает false, иначе - true. Во время обработки символов он выводит текущий символ и состояние для отслеживания процесса работы автомата.

**Результаты работы программы:**



*Рисунок 2 – результат работы программы*

**Листинг программы:**

using System;

public class Program

{

public static void Main()

{

while (true)

{

Console.WriteLine("Введите слово: ");

if (RecognizingAutomaton.IsWord(Console.ReadLine()))

Console.WriteLine("+");

else

Console.WriteLine("-");

}

}

}

public class RecognizingAutomaton

{

private enum States

{

Start,

A,

B,

C,

D,

AA,

BB,

CC,

DD,

Final

}

private static readonly char[] alphabet = { 'a', 'b', 'c', 'd' };

private static readonly int[,] matrix = new int[,]

{

{1, 5, 2, 3, 4, 5, 2, 3, 4, 9},

{2, 1, 6, 3, 4, 1, 6, 3, 4, 9},

{3, 1, 2, 7, 4, 1, 2, 7, 4, 9},

{4, 1, 2, 3, 8, 1, 2, 3, 8, 9}

};

private static States Transition(States currentState, char input)

{

int inputIndex = Array.IndexOf(alphabet, input);

if (inputIndex == -1)

{

Console.WriteLine("такой буквы не может быть в алфавите!");

return States.Final;

}

return (States)matrix[inputIndex, (int)currentState];

}

public static bool IsWord(string word)

{

States state = States.Start;

foreach (char letter in word)

{

state = Transition(state, letter);

Console.WriteLine($"Letter: {letter}; State: {state}");

if (state == States.Final)

break;

}

return state == States.AA || state == States.BB || state == States.CC || state == States.DD;

}

}