Индивидуальное задание №1

Проектирование и реализация конечного распознавателя

Патычкина Елизавета Вадимовна, КТбо1-10

Вариант 1-9

1. **Постановка задачи**

Построить детерминированный конечный автомат, распознающий все такие цепочки из алфавита {0, 1, 2}, в которых каждый из символов встречается нечетное количество раз.

1. **Словесное описание автомата**

Функция machine\_insert инициализирует таблицу переходов. Автомат запрашивает у пользователя входное слово. Функция check проверяет введенное слово на корректность. В случае введения символа, который не содержится в алфавите автомата, он предлагает начать заново работу. В случае, когда слово введено верно, автомат меняет свое состояние в зависимости от текущего состояние и текущего символа из входного слова. Функция СharacterProcessing посимвольно обрабатывает слово.

q0 – начальное состояние, четное количество всех элементов входного алфавита.

q1 – нечетное количество элемента 0, четное количество элементов 1 и 2.

q2 - нечетное количество элемента 1, четное количество элементов 0 и 2.

q3 - нечетное количество элемента 2, четное количество элементов 0 и 1.

q4 - нечетное количество элементов 0 и 1, четное количество элемента 2.

q5 - нечетное количество элементов 0 и 2, четное количество элемента 1.

q6 - нечетное количество элементов 1 и 2, четное количество элемента 0.

q7 - нечетное количество элементов 0, 1 и 2.

В результате автомат выдает строку “Yes”, если конечное состояние является допускающим, или “No”, если конечное состояние не является допускающим. В конце работы автомат предлагает пользователю повторить работу.

1. **Описание распознавателя**

V = {0, 1, 2} – входной алфавит.

K = {q0, q1, q2, q3, q4, q5, q6, q7} – множество состояний.

M – функция переходов. Таблица 1 – таблица переходов автомата. Рисунок 1 – диаграмма переходов автомата.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Состояние  /символ | 0 | 1 | 2 | Y/N |
| q0 | q1 | q2 | q3 | N |
| q1 | q0 | q4 | q5 | N |
| q2 | q4 | q0 | q6 | N |
| q3 | q5 | q6 | q0 | N |
| q4 | q2 | q1 | q7 | N |
| q5 | q3 | q7 | q1 | N |
| q6 | q7 | q3 | q2 | N |
| q7 | q6 | q5 | q4 | Y |

Таблица 1

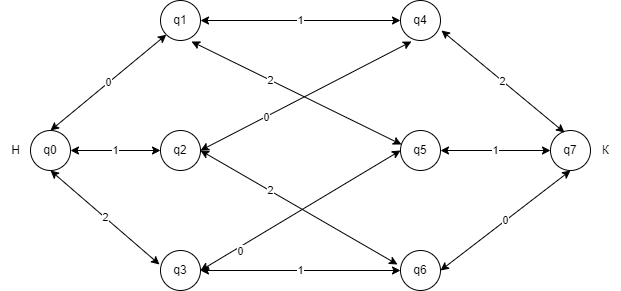


Рисунок 1

S = {q0} – множество начальных состояний.

Z = {q7} – множество заключительных состояний.

1. **Тестирование автомата**
2. 010210122

0 q0 -> q1

1 q1 -> q4

0 q4 -> q2

2 q2 -> q6

1 q6 -> q3

0 q3 -> q5

1 q5 -> q7

2 q7 -> q4

2 q4 -> q7

Yes

1. 012

0 q0 -> q1

1 q1 -> q4

2 q4 -> q7

Yes

1. 0 (1, 2)

0 q0 -> q1

No

1. 123 Строка содержит символы, которые не соответствуют алфавиту автомата
2. 0112010

0 q0 -> q1

1 q1 -> q4

1 q4 -> q1

2 q1 -> q5

0 q5 -> q3

1 q3 -> q6

0 q6 -> q7

Yes

1. 111222

1 q0 -> q2

1 q2 -> q0

1 q0 -> q2

2 q2 -> q6

2 q6 -> q2

2 q2 -> q6

No

1. 01202211

0 q0 -> q1

1 q1 -> q4

2 q4 -> q7

0 q7 -> q6

2 q6 -> q2

2 q2 -> q6

1 q6 -> q3

1 q3 -> q6

No

1. **Прогон автомата на тестах**
2. Тест 1
3. Тест 2
4. Тест 3
5. Тест 4
6. Тест 5
7. Тест 6
8. Тест 7

Изображение выглядит как текст

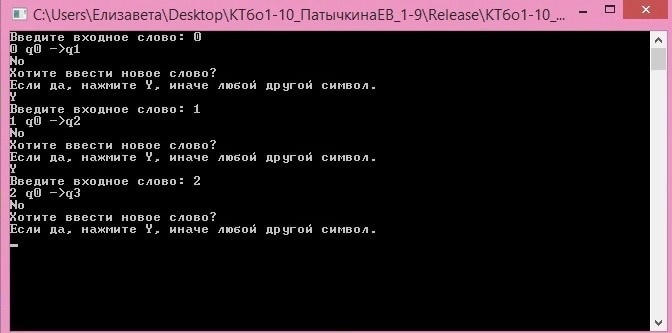
Автоматически созданное описание

Тест 1

Изображение выглядит как текст

Автоматически созданное описание

Тест 2



Тест 3

Изображение выглядит как текст

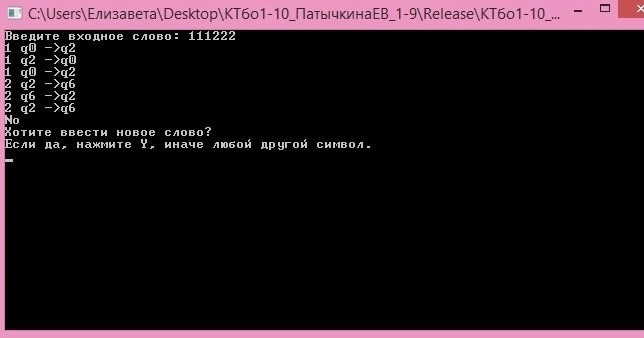
Автоматически созданное описание

Тест 4

Изображение выглядит как текст

Автоматически созданное описание

Тест 5



Тест 6

Изображение выглядит как текст

Автоматически созданное описание

Тест 7

1. **Структура данных для хранения системы переходов автомата**

Для хранения системы переходов автомата используется структура machine. В нее входит таблица переходов (двумерный массив) размерностью 8 × 3. Структура map (состояние и буква, характеризующая допуск или недопуск соответствующему состоянию), хранящая данные о допустимости или не допустимости конкретного состояния автомата. Две целочисленные переменные state\_start и state\_finish. Первая переменная хранит начальное состояние, а вторая конечное состояние при текущем обрабатываемом символе.

1. **Словесное описание идеи программной реализации одного шага работы автомата**

Рассмотрим вводное слово: 010210122

0 q0 -> q1 нечетное количество элемента 0, четное количество элементов 1 и 2

1 q1 -> q4 нечетное количество элементов 0 и 1, четное количество элемента 2

0 q4 -> q2 нечетное количество элемента 1, четное количество элементов 0 и 2

2 q2 -> q6 нечетное количество элементов 1 и 2, четное количество элемента 0

1 q6 -> q3 нечетное количество элемента 2, четное количество элементов 0 и 1

0 q3 -> q5 нечетное количество элементов 0 и 2, четное количество элемента 1

1 q5 -> q7 нечетное количество элементов 0, 1 и 2

2 q7 -> q4 нечетное количество элементов 0 и 1, четное количество элемента 2

2 q4 -> q7 нечетное количество элементов 0, 1 и 2

Заключительное состояние q7 является допускающим, следовательно выводим “Yes”.

1. **Листинг программы**

//ЮФУ, ИКТИБ, МОП ЭВМ

//Программирование и основы теории алгоритмов

//Индивидуальное задание №1

//Проектирование и реализация конечного распознавателя

//КТбо1-10, Патычкина Елизавета Вадимовна

#include <iostream>

#include <vector>

#include <map>

#include <string>

#define row 8

#define column 3

using namespace std;

void machine\_insert(int m[row][column], map <int, char> &YN);

bool check(string line);

int СharacterProcessing(int t[row][column], char c, int state);

struct machine

{

int table[row][column];

map <int, char> YN;

int state\_start = 0;

int state\_finish = 0;

};

int main()

{

setlocale(LC\_ALL, "Russian");

struct machine mymachine;

string answer;

string line;

machine\_insert(mymachine.table, mymachine.YN);

do

{

cout << "Введите входное слово: ";

cin >> line;

if (check(line) == false)

{

cout << "Строка содержит символы, которые не соответствуют алфавиту автомата." << endl;

}

else

{

for (int i = 0; i < line.size(); i++)

{

mymachine.state\_finish = СharacterProcessing(mymachine.table, line[i], mymachine.state\_start);

cout << line[i] << ' ' << 'q' << mymachine.state\_start << ' ' << "->" << 'q' << mymachine.state\_finish << endl;

mymachine.state\_start = mymachine.state\_finish;

}

int FinalState = mymachine.state\_finish;

if (mymachine.YN[FinalState] == 'Y')

cout << "Yes" << endl;

else

cout << "No" << endl;

}

mymachine.state\_start = 0;

cout << "Хотите ввести новое слово?" << '\n' << "Если да, нажмите Y, иначе любой другой символ." << endl;

cin >> answer;

} while (answer == "Y");

}

//Входные параметры: int m[row][column] - таблица переходов

// map <int, char> &YN - ссылка на структуру map с допускающими/не допускающими состояниями

//Функция: заполняет таблицу переходов и структуру допускающими/не допускающими состояниями

//Выходные параметры: функция ничего не возвращает

void machine\_insert(int m[row][column], map <int, char> &YN)

{

m[0][0] = 1;

m[0][1] = 2;

m[0][2] = 3;

m[1][0] = 0;

m[1][1] = 4;

m[1][2] = 5;

m[2][0] = 4;

m[2][1] = 0;

m[2][2] = 6;

m[3][0] = 5;

m[3][1] = 6;

m[3][2] = 0;

m[4][0] = 2;

m[4][1] = 1;

m[4][2] = 7;

m[5][0] = 3;

m[5][1] = 7;

m[5][2] = 1;

m[6][0] = 7;

m[6][1] = 3;

m[6][2] = 2;

m[7][0] = 6;

m[7][1] = 5;

m[7][2] = 4;

YN[0] = 'N';

YN[1] = 'N';

YN[2] = 'N';

YN[3] = 'N';

YN[4] = 'N';

YN[5] = 'N';

YN[6] = 'N';

YN[7] = 'Y';

}

//Входные параметры: string line - введенная пользователем строка(слово)

//Функция: проверяет строку на верность введенных символов

//Выходные параметры: функция возвращает ИСТИНА, если все символы удовлетворяют алфавиту автомата

// возвращает ЛОЖЬ, если хотя бы один символ не удовлетворяют алфавиту автомата

bool check(string line)

{

for (int i = 0; i < line.size(); i++)

{

if ((line[i] == '0') || (line[i] == '1') || (line[i] == '2'))

{

}

else

return false;

}

return true;

}

//Входные параметры: int t[row][column] - таблица переходов

// char c - рассматриваемый символ строки

// int state - текущее начальное состояние

//Функция: обрабатывает один символ входной строки, определяет новое состояние,

// определяет конечное состояние

//Выходные параметры: int t[state][num] - новое конечное состояние

int СharacterProcessing(int t[row][column], char c, int state)

{

int num;

if (c == '0')

num = 0;

else

{

if (c == '1')

num = 1;

else

{

if (c == '2')

num = 2;

}

}

return t[state][num];

}