МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**
**(ФГБОУ ВО «КУБГУ»)**
Кафедра вычислительных технологий

**Отчёт**

**по лабораторной работе №1 по курсу**

**«Методы разработки трансляторов»**

**на тему: «Построение лексического анализатора»**

Работу выполнил

Студент 36 группы

Усов П.Е.

_____

*(подпись)*

Преподаватель:

Вишняков Ю.М.

_____

*(подпись)*

Краснодар 2019

**Цель работы.** Требуется построить лексический анализатор заданного подмножества языковых конструкций и операторов входного языка программирования – языка Pascal.

**Задание.** Необходимо построить программу лексического анализатора, корректно распознающего все заданные лексемы и формирующего таблицы лексем и внутреннее представление проанализированного текста. На вход программы подаётся файл, содержащий текст программы на входном языке программирования – языке Pascal. Результатом работы программы должен быть файл, содержащий последовательность кодов лексем входной программы, а также один или несколько файлов, содержащие все таблицы лексем.

## Ход работы:

### 1 Описание подмножества языковых конструкций и операторов входного языка

*Идентификаторы* в языке Pascal представляют собой произвольные последовательности букв и цифр, начинающиеся с буквы. Могут включать символы подчёркивания.

*Числовые константы целого типа* представляют собой произвольные последовательности цифр без знака.

*Числовые константы вещественного типа с фиксированной точкой* представляют собой последовательности цифр, включающие одну десятичную точку. Примеры: 123.45; .25

*Числовые константы вещественного типа с плавающей точкой* представляют собой последовательности, включающие цифры, десятичную точку (необязательную), символ «e» или «E», а также знак «+» или «-» (необязательный). Примеры: 1.23e-25; 1.23E-25; 1.23e+25; 1.23E+25; 1.23e2; 1.23E2; 1E-78; 1e67.

*Символьные (строковые) константы* представляют собой последовательности символов, заключённые в апострофы, расположенные в пределах одной строки. Пример: ‘acb 12_& ?tu’.

*Переменные с индексами (массивы и элементы массивов)* представляют собой идентификаторы, после которых в квадратных скобках через запятую перечислены выражения-индексы. Примеры: Abc[12, I, i-6]; C[1+i].

*Комментарии – только блочные* - последовательность символов, заключенная в фигурные скобки, возможно содержащая несколько строк. Пример:

{ Это комментарий,

Который содержит 2 строки}

*Обращение к процедурам и функциям пользователя* - идентификатор, после которого в круглых скобках следует последовательность выражений-аргументов, разделённых запятыми. Отсутствие аргументов не допускается. Примеры: F(12, 4, i); f(av-6).

*Арифметические операции:* сложение («+»); вычитание (“-“); умножение («*»); деление («/»); возведение в степень («^»).

*Операции сравнения:* меньше («<»); больше («>»); равно («=»); не равно («<>»); меньше или равно («<=»); больше или равно («>=»).

*Оператор присваивания* имеет вид «:=». Слева стоит идентификатор или элемент массива, а справа – выражение. Заканчивается символом «;». Примеры: a:=b+c; b[2,i-9]:=12;

*Операторы блока:*

Begin – начало блока

…

End; - конец блока

*Оператор описания программы.* Программа начинается оператором Program с указанием имени программы. Затем могут идти описания данных,

процедур и функций, а затем тело программы, заключённое в операторы блока, оканчивающееся точкой. Пример:

Program <идентификатор>;

…

Begin

…

End.

*Оператор описания данных (идентификаторов и массивов)* начинается оператором Var и может содержать несколько строк описаний, состоящих из перечисления идентификаторов через запятую и названия типа этих идентификаторов после двоеточия. Пример: Var A,b: real; C: integer;

*Типы переменных:* integer (целый), real (вещественный), string (строковый)

Для массивов после двоеточия указывается ключевое слово массива «array of», в квадратных скобках через запятую перечисляются границы изменения каждого из индексов, разделённые символами «..», и затем тип элементов. Пример: Var a,b,c : array of [1..3, 10..20] of integer;

*Операторы описания процедур и функций.* Процедуры имеют заголовок вида

procedure <идентификатор> (<список формальных параметров>);

тело – список операторов, заключённый в операторы блока begin … end;

Между заголовком и телом может присутствовать оператор описания данных Var. Пример:

```
procedure abc (r: real);
var r1,r2:real;
begin
y:=sinr(r1)/cos(r2)*tan(r);
end;
```

Функции имеют заголовок вида

function <идентификатор> (<список формальных параметров>): <тип возвращаемого значения>;

В остальном структура функций аналогична структуре процедур. Исключение составляет обязательное присутствие в теле функции хотя бы одного оператора return <значение>;

*Оператор безусловного перехода и метки* имеет вид goto <метка>;

Метка - идентификатор, расположенный в теле программы в начале строки, после которого стоит знак «:». Пример:

a: str:="ujhti";

*Оператор условного перехода* начинается с ключевого слова «if», имеет полный и неполный формат. Примеры: If <условие> then <оператор_1> else <оператор_2>; If <условие> then <оператор_1>;

## 2 Таблицы, используемые лексическим анализатором

*Таблица служебных слов:*

| № | Слово |
|---|---|
| 1 | Program |
| 2 | Begin |
| 3 | End |
| 4 | Var |
| 5 | integer |
| 6 | real |
| 7 | string |
| 8 | array |
| 9 | of |
| 10 | procedure |
| 11 | function |
| 12 | return |
| 13 | goto |

| | |
|---|---|
| 14 | if |
| 15 | then |
| 16 | else |
| 17 | not |
| 18 | and |
| 19 | or |
| 20 | xor |
| 21 | Read |
| 22 | Write |
| 23 | boolean |
| 24 | while |
| 25 | repeat |
| 26 | until |
| 27 | do |
| 28 | to |
| 29 | longint |
| 30 | true |
| 31 | false |
| 32 | for |
| 33 | sqrt |
| 34 | div |
| 35 | mod |
| 36 | sqr |

*Таблица разделителей:*

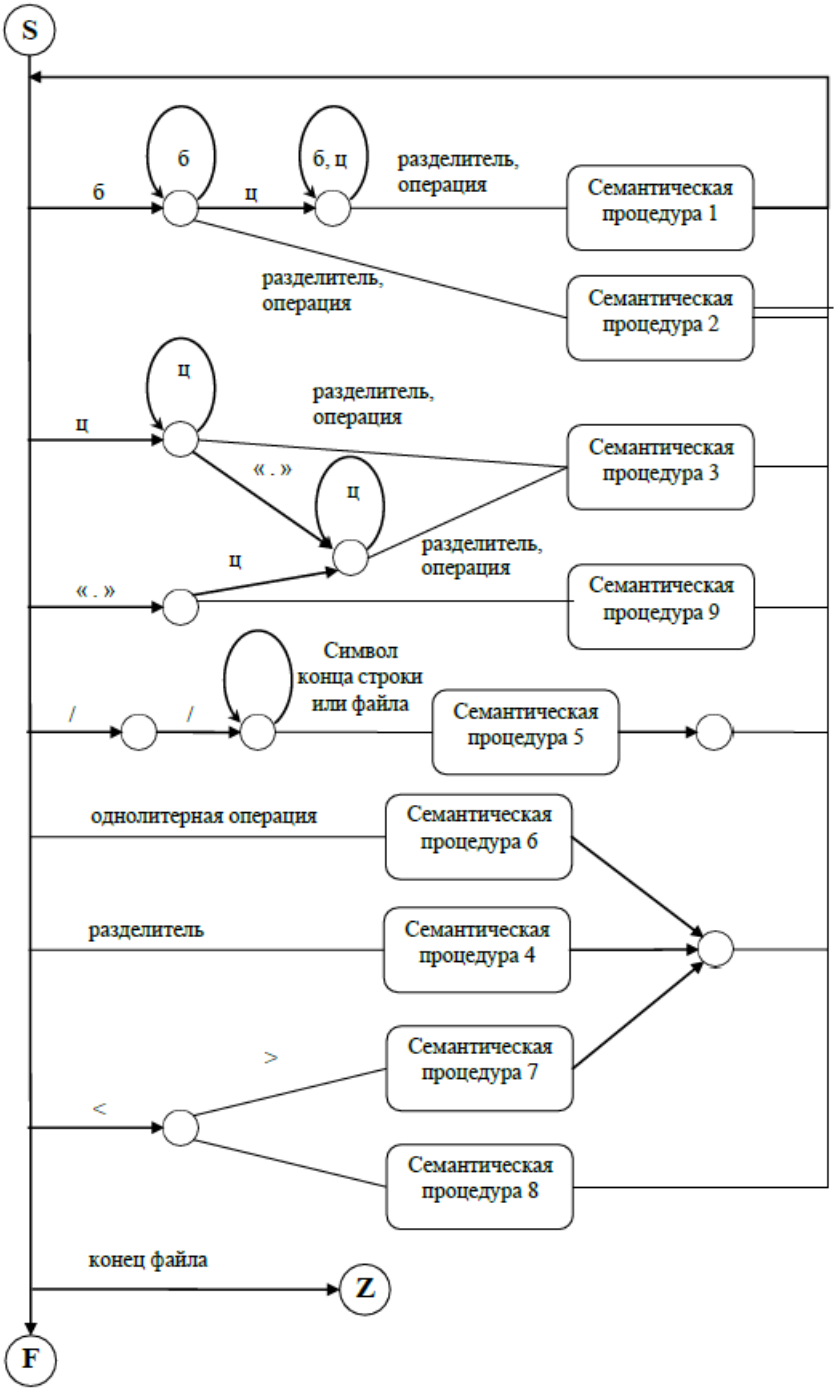
| № | Разделитель |
|---|---|
| 1 | . |
| 2 | ‘ |
| 3 | [ |
| 4 | ] |

| | |
|---|---|
| 5 | , |
| 6 | { |
| 7 | } |
| 8 | ( |
| 9 | ) |
| 10 | ; |
| 11 | “ “ (пробел) |
| 12 | “\r” (возврат каретки) |
| 13 | : |
| 14 | “\n” (переход на новую строку) |
| 15 | “\b” (возврат на одну позицию) |
| 16 | “\t” (горизонтальная табуляция) |
| 17 | \ |
| 18 | “ |

*Таблица операций:*

| № | Операция |
|---|---|
| 1 | + |
| 2 | - |
| 3 | * |
| 4 | / |
| 5 | ^ |
| 6 | < |
| 7 | > |
| 8 | = |
| 9 | >= |
| 10 | <= |
| 11 | <> |
| 12 | := |

## 3 Диаграмма состояний сканера




На данной диаграмме состояния сканера представлены вершинами, а переходы между состояниями - дугами (направленными линиями). Каждый переход связан с чтением очередного символа их текста входной программы. Поэтому дуга взвешена (помечена) символом или множеством символов, которые вызывают данный переход. Если в диаграмме состояний есть невзвешенная дуга, ведущая из какого-либо состояния, то считается, что она взвешена любыми символами кроме тех, которыми взвешены другие дуги, исходящие из данного

состояния. Это позволяет не перегружать диаграмму лишними символами. Скруглённый прямоугольник в разрыве дуги указывает на семантическую процедуру, выполняемую при данном переходе. Если переход не сопровождается семантической процедурой, то текущий символ добавляется к буферу, в котором формируется лексема. Работа каждой семантической процедуры завершается очищением буфера.

Сканер всегда содержит три стандартных состояния:

- S – начальное состояние сканера. Лексический разбор всегда начинается из этого состояния;

- Z – заключительное состояние сканера. Если в процессе разбора достигнуто данное состояние, это означает, что разбор успешно завершен;

- F – состояние ошибки. Если встретился символ, не входящий во входной алфавит, то сканер переходит в состояние ошибки и прекращает разбор. Кроме того, если в каком-либо состоянии на входе появился символ, по которому не предусмотрен переход из этого состояния, сканер также переходит в состояние ошибки (на диаграмме эти переходы не изображаются, чтобы избежать лишнего загромождения рисунка).

За семантическими процедурами процессора закреплены следующие функции.

Семантическая процедура 1: провести поиск сформированного слова в таблице идентификаторов. Если такое слово в таблице идентификаторов не найдено, то занести сформированное слово в таблицу идентификаторов. Сформировать и выдать в выходную последовательность лексему идентификатора во внутреннем представлении.

Семантическая процедура 2: провести поиск сформированного слова в таблице служебных слов. Если такое слово в таблице служебных слов не найдено, то выполнить Семантическую процедуру 1, иначе сформировать и выдать в выходную последовательность лексему служебного слова во внутреннем представлении.

Семантическая процедура 3: занести сформированное слово в таблицу констант. Сформировать и выдать в выходную последовательность лексему константы во внутреннем представлении.

Семантическая процедура 4: провести поиск текущего символа в таблице разделителей. Сформировать и выдать в выходную последовательность лексему разделителя во внутреннем представлении.

Семантическая процедура 5: удалить сформированную последовательность символов.

Семантическая процедура 6: провести поиск текущего символа в таблице операций. Сформировать и выдать в выходную последовательность лексему операции во внутреннем представлении.

Семантическая процедура 7: добавить текущий символ к сформированному слову. Провести поиск сформированного слова в таблице операций. Если такое слово в таблице операций не найдено, то перейти в состояние ошибки. Если поиск успешен, то сформировать и выдать в выходную последовательность лексему операции во внутреннем представлении.

Семантическая процедура 8: провести поиск сформированного слова в таблице операций. Сформировать и выдать в выходную последовательность лексему операции во внутреннем представлении.

Семантическая процедура 9: провести поиск сформированного слова в таблице разделителей. Сформировать и выдать в выходную последовательность лексему разделителя во внутреннем представлении.

## 4 Листинг программы

Лексический анализатор был разработан на языке C#. Ниже представлен текст программы.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using System.Globalization;
namespace WindowsFormsApplication2
{

    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
            public struct lexema
            {
                public int id;
                public string s;
            };
            static FileStream g = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\output.txt",
FileMode.Create, FileAccess.Write);
            static StreamWriter writer = new StreamWriter(g);
            static FileStream lex = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\lexemy.txt",
FileMode.Create, FileAccess.Write);
            static StreamWriter writer_lex = new StreamWriter(lex);
            static lexema[] W = new lexema[36]; //массив служебных слов
            static lexema[] R = new lexema[19]; //массив разделителей
            static lexema[] O = new lexema[12]; //массив операций
            static lexema[] I = new lexema[1000]; //массив идентификаторов
            static lexema[] N = new lexema[1000]; //массив чисел
            static bool err = false; //переменная наличия ошибки
            static void er() //процедура-ошибка
            {
                err = true;
            }
            static void init() //инициализация таблиц лексем
            {
                W[0].id = 1; W[0].s = "Program";
                W[1].id = 2; W[1].s = "Begin";
                W[2].id = 3; W[2].s = "End";
                W[3].id = 4; W[3].s = "Var";
                W[4].id = 5; W[4].s = "integer";
                W[5].id = 6; W[5].s = "real";
                W[6].id = 7; W[6].s = "string";
                W[7].id = 8; W[7].s = "array";
                W[8].id = 9; W[8].s = "of";
                W[9].id = 10; W[9].s = "procedure";
                W[10].id = 11; W[10].s = "function";
```

```
W[11].id = 12; W[11].s = "return";
W[12].id = 13; W[12].s = "goto";
W[13].id = 14; W[13].s = "if";
W[14].id = 15; W[14].s = "then";
W[15].id = 16; W[15].s = "else";
W[16].id = 17; W[16].s = "not";
W[17].id = 18; W[17].s = "and";
W[18].id = 19; W[18].s = "or";
W[19].id = 20; W[19].s = "xor";
W[20].id = 21; W[20].s = "Read";
W[21].id = 22; W[21].s = "Write";
W[22].id = 23; W[22].s = "boolean";
W[23].id = 24; W[23].s = "while";
W[24].id = 25; W[24].s = "repeat";
W[25].id = 26; W[25].s = "until";
W[26].id = 27; W[26].s = "do";
W[27].id = 28; W[27].s = "to";
W[28].id = 29; W[28].s = "longint";
W[29].id = 30; W[29].s = "true";
W[30].id = 31; W[30].s = "false";
W[31].id = 32; W[31].s = "for";
W[32].id = 33; W[32].s = "sqrt";
W[33].id = 34; W[33].s = "div";
W[34].id = 35; W[34].s = "mod";
W[35].id = 36; W[35].s = "sqr";
R[0].id = 1; R[0].s = ".";
R[1].id = 2; R[1].s = "'";
R[2].id = 3; R[2].s = "[";
R[3].id = 4; R[3].s = "]";
R[4].id = 5; R[4].s = ",";
R[5].id = 6; R[5].s = "{";
R[6].id = 7; R[6].s = "}";
R[7].id = 8; R[7].s = "(";
R[8].id = 9; R[8].s = ")";
R[9].id = 10; R[9].s = ";";
R[10].id = 11; R[10].s = " ";
R[11].id = 12; R[11].s = "\r"; //возврат каретки
R[12].id = 13; R[12].s = ":";
R[13].id = 14; R[13].s = "\n"; //новая строка
R[14].id = 15; R[14].s = "\b"; //возврат на 1 позицию
R[15].id = 16; R[15].s = "\t"; //горизонтальная табуляция
R[16].id = 17; R[16].s = "\\";
R[17].id = 18; R[17].s = "\"";
R[18].id = 19; R[18].s = "..";
O[0].id = 1; O[0].s = "+";
O[1].id = 2; O[1].s = "-";
O[2].id = 3; O[2].s = "*";
O[3].id = 4; O[3].s = "/";
O[4].id = 5; O[4].s = "^";
O[5].id = 6; O[5].s = "<";
O[6].id = 7; O[6].s = ">";
```

```
    O[7].id = 8; O[7].s = "=";
    O[8].id = 9; O[8].s = ">=";
    O[9].id = 10; O[9].s = "<=";
    O[10].id = 11; O[10].s = "<>";
    O[11].id = 12; O[11].s = ":=";
}
static void p1(string s) //семантическая процедура 1
{
    init();
    int i = 0; bool f = false; int j = 0;
    while ((I[i].s != null) && (i < 1000) && (!f))
    {
        if (I[i].s == s)
        {
            f = true; j = i;
        }
        else
        {
            i++;
        }
    }
    if (!f)
    {
        j = 0;
        while ((I[j].s != null) && (j < 1000))
        {
            j++;
        }
        I[j].id = j + 1; I[j].s = s;
    }
    writer.WriteLine("I" + I[j].id.ToString());
}
static void p2(string s) //семантическая процедура 2
{
    init(); bool f = false; int i = 0; int j = 0;
    while ((!f) && (i < 36))
    {
        if (W[i].s == s)
        {
            f = true; j = W[i].id;
        }
        else
        {
            i++;
        }
    }
    if (f)
    {
        writer.WriteLine("W" + j.ToString());
    }
    else
```

```
    {
        p1(s);
    }
}
static void p3(string s) //семантическая процедура 3
{
    init(); int i = 0; bool f = false; int j = 0;
    while ((N[i].s != null) && (i < 1000) && (!f))
    {
        if (N[i].s == s) { f = true; j = i; }
        else { i++; }
    }
    if (!f)
    {
        j = 0;
        while ((N[j].s != null) && (j < 1000))
        {
            j++;
        }
        N[j].id = j + 1; N[j].s = s;
    }
    writer.WriteLine("N" + N[j].id.ToString());
}
static void p4(string s) //семантическая процедура 4
{
    init(); int i = 0; bool f = false; int j = 0;
    while ((!f) && (i < 19))
    {
        if (R[i].s == s)
        {
            f = true; j = R[i].id;
        }
        else
        {
            i++;
        }
    }
    if (f)
    {
        writer.WriteLine("R" + j.ToString());
    }
}
static void p5(string s) //семантическая процедура 5
{
    init(); s = "";
}
static void p6(string s) //семантическая процедура 6
{
    init(); int i = 0; bool f = false; int j = 0;
    while ((!f) && (i < 12))
    {
```

```
                if (O[i].s == s)
                {
                    f = true; j = O[i].id;
                }
                else
                {
                    i++;
                }
            }
            if (f)
            {
                writer.WriteLine("O" + j.ToString());
            }
        }
        static void p7(string s, string c) //семантическая процедура 7
        {
            string st = s + c;
            init(); int i = 0; bool f = false; int j = 0;
            while ((!f) && (i < 12))
            {
                if (O[i].s == st)
                {
                    f = true; j = O[i].id;
                }
                else
                {
                    i++;
                }
            }
            if (f)
            {
                writer.WriteLine("O" + j.ToString());
            }
            else
            {
                er();
            }
        }
        static void p8(string s) //семантическая процедура 8
        {
            p6(s);
        }
        static void p9(string s) //семантическая процедура 9
        {
            p4(s);
        }
        static void S(string s)
        {
            int i = 0; err = false; init(); string buf = "";
if(s==””){er();}
            while ((i < s.Length) && (!err))
```

```
                {
                    if ((i < s.Length) && ((s[i] >= 'a') && (s[i] <= 'z')
|| (s[i] >= 'A') && (s[i] <= 'Z') || (s[i] == '_')))
                    {
                        while ((i < s.Length) && ((s[i] >= 'a') && (s[i] <=
'z') || (s[i] >= 'A') && (s[i] <= 'Z') || (s[i] == '_')))
                        {
                            buf = buf + s[i]; i++;
                        }
                        if ((i < s.Length) && (s[i] >= '0') && (s[i] <=
'9'))
                        {
                            while ((i < s.Length) && ((s[i] >= 'a') &&
(s[i] <= 'z') || (s[i] >= 'A') && (s[i] <= 'Z') || (s[i] >= '0') && (s[i]
<= '9') || (s[i] == '_')))
                            {
                                buf = buf + s[i]; i++;
                            }
                            string c = ""; if (i < s.Length) { c = "" +
s[i]; }
                            bool f = false; int j = 0; int k = 0;
                            while ((!f) && (j < 19))
                            {
                                if (R[j].s == c) { f = true; }
                                else { j++; }
                            }
                            while ((!f) && (k < 12))
                            {
                                if (O[k].s == c) { f = true; }
                                else { k++; }
                            }
                            if (f) { p1(buf); Console.WriteLine(buf);
writer_lex.WriteLine(buf); buf = ""; }
                            else { er(); }
                        }
                        else
                        {
                            string c = ""; if (i < s.Length) { c = "" +
s[i]; }
                            bool f = false; int j = 0; int k = 0;
                            while ((!f) && (j < 19))
                            {
                                if (R[j].s == c) { f = true; }
                                else { j++; }
                            }
                            while ((!f) && (k < 12))
                            {
                                if (O[k].s == c) { f = true; }
                                else { k++; }
                            }
```

```
                            if (f) { p2(buf); Console.WriteLine(buf);
writer_lex.WriteLine(buf); buf = ""; }
                            else { er(); }
                        }
                    }
                    else
                    {
                        if ((i < s.Length) && (s[i] >= '0') && (s[i] <=
'9'))
                        {
                            while ((i < s.Length) && (s[i] >= '0') && (s[i]
<= '9'))
                            {
                                buf = buf + s[i]; i++;
                            }
                            if ((i < s.Length) && (s[i] == '.'))
                            {
                                i++; string c2 = "";
                                while ((i < s.Length) && (s[i] >= '0') &&
(s[i] <= '9'))
                                {
                                    c2 = c2 + s[i]; i++;
                                }
                                if (c2 != "")
                                {
                                    if ((i < s.Length) && ((s[i] == 'e') ||
(s[i] == 'E')))
                                    {
                                        if ((i < s.Length - 1) && ((s[i +
1] == '+') || (s[i + 1] == '-')))
                                        {
                                            if ((i < s.Length - 2) && (s[i
+ 2] >= '0') && (s[i + 2] <= '9'))
                                            {
                                                string c4 = ""; int j4 = i
+ 2;
                                                while ((j4 < s.Length) &&
(s[j4] >= '0') && (s[j4] <= '9'))
                                                {
                                                    c4 = c4 + s[j4]; j4++;
                                                }
                                                if (c4 != "")
                                                {
                                                    c2 = c2 + s[i] + s[i +
1] + c4;
                                                    i = j4;
                                                    string c5 = ""; if (i <
s.Length) { c5 = "" + s[i]; }
                                                    bool f5 = false; int j5
= 0; int k5 = 0;
```

```
                                                    while ((!f5) && (j5 <
19))
                                                    {
                                                        if ((R[j5].s == c5)
&& (c5 != ".")) { f5 = true; }
                                                        else { j5++; }
                                                    }
                                                    while ((!f5) && (k5 <
12))
                                                    {
                                                        if (O[k5].s == c5)
{ f5 = true; }
                                                        else { k5++; }
                                                    }
                                                    if (!f5) { er(); }
                                                }
                                            }
                                            else { er(); }
                                        }
                                        else
                                        {
                                            if ((i < s.Length - 1) && (s[i
+ 1] >= '0') && (s[i + 1] <= '9'))
                                            {
                                                string c4 = ""; int j4 = i
+ 1;
                                                while ((j4 < s.Length) &&
(s[j4] >= '0') && (s[j4] <= '9'))
                                                {
                                                    c4 = c4 + s[j4]; j4++;
                                                }
                                                if (c4 != "")
                                                {
                                                    c2 = c2 + s[i] + c4;
                                                    i = j4;
                                                    string c5 = ""; if (i <
s.Length) { c5 = "" + s[i]; }
                                                    bool f5 = false; int j5
= 0; int k5 = 0;
                                                    while ((!f5) && (j5 <
19))
                                                    {
                                                        if ((R[j5].s == c5)
&& (c5 != ".")) { f5 = true; }
                                                        else { j5++; }
                                                    }
                                                    while ((!f5) && (k5 <
12))
                                                    {
                                                        if (O[k5].s == c5)
{ f5 = true; }
```

```
                                                    else { k5++; }
                                                }
                                                if (!f5) { er(); }
                                            }
                                        }
                                        else { er(); }
                                    }
                                }
                            }
                            if (c2 != "")
                            {
                                buf = buf + "." + c2;
                                string c3 = ""; if (i < s.Length) { c3
= "" + s[i]; }
                                bool f3 = false; int j3 = 0; int k3 =
0;
                                while ((!f3) && (j3 < 19))
                                {
                                    if ((R[j3].s == c3) && (c3 != "."))
{ f3 = true; }
                                    else { j3++; }
                                }
                                while ((!f3) && (k3 < 12))
                                {
                                    if (O[k3].s == c3) { f3 = true; }
                                    else { k3++; }
                                }
                                if (f3) { p3(buf);
Console.WriteLine(buf); writer_lex.WriteLine(buf); buf = ""; }
                                else { er(); }
                            }
                            else
                            {
                                p3(buf); Console.WriteLine(buf);
writer_lex.WriteLine(buf); buf = "";
                            }

                        }
                        else
                        {
                            if ((i < s.Length) && ((s[i] == 'e') ||
(s[i] == 'E')))
                            {
                                if ((i < s.Length - 1) && ((s[i + 1] ==
'+') || (s[i + 1] == '-')))
                                {
                                    if ((i < s.Length - 2) && (s[i + 2]
>= '0') && (s[i + 2] <= '9'))
                                    {
                                        int j6 = i + 2; buf = buf +
s[i] + s[i + 1];
```

```
                                                while ((j6 < s.Length) &&
(s[j6] >= '0') && (s[j6] <= '9'))
                                                {
                                                    buf = buf + s[j6]; j6++;
                                                }
                                                p3(buf);
Console.WriteLine(buf); writer_lex.WriteLine(buf); buf = "";
                                                i = j6; int j7 = 0; int k7 = 0;
bool f7 = false; string c7 = ""; string c8 = ""; bool g7 = false;
                                                if (i < s.Length) { c7 = "" +
s[i]; }
                                                if (i < s.Length - 1) { c8 = c8
+ s[i] + s[i + 1]; }
                                                while ((!g7) && (k7 < 12))
                                                {
                                                    if (O[k7].s == c8) { g7 =
true; }
                                                    else { k7++; }
                                                }
                                                if (g7) { p6(c8);
Console.WriteLine(c8); writer_lex.WriteLine(c8); c8 = ""; i += 2; }
                                                else
                                                {
                                                    k7 = 0;
                                                    while ((!f7) && (j7 < 19))
                                                    {
                                                        if ((R[j7].s == c7) &&
(c7 != ".")) { f7 = true; }
                                                        else { j7++; }
                                                    }
                                                    if (f7) { p4(c7);
Console.WriteLine(c7); writer_lex.WriteLine(c7); c7 = ""; i++; }
                                                    else
                                                    {
                                                        while ((!f7) && (k7 <
12))
                                                        {
                                                            if (O[k7].s == c7)
{ f7 = true; }
                                                            else { k7++; }
                                                        }
                                                        if (!f7) { er(); }
                                                        else { p6(c7);
Console.WriteLine(c7); writer_lex.WriteLine(c7); c7 = ""; i++; }
                                                    }
                                                }
                                            }
                                            else { er(); }
                                        }
                                        else
                                        {
```

```
                                        if ((i < s.Length - 1) && (s[i + 1]
>= '0') && (s[i + 1] <= '9'))
                                        {
                                            int j6 = i + 1; buf = buf +
s[i];
                                            while ((j6 < s.Length) &&
(s[j6] >= '0') && (s[j6] <= '9'))
                                            {
                                                buf = buf + s[j6]; j6++;
                                            }
                                            p3(buf);
Console.WriteLine(buf); writer_lex.WriteLine(buf); buf = "";
                                            i = j6; int j7 = 0; int k7 = 0;
bool f7 = false; string c7 = ""; string c8 = ""; bool g7 = false;
                                            if (i < s.Length) { c7 = "" +
s[i]; }
                                            if (i < s.Length - 1) { c8 = c8
+ s[i] + s[i + 1]; }
                                            while ((!g7) && (k7 < 12))
                                            {
                                                if (O[k7].s == c8) { g7 =
true; }
                                                else { k7++; }
                                            }
                                            if (g7) { p6(c8);
Console.WriteLine(c8); writer_lex.WriteLine(c8); c8 = ""; i += 2; }
                                            else
                                            {
                                                k7 = 0;
                                                while ((!f7) && (j7 < 19))
                                                {
                                                    if ((R[j7].s == c7) &&
(c7 != ".")) { f7 = true; }
                                                    else { j7++; }
                                                }
                                                if (f7) { p4(c7);
Console.WriteLine(c7); writer_lex.WriteLine(c7); c7 = ""; i++; }
                                                else
                                                {
                                                    while ((!f7) && (k7 <
12))
                                                    {
                                                        if (O[k7].s == c7)
{ f7 = true; }
                                                        else { k7++; }
                                                    }
                                                    if (!f7) { er(); }
                                                    else { p6(c7);
Console.WriteLine(c7); writer_lex.WriteLine(c7); c7 = ""; i++; }
                                                }
                                            }
```

```
                                    }
                                    else { er(); }
                                }
                            }
                            else
                            {
                                string c3 = ""; if (i < s.Length) { c3
= "" + s[i]; }
                                bool f3 = false; int j3 = 0; int k3 =
0;
                                while ((!f3) && (j3 < 19))
                                {
                                    if (R[j3].s == c3) { f3 = true; }
                                    else { j3++; }
                                }
                                while ((!f3) && (k3 < 12))
                                {
                                    if (O[k3].s == c3) { f3 = true; }
                                    else { k3++; }
                                }
                                if (f3) { p3(buf);
Console.WriteLine(buf); writer_lex.WriteLine(buf); buf = ""; }
                                else { er(); }
                            }
                        }
                    }
                    else
                    {
                        if ((i < s.Length) && (s[i] == '.'))
                        {
                            if ((i < s.Length - 1) && (s[i + 1] >= '0')
&& (s[i + 1] <= '9'))
                            {
                                buf = buf + s[i]; i++;
                                while ((i < s.Length) && (s[i] >= '0')
&& (s[i] <= '9'))
                                {
                                    buf = buf + s[i]; i++;
                                }
                                string c3 = ""; if (i < s.Length) { c3
= "" + s[i]; }
                                bool f3 = false; int j3 = 0; int k3 =
0;
                                while ((!f3) && (j3 < 19))
                                {
                                    if ((R[j3].s == c3) && (c3 != "."))
{ f3 = true; }
                                    else { j3++; }
                                }
                                while ((!f3) && (k3 < 12))
                                {
```

```
                                    if (O[k3].s == c3) { f3 = true; }
                                    else { k3++; }
                                }
                                if (f3) { p3(buf);
Console.WriteLine(buf); writer_lex.WriteLine(buf); buf = ""; }
                                else { er(); }
                            }
                            else
                            {
                                if ((i < s.Length - 1) && (s[i + 1] ==
'.'))
                                {
                                    p9("" + s[i] + s[i + 1]);
                                    Console.WriteLine("" + s[i] + s[i +
1]); writer_lex.WriteLine("" + s[i] + s[i + 1]); i += 2;
                                }
                                else { p9("" + s[i]);
Console.WriteLine("" + s[i]); writer_lex.WriteLine("" + s[i]); i++; }
                            }
                        }
                        else
                        {
                            if ((i < s.Length) && (s[i] == '{'))
                            {
                                Console.WriteLine("" + s[i]);
writer_lex.WriteLine("" + s[i]);
                                while ((i < s.Length) && (s[i] != '}'))
                                {
                                    i++;
                                }
                                if ((i < s.Length) && (s[i] == '}')) {
Console.WriteLine(s[i]); writer_lex.WriteLine("" + s[i]); }
                            }
                            else
                            {
                                if ((i < s.Length) && (s[i] == '<'))
                                {
                                    if ((i < s.Length - 1) && ((s[i +
1] == '=') || (s[i + 1] == '')))
                                    {
                                        p6("" + s[i] + s[i + 1]);
Console.WriteLine("" + s[i] + s[i + 1]); writer_lex.WriteLine("" + s[i] +
s[i + 1]); i += 2;
                                    }
                                    else { p6("" + s[i]);
Console.WriteLine("" + s[i]); writer_lex.WriteLine("" + s[i]); i++; }
                                }
                                else
                                {
                                    if ((i < s.Length) && (s[i] ==
'>'))
```

```
                                        {
                                            if ((i < s.Length - 1) && (s[i
+ 1] == '=')) { p6("" + s[i] + s[i + 1]); Console.WriteLine("" + s[i] + s[i
+ 1]); writer_lex.WriteLine("" + s[i] + s[i + 1]); i += 2; }
                                            else { p6("" + s[i]);
Console.WriteLine("" + s[i]); writer_lex.WriteLine("" + s[i]); i++; }
                                        }
                                        else
                                        {
                                            if ((i < s.Length) && (s[i] ==
':'))
                                            {
                                                if ((i < s.Length - 1) &&
(s[i + 1] == '=')) { p6("" + s[i] + s[i + 1]); Console.WriteLine("" + s[i]
+ s[i + 1]); writer_lex.WriteLine("" + s[i] + s[i + 1]); i += 2; }
                                                else { p4("" + s[i]);
Console.WriteLine("" + s[i]); writer_lex.WriteLine("" + s[i]); i++; }
                                            }
                                            else
                                            {
                                                string c = ""; if (i <
s.Length) { c = "" + s[i]; }
                                                bool f = false; int j = 0;
int k = 0;
                                                while ((!f) && (j < 19))
                                                {
                                                    if (R[j].s == c) { f =
true; }
                                                    else { j++; }
                                                }
                                                if (f)
                                                {
                                                    p4(c);
Console.WriteLine(c); writer_lex.WriteLine(c); i++;
                                                }
                                                else
                                                {
                                                    while ((!f) && (k <
12))
                                                    {
                                                        if (O[k].s == c) {
f = true; }
                                                        else { k++; }
                                                    }
                                                    if (!f) { er(); }
                                                    else { p6(c);
Console.WriteLine(c); writer_lex.WriteLine(c); i++; }
                                                }
                                            }
                                        }
                                    }
```

```
                            }
                        }
                    }
                }
            }
            static void v()
            {
                FileStream f = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\input.txt",
FileMode.Open);
                StreamReader reader = new StreamReader(f); // создаём
«потоковый читатель» и связываем его с файловым потоком
                string s = reader.ReadToEnd(); //считываем все данные с
потока
                reader.Close(); //закрываем поток
                string[] stroki = s.Split(new char[] { '\n' });
                Console.WriteLine(s);
                S(s);
                if (err)
                {
                    writer.WriteLine("Yest' oshibka(-i)");
                }
                else
                {
                    writer.WriteLine("Net oshibok");
                }
                writer.WriteLine("");
                FileStream h = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\tablitsy.txt",
FileMode.Create, FileAccess.Write);
                StreamWriter writer1 = new StreamWriter(h);
                int i = 0;
                writer1.WriteLine("Chisla:");
                while (N[i].s != null)
                {
                    bool bo = false; int j = 0; string st = ""; bool bo1 =
false;
                    while (j < N[i].s.Length)
                    {
                        if (N[i].s[j] == '.')
                        {
                            bo1 = true; j++;
                        }
                        else
                        {
                            if ((N[i].s[j] == 'e') || (N[i].s[j] == 'E'))
                            {
                                bo = true; bo1 = false; j++;
```

```
                        }
                        else
                        {
                            j++;
                        }
                    }
                }
                if (bo) { st = " - chislo s plavayushchey tochkoy "; }
                else
                {
                    if (bo1) { st = " - chislo s fiksirovannoy tochkoy
"; }
                    else { st = " - tseloye chislo "; }
                }
                writer1.WriteLine(N[i].id.ToString() + " " + N[i].s +
st); i++;
            }
            writer1.WriteLine("Sluzhebnye slova:");
            for (int i1 = 0; i1 < 36; i1++)
            {
                writer1.WriteLine(W[i1].id.ToString() + " " + W[i1].s);
            }
            writer1.WriteLine("Razdeliteli:");
            for (int i1 = 0; i1 < 19; i1++)
            {
                writer1.WriteLine(R[i1].id.ToString() + " " + R[i1].s);
            }
            writer1.WriteLine("Operatsii:");
            for (int i1 = 0; i1 < 12; i1++)
            {
                writer1.WriteLine(O[i1].id.ToString() + " " + O[i1].s);
            }
            writer.Close();
            FileStream f1 = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\output.txt",
FileMode.Open);
            StreamReader reader1 = new StreamReader(f1);
            string s1 = reader1.ReadToEnd();
            reader1.Close();
            //Console.WriteLine(s1);
            int o = 0; string s2 = ""; int k = 0; int max = 0;
            while (o < s1.Length)
            {
                if (s1[o] == 'I')
                {
                    string s3 = "";
                    s2 = s2 + s1[o]; o++;
                    while ((s1[o] >= '0') && (s1[o] <= '9') && (o <
s1.Length))
                    {
```

```
                        s2 = s2 + s1[o]; s3 = s3 + s1[o]; o++;
                    }
                    int a = o; k = Convert.ToInt16(s3);
                    if (k > max) { max = k; }
                    while ((s1[a] != 'W') && (a < s1.Length))
                    {
                        a++;
                    }
                    if (a < s1.Length)
                    {
                        s2 = s2 + s1[a]; a++;
                        while ((s1[a] >= '0') && (s1[a] <= '9') && (a <
s1.Length))
                        {
                            s2 = s2 + s1[a]; a++;
                        }
                    }
                }
                else
                {
                    o++;
                }
            }
            int[] mas = new int[max]; o = 0;
            while (o < s2.Length)
            {
                if (s2[o] == 'I')
                {
                    o++; string s4 = "";
                    while ((o < s2.Length) && (s2[o] >= '0') && (s2[o]
<= '9'))
                    {
                        s4 = s4 + s2[o]; o++;
                    }
                    int b = Convert.ToInt16(s4) - 1;
                    o++; s4 = "";
                    while ((o < s2.Length) && (s2[o] >= '0') && (s2[o]
<= '9'))
                    {
                        s4 = s4 + s2[o]; o++;
                    }
                    int u = Convert.ToInt16(s4);
                    if ((b < max) && (mas[b] == 0)) { mas[b] = u; }
                }
                else
                {
                    o++;
                }
            }
            i = 0;
            writer1.WriteLine("Identifikatory:");
```

```
                while (I[i].s != null)
                {
                    if ((mas[i] - 1 == 4) || (mas[i] - 1 == 5) || (mas[i] -
1 == 6) || (mas[i] - 1 == 22) || (mas[i] - 1 == 28))
                    {
                        writer1.WriteLine(I[i].id.ToString() + " " + I[i].s
+ " - " + W[mas[i] - 1].s + " "); i++;
                    }
                    else
                    {
                        writer1.WriteLine(I[i].id.ToString() + " " + I[i].s
+ " "); i++;
                    }
                }
                writer1.WriteLine("");
                writer1.Close();
                writer_lex.Close();
            }
        private void button1_Click(object sender, EventArgs e)
        {
            FileStream h = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\tablitsy.txt",
FileMode.Open);
            StreamReader reader = new StreamReader(h);
            string s = reader.ReadToEnd();
            reader.Close();
            string[] stroki = s.Split(new char[] { '\n' });
            for(int i=0; i<stroki.Length; i++)
            {
                string s0 = "";
                for(int j=0; j<stroki[i].Length; j++)
                {
                    if ((stroki[i][j] != '\n') && (stroki[i][j] != '\r'))
                    {
                        s0 = s0 + stroki[i][j];
                    }
                }
                stroki[i] = s0;
            }
            int x = 0;
            while ((x < stroki.Length) && (stroki[x] != "Identifikatory:"))
            {
                x++;
            }
            x++;
            string s1 = "";
            while (x < stroki.Length)
            {
                s1 = s1 + stroki[x] + "\n";
                x++;
```

```
            }
            MessageBox.Show(s1);
        }
        private void button6_Click(object sender, EventArgs e)
        {
            string st = textBox1.Text;
            FileStream g = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\input.txt",
FileMode.Create, FileAccess.Write);
            StreamWriter writer = new StreamWriter(g);
            writer.Write(st);
            writer.Close();
            v();
            FileStream h = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\output.txt",
FileMode.Open);
            StreamReader reader = new StreamReader(h);
            string s = reader.ReadToEnd();
            reader.Close();
            string[] stroki = s.Split(new char[] { '\n' });
            for (int i = 0; i < stroki.Length; i++)
            {
                string s0 = "";
                for (int j = 0; j < stroki[i].Length; j++)
                {
                    if ((stroki[i][j] != '\n') && (stroki[i][j] != '\r'))
                    {
                        s0 = s0 + stroki[i][j];
                    }
                }
                stroki[i] = s0;
            }
            string s1 = "";
            for(int i=0; i<stroki.Length; i++)
            {
                s1 = s1 + stroki[i] + " ";
            }
            MessageBox.Show(st+"\n"+s1);
        }
        private void button2_Click(object sender, EventArgs e)
        {
            FileStream h = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\tablitsy.txt",
FileMode.Open);
            StreamReader reader = new StreamReader(h);
            string s = reader.ReadToEnd();
            reader.Close();
            string[] stroki = s.Split(new char[] { '\n' });
```

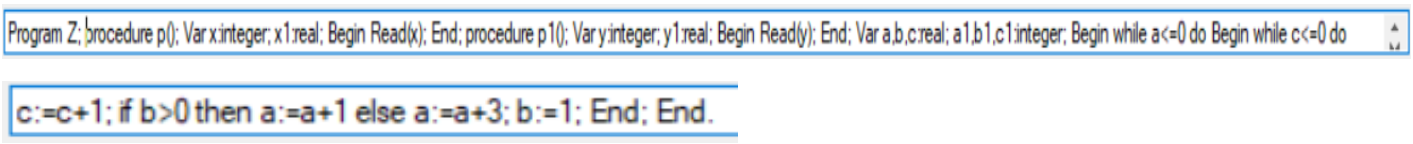
```
            for (int i = 0; i < stroki.Length; i++)
            {
                string s0 = "";
                for (int j = 0; j < stroki[i].Length; j++)
                {
                    if ((stroki[i][j] != '\n') && (stroki[i][j] != '\r'))
                    {
                        s0 = s0 + stroki[i][j];
                    }
                }
                stroki[i] = s0;
            }
            int x = 0;
            while ((x < stroki.Length) && (stroki[x] != "Chisla:"))
            {
                x++;
            }
            x++;
            string s1 = "";
            while ((x < stroki.Length)&&(stroki[x]!= "Sluzhebnye slova:"))
            {
                s1 = s1 + stroki[x] + "\n";
                x++;
            }
            MessageBox.Show(s1);
        }
        private void button3_Click(object sender, EventArgs e)
        {
            FileStream h = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\tablitsy.txt",
FileMode.Open);
            StreamReader reader = new StreamReader(h);
            string s = reader.ReadToEnd();
            reader.Close();
            string[] stroki = s.Split(new char[] { '\n' });
            for (int i = 0; i < stroki.Length; i++)
            {
                string s0 = "";
                for (int j = 0; j < stroki[i].Length; j++)
                {
                    if ((stroki[i][j] != '\n') && (stroki[i][j] != '\r'))
                    {
                        s0 = s0 + stroki[i][j];
                    }
                }
                stroki[i] = s0;
            }
            int x = 0;
            while ((x < stroki.Length) && (stroki[x] != "Sluzhebnye
slova:"))
```

```
            {
                x++;
            }
            x++;
            string s1 = "";
            while ((x < stroki.Length) && (stroki[x] != "Razdeliteli:"))
            {
                s1 = s1 + stroki[x] + "\n";
                x++;
            }
            MessageBox.Show(s1);
        }
        private void button4_Click(object sender, EventArgs e)
        {
            FileStream h = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\tablitsy.txt",
FileMode.Open);
            StreamReader reader = new StreamReader(h);
            string s = reader.ReadToEnd();
            reader.Close();
            string[] stroki = s.Split(new char[] { '\n' });
            for (int i = 0; i < stroki.Length; i++)
            {
                string s0 = "";
                for (int j = 0; j < stroki[i].Length; j++)
                {
                    if ((stroki[i][j] != '\n') && (stroki[i][j] != '\r'))
                    {
                        s0 = s0 + stroki[i][j];
                    }
                }
                stroki[i] = s0;
            }
            int x = 0;
            while ((x < stroki.Length) && (stroki[x] != "Razdeliteli:"))
            {
                x++;
            }
            x++;
            string s1 = "";
            while ((x < stroki.Length) && (stroki[x] != "Operatsii:"))
            {
                s1 = s1 + stroki[x] + "\n";
                x++;
            }
            MessageBox.Show(s1);
        }
        private void button5_Click(object sender, EventArgs e)
        {
```

```
            FileStream h = new
FileStream("C:\\Users\\Lizard\\Documents\\Visual Studio
2015\\Projects\\ConsoleApplication4\\ConsoleApplication4\\tablitsy.txt",
FileMode.Open);
            StreamReader reader = new StreamReader(h);
            string s = reader.ReadToEnd();
            reader.Close();
            string[] stroki = s.Split(new char[] { '\n' });
            for (int i = 0; i < stroki.Length; i++)
            {
                string s0 = "";
                for (int j = 0; j < stroki[i].Length; j++)
                {
                    if ((stroki[i][j] != '\n') && (stroki[i][j] != '\r'))
                    {
                        s0 = s0 + stroki[i][j];
                    }
                }
                stroki[i] = s0;
            }
            int x = 0;
            while ((x < stroki.Length) && (stroki[x] != "Operatsii:"))
            {
                x++;
            }
            x++;
            string s1 = "";
            while ((x < stroki.Length) && (stroki[x] != "Identifikatory:"))
            {
                s1 = s1 + stroki[x] + "\n";
                x++;
            }
            MessageBox.Show(s1);
        }
    }
    }
```

## 5 Результаты работы программы

*Текстовое поле, в которое вводится программа:*




*Полученные коды лексем:*

W1 R11 I1 R10 R12 R14 W10 R11 I2 R8 R9 R10 R12 R14 W4 R11 I3 R13 W5 R10 R11 I4 R13 W6 R10 R12 R14 W2 R12 R14 W21 R8 I3 R9 R10 R12 R14 W3 R10 R12

R14 W10 R11 I5 R8 R9 R10 R12 R14 W4 R11 I6 R13 W5 R10 R11 I7 R13 W6 R10 R12 R14 W2 R12 R14 W21 R8 I6 R9 R10 R12 R14 W3 R10 R12 R14 W4 R11 I8 R5 I9 R5 I10 R13 W6 R10 R11 I11 R5 I12 R5 I13 R13 W5 R10 R12 R14 W2 R12 R14 W24 R11 I8 O10 N1 R11 W27 R11 W2 R12 R14 W24 R11 I10 O10 N1 R11 W27 R11 I10 O12 I10 O1 N2 R10 R12 R14 W14 R11 I9 O7 N1 R11 W15 R11 I8 O12 I8 O1 N2 R11 W16 R11 I8 O12 I8 O1 N3 R10 R12 R14 I9 O12 N2 R10 R11 R12 R14 W3 R10 R12 R14 W3 R1 Net oshibok

*Полученные таблицы лексем:*

Chisla:

1 0 - tseloye chislo

2 1 - tseloye chislo

3 3 - tseloye chislo

Sluzhebnye slova:

1 Program

2 Begin

3 End

4 Var

5 integer

6 real

7 string

8 array

9 of

10 procedure

11 function

12 return

13 goto

14 if

15 then

16 else

17 not

18 and

19 or

20 xor

21 Read

22 Write

23 boolean

24 while

25 repeat

26 until

27 do

28 to

29 longint

30 true

31 false

32 for

33 sqrt

34 div

35 mod

36 sqr

Razdeliteli:

1 .

2 '

3 [

4 ]

5 ,

6 {

7 }

8 (

9 )

10 ;

11

12

13 :

14

15

16

17 \

18 "

19 ..

Operatsii:

1 +

2 -

3 *

4 /

5 ^

6 <

7 >

8 =

9 >=

10 <=

11 <>

12 :=

Identifikatory:

1 Z

2 p

3 x - integer

4 x1 - real

5 p1

6 y - integer

7 y1 - real

8 a - real

9 b - real

10 c - real

11 a1 - integer

12 b1 - integer

13 c1 – integer

## 6 Вывод

Во время работы был построен лексический анализатор заданного подмножества языковых конструкций и операторов языка Pascal.