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 №8

Выполнил  
студент группы РИС-22-2б  
Зубов Р. А.

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 12:*

|  |
| --- |
| содержащих 3 вхождения символа символов «*с*» и 2 вхождения символа «*d*» |

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, qC, qD, qCC, qDD, qCCС, qDDC, qDCCC, qDDCC, qF}

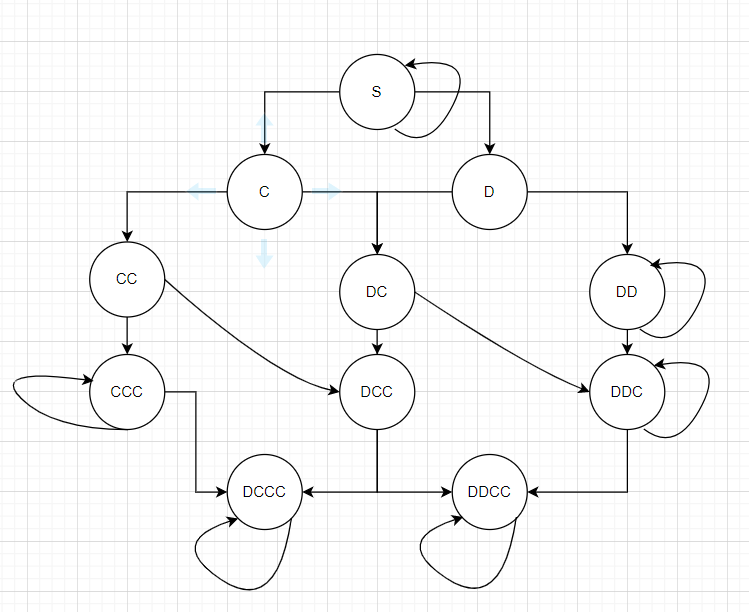
Где qS – стартовое состояние,

qF – финальное состояние,

qC, qD, qCC… – 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 qC, qD. Если встречается буква С или D, то автомат переходит в состояния qC, qCC, qCCС, qD, qDD и т.д., иначе в одну из букв при первом переходе. При встрече отсутствующей буквы в алфавите, автомат выдаст ошибку и перейдет в состояние qF. Если состояния qS и qF совпадают, значит в слове присутствуют ССС и DD.



*Рисунок 1 – Диаграмма Мура.*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0:S | 1:C | 2:D | 3:DC | 4:CC | 5:DD | 6:DCC | 7:DDC | 8:CCC | 9:DCCC | 10:DDCC | 11:F |  |
| 0: a | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |  |
| 1: b | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |  |
| 2: c | 3 | 4 | 3 | 6 | 8 | 7 | 9 | 10 | 8 | 9 | 11 | 11 |  |
| 3: d | 4 | 3 | 5 | 7 | 6 | 5 | 10 | 7 | 9 | 11 | 10 | 11 |  |

*Таблица 1 – таблица переходов.*

Алгоритм работы программы

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

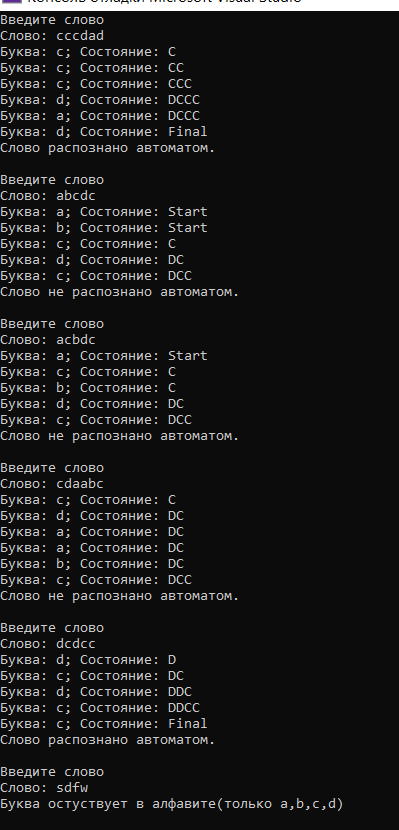
alphabet - массив, содержащий символы, которые могут встречаться во входной строке.

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:**

public class Program

{

static void Main(string[] args)

{

string[] words = { "cccdad", "abcdc", "acbdc", "cdaabc", "dcdcc", "sdfw" };

foreach (string word in words)

{

Console.WriteLine("Введите слово");

Console.WriteLine($"Слово: {word}");

if (IsWord(/\*Console.ReadLine()\*/word))

{

Console.WriteLine("Слово распознано автоматом.");

}

else

{

Console.WriteLine("Слово не распознано автоматом.");

}

Console.WriteLine();

}

}

private enum States

{

Start,

C,

D,

DC,

CC,

DD,

DCC,

DDC,

CCC,

DCCC,

DDCC,

Final

}

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

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

{

{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 },

{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11},

{1, 4, 3, 6, 8, 7, 9, 10, 8, 9, 11, 11 },

{2, 3, 5, 7, 6, 5, 10, 7, 9, 11, 10, 11},

};

private static States Transition(States currentState, char input)

{

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

if (inputIndex == -1)

{

Console.WriteLine("Буква остуствует в алфавите(только a,b,c,d)");

return States.Final;

}

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

}

public static bool IsWord(string word)

{

States state = States.Start;

bool enCountCCC = false;

bool enCountDD = false;

States stateCCC = States.CCC;

States stateDD = States.DD;

foreach (char symbol in word)

{

state = Transition(state, symbol);

Console.WriteLine($"Буква: {symbol}; Состояние: {state}");

if (state == stateCCC || state == States.DCCC || state == States.Final)

{

enCountCCC = true;

}

if (state == stateDD || state == States.DDC || state == States.Final)

{

enCountDD = true;

}

if (state == States.Final)

{

break;

}

}

return enCountCCC && enCountDD;

}

}