МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ И.С. ТУРГЕНЕВА»

Кафедра программной инженерии

**Отчет по лабораторной работе**

по дисциплине «Теория языков программирования и методы трансляции»

на тему: «Использование конечного автомата для построения лексического анализатора»

Студенты Логвин А.С. Шифр:160589

Бажин М.И. Шифр:160582

Институт приборостроения, автоматизации и информационных технологий

Специальность 09.03.01 «Информатика и вычислительная техника»

Группа 61-ИВТ

Руководитель Гордиенко А.П.

Орел 2018

**Регулярные выражения:**

Регулярные выражения, описывающие распознаваемые лексемы:

1) Регулярное выражение для *идентификатора*:

L(L|D)\* Space, где L = {A..z}, D = {0..9}, Space – пробел

2) Регулярное выражение для *числа*:

(E| '-' )(D+)(E|('.'|',')D+)Space

3) Регулярное выражение для *присвоения*:

':''='Space (конкатенация)

4) Регулярное выражение для *знака операции*:

'-'Space (конкатенация)

'+'Space (конкатенация)

5) Для распознания *служебных слов* используется массив значений:

[Begin, begin, End, end]

**Работа с автоматами для «Числа»**

1) Недетерминированный конечный автомат (Рисунок 1):

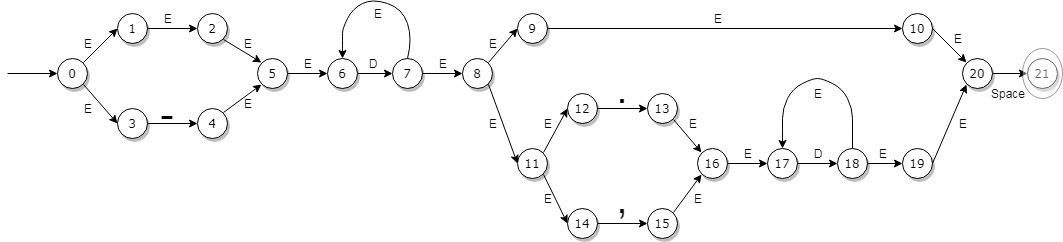


Рисунок 1 – НКА

2) Таблица переходов для приведения к ДКА:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | \_ | . | , | D | Space |
| 1) {0,1,2,3,5,6} | {4,5,6} |  |  | {6,7,8,9,10,11  12,14,20} |  |
| 2) {4,5,6} |  |  |  | {6,7,8,9,10,11  12,14,20} |  |

Продолжение таблицы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3) {6,7,8,9,10,  11,12,14,20} |  | {13,16,17} | {15,16,17} | {6,7,8,9,10,11  12,14,20} | {21} |
| 4) {13,16,17} |  |  |  | {17,18,19,20} |  |
| 5) {15,16,17} |  |  |  | {17,18,19,20} |  |
| 6) {17,18,19,20} |  |  |  | {17,18,19,20} | {21} |
| 7) {21} |  |  |  |  |  |

Таблица 1 – таблица переходов состояний автомата

3) Детерминированный конечный автомат (Рисунок 2):

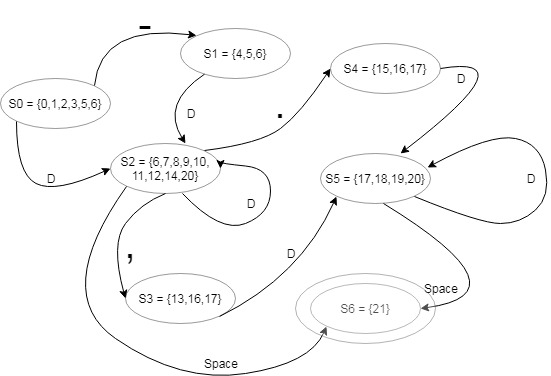


Рисунок 2 – ДКА

4) Минимизация ДКА:

Нефинальные состояния: {S0,S1,S2,S3,S4,S5}

Финальные состояния: {S6}

После разбиения на группы:

0 => {S0} 3 => {S3,S4}

1 => {S1} 4 => {S5}

2 => {S2} 5 => {S6}

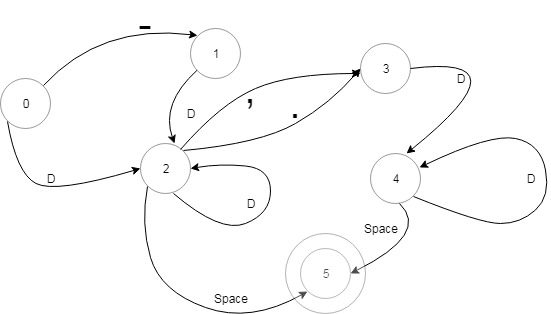


Рисунок 3 – Минимизированный ДКА

**Конечный автомат для распознавания лексем**

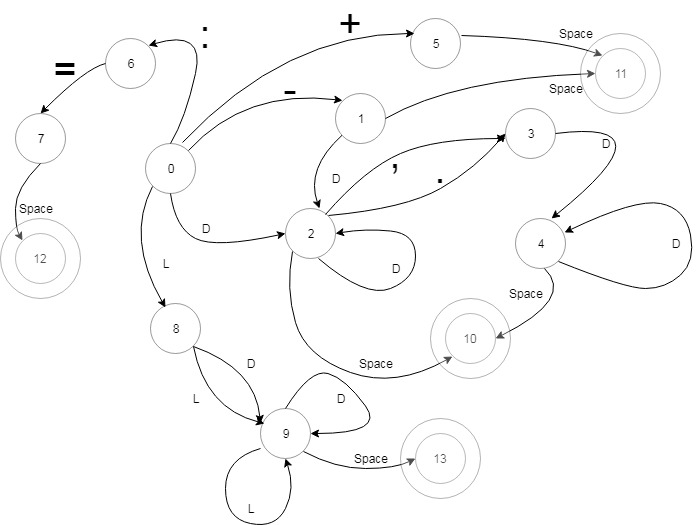


Рисунок 4 – Конечный автомат для распознавания заданных лексем

Листинг программы:

unit Unit1;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, FileUtil, Forms, Controls, Graphics, Dialogs, StdCtrls;

type

input\_signal = (plus,dash,dot,com,L,D,colon,equality,space,other,endf);

state = (S0,S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11,S12,S13,S\_er);

lexeme\_class = (digit,identifier,assignment,operation\_mark,service\_word);

{ TForm1 }

TForm1 = class(TForm)

Button1: TButton;

Edit1: TEdit;

Label1: TLabel;

Label2: TLabel;

Memo1: TMemo;

function state\_machine: lexeme\_class;

function recognize: input\_signal;

function is\_service\_word: lexeme\_class;

procedure Button1Click(Sender: TObject);

private

{ private declarations }

public

{ public declarations }

end;

const

length\_array = 3;

service\_word\_array: array[0..length\_array] of string = ('Begin','begin','End','end');

next\_state: array[S0..S9,plus..other] of state =

((S5,S1,S\_er,S\_er,S8,S2,S6,S\_er,S\_er,S\_er),

(S\_er,S\_er,S\_er,S\_er,S\_er,S2,S\_er,S\_er,S11,S\_er),

(S\_er,S\_er,S3,S3,S\_er,S2,S\_er,S\_er,S10,S\_er),

(S\_er,S\_er,S\_er,S\_er,S\_er,S4,S\_er,S\_er,S\_er,S\_er),

(S\_er,S\_er,S\_er,S\_er,S\_er,S4,S\_er,S\_er,S10,S\_er),

(S\_er,S\_er,S\_er,S\_er,S\_er,S\_er,S\_er,S\_er,S11,S\_er),

(S\_er,S\_er,S\_er,S\_er,S\_er,S\_er,S\_er,S7,S\_er,S\_er),

(S\_er,S\_er,S\_er,S\_er,S\_er,S\_er,S\_er,S\_er,S12,S\_er),

(S\_er,S\_er,S\_er,S\_er,S9,S9,S\_er,S\_er,S13,S\_er),

(S\_er,S\_er,S\_er,S\_er,S9,S9,S\_er,S\_er,S13,S\_er));

var

Form1: TForm1;

entry, lex\_val: String;

implementation

{$R \*.lfm}

{ TForm1 }

function TForm1.state\_machine: lexeme\_class;

var cur\_state: state; cur\_input: input\_signal;

begin

lex\_val:=''; cur\_state:=S0; cur\_input:=recognize;

while (cur\_state <> S10) and (cur\_state <> S11) and (cur\_state <> S12) and (cur\_state <> S13) and (cur\_input <> endf) do

begin

cur\_state:=next\_state[cur\_state,cur\_input];

if cur\_state = S\_er

then raise exception.create('Лексическая ошибка!');

if (cur\_state <> S10) and (cur\_state <> S11) and (cur\_state <> S12) and (cur\_state <> S13) then cur\_input:=recognize;

end;

if (cur\_state <> S10) and (cur\_state <> S11) and (cur\_state <> S12) and (cur\_state <> S13) and (cur\_input <> endf)

then raise exception.create('Лексическая ошибка!')

else

case cur\_state of

S10: result:=digit;

S11: result:=operation\_mark;

S12: result:=assignment;

S13: result:=is\_service\_word;

end;

end;

function TForm1.recognize: input\_signal;

begin

if entry = '' then result:=endf

else

begin

case entry[1] of

'+': result:=plus;

'-': result:=dash;

'.': result:=dot;

',': result:=com;

':': result:=colon;

'=': result:=equality;

' ': result:=space;

'0'..'9': result:=D;

'a'..'z': result:=L;

'A'..'Z': result:=L;

else result:=other;

end;

lex\_val:=lex\_val+entry[1];

entry:=copy(entry,2,length(entry)-1);

end;

end;

function TForm1.is\_service\_word: lexeme\_class;

var i: integer;

lex: string;

service\_word\_flag: boolean;

begin

i:=0;

lex:=TrimRight(lex\_val);

service\_word\_flag:=false;

while (not service\_word\_flag) and (i <= length\_array) do

begin

if lex = service\_word\_array[i] then service\_word\_flag:=true;

inc(i);

end;

if service\_word\_flag

then result:=service\_word

else result:=identifier;

end;

procedure TForm1.Button1Click(Sender: TObject);

begin

entry:=TrimRight(edit1.Text) + ' ';

while (entry <> '') do

begin

case state\_machine of

digit: Memo1.Lines.Add('Цифра: ' + lex\_val);

identifier: Memo1.Lines.Add('Идентификатор: ' + lex\_val);

assignment: Memo1.Lines.Add('Присвоение: ' + lex\_val);

operation\_mark: Memo1.Lines.Add('Знак операции: ' + lex\_val);

service\_word: Memo1.Lines.Add('Служебное слово: ' + lex\_val);

end;

end;

Memo1.Lines.Add('----------------------------------');

end;

end.