МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С. П. КОРОЛЕВА»

(САМАРСКИЙ УНИВЕРСИТЕТ)

Отчёт по лабораторной работе

по курсу «Теория формальных языков и грамматик»

Вариант №19

Выполнил:
Власов Г.В.
Группа 6203

Проверил:
Литвинов В. Г.

Самара 2019

Задание:

Синтаксис: Написать программу синтаксического анализа оператора присваивания языка Modula-2 (язык регистро-независимый). Грамматика имеет следующий вид:

№	Правила		
(1)	*start*	=	*var* **:=** *term* { *math-oper term* } **;**
(2)	*term*	=	*var* | *any-numb*
(3)	*var*	=	*id* [**[** *indexes* **]**]
(4)	*indexes*	=	*index* { **,** *index* }
(5)	*index*	=	*id* | *int-numb*
(6)	*any-numb*	=	*int-numb* | *fix-point-numb* | *real-numb*
(7)	*math-oper*	=	+ | - | * | / | **MOD** | **DIV** | = | < | <= | > | >= | #

- *id* — идентификатор языка Modula-2, начинается с буквы или знака подчеркивания, далее могут следовать буквы, цифры и знаки подчеркивания. Ограничения:
  - имеет длину не более 8 символов;
  - не является зарезервированным словом: **MOD**, **DIV**.
- *int-numb* — целое число в диапазоне -32768 – 32767;
- *fix-point-numb* — число с фиксированной точкой;
- *real-numb* — число с плавающей точкой.

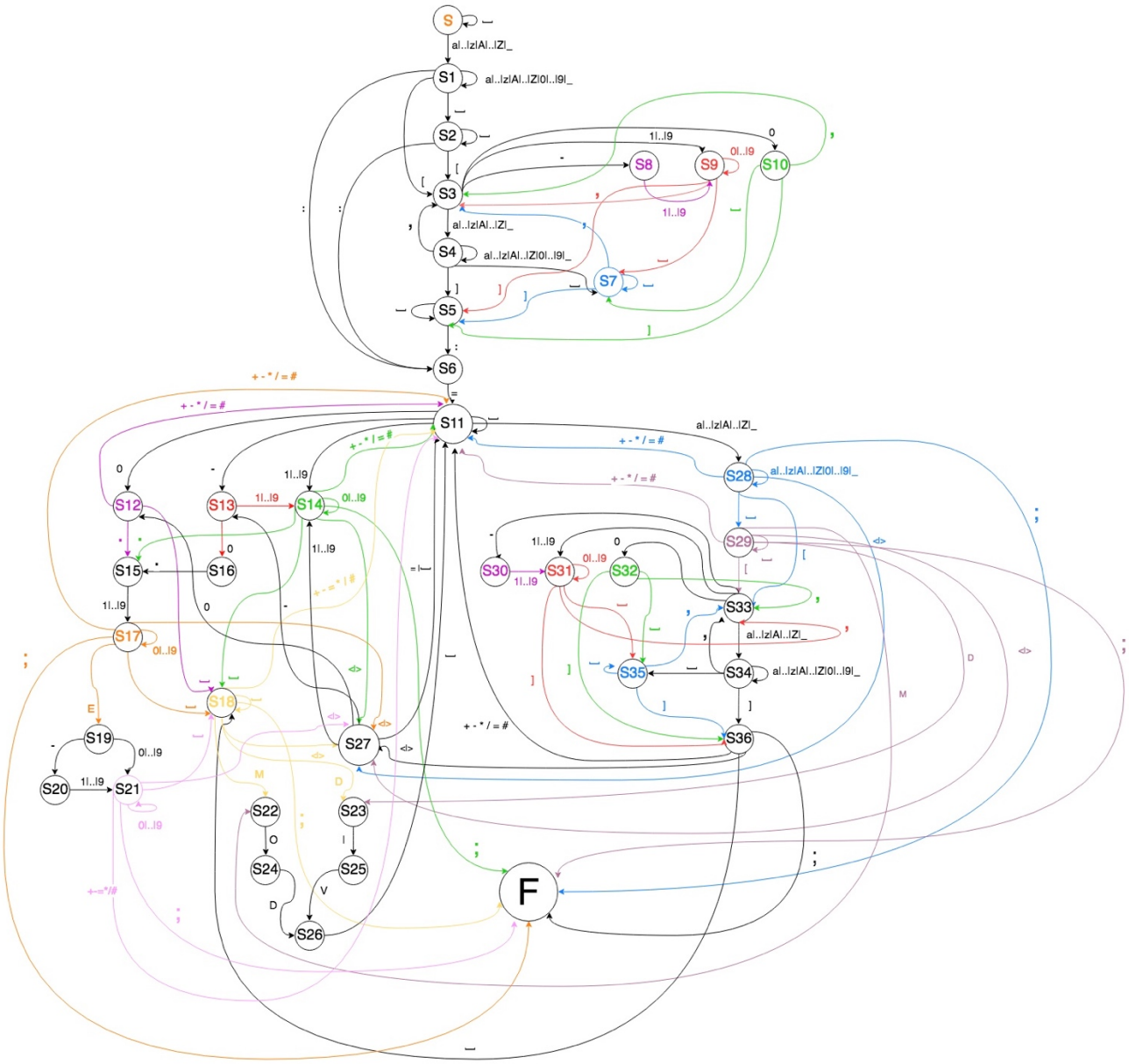
Семантика:

- Построить и вывести на печать по завершении анализа таблицы идентификаторов и констант.
- Учесть перечисленные ограничения на идентификаторы и константы.
- Сообщать об ошибках, когда число цифр в мантиссе более 15.
- Сообщать об ошибках при анализе, указывая курсором место ошибки и ее содержание.

Примеры правильных цепочек:

```
ABC[1, I, LF, 25] := AB + 135.5E8 - LF * DKL1 / ZP MOD KP;
ASDE := Tgfr[1, 2, 7] > DK[Y, I] + 135 / 987.9;
```

Детерминированные конечные автоматы языка:



Листинг программы:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Windows.Forms;
using System.Reflection;
using System.Resources;
using System.Globalization;

namespace Analizator
{
    class Controller
    {
        enum State
        {
            S, S1, S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12, S13, S14, S15,
            S16, S17, S18, S19, S20, S21, S22, S23, S24, S25, S26, S27, S28, S29, S30, S31,
            S32, S33, S34, S35, S36, F, E
        }
        enum TermType
        {
            Int, Double, Id
        }
        public static CheckingResult CheckString(String str)
        {
            State state = State.S;
            TermType termType = TermType.Id;
            String Term = "";
            List<string> ReserveWords = new List<string>
            {
                "div",
                "mod",
            };
            List<char> MathOpers = new List<char>
            {
                '+',
                '-',
                '/',
                '*',
                '#',
                '=',
            };

            ExceptionResult error = null;
            List<String> vars = new List<String>();
            List<String> consts = new List<String>();

            str = str.Replace('\r', ' ');
            char[] strr = str.ToLower().ToCharArray();
            bool FinishTerm = false;
            int IndexError = -1;
            int i = 0;
            while (i < strr.Length && state != State.E)
            {
                char ligne = strr[i];
                FinishTerm = false;
                switch (state)
                {
```

```
                    case State.S:
                        {
                            if (ligne == ' ')
                            {
                                state = State.S;
                            }
                            else if (char.IsLetter(ligne) || ligne == '_')
                            {
                                state = State.S1;
                                termType = TermType.Id;
                                IndexError = i;
                                Term += ligne;
                            }
                            else
                            {
                                state = State.E;
                                error = new ExceptionResult("S ошибка. ожидалась буква или знак
подчёркивания", i);
                            }
                            break;
                        }
                    case State.S1:
                        {
                            if (ligne == ' ')
                            {
                                state = State.S2;
                                FinishTerm = true;
                            }
                            else if (char.IsLetterOrDigit(ligne) || (ligne == '_'))
                            {
                                termType = TermType.Id;
                                Term += ligne;
                                state = State.S1;
                            }
                            else if (ligne == '[')
                            {
                                state = State.S3;
                                FinishTerm = true;
                            }
                            else if (ligne == ':')
                            {
                                state = State.S6;
                                FinishTerm = true;
                            }
                            else
                            {
                                state = State.E;
                                error = new ExceptionResult("S1 ошибка.ожидалась буква, цифра,
знак подчёркивания, [ или :", i);
                            }
                            break;
                        }
                    case State.S2:
                        {
                            if (ligne == ':')
                            {
                                state = State.S6;
                                FinishTerm = true;
                            }
                            else if (ligne == ' ')
                            {
                                state = State.S2;
                            }
                            else if (ligne == '[')
                            {
```

```
                                state = State.S3;
                                FinishTerm = true;
                            }
                            else
                            {
                                state = State.E;
                                error = new ExceptionResult("S2 ошибка. ожидался пробел,
двоеточие или [", i);
                            }
                            break;

                        }
                    case State.S3:
                        {

                            if (char.IsLetter(ligne) || ligne == '_')
                            {
                                termType = TermType.Id;
                                IndexError = i;
                                Term += ligne;
                                state = State.S4;
                            }
                            else if (ligne == ' ')
                            {
                                state = State.S3;
                            }
                            else if (ligne == '-')
                            {

                                termType = TermType.Int;
                                IndexError = i;
                                Term += ligne;
                                state = State.S8;
                            }
                            else if (ligne == '0')
                            {
                                termType = TermType.Int;
                                IndexError = i;
                                Term += ligne;
                                state = State.S10;
                            }
                            else if (char.IsDigit(ligne) && (ligne != '0'))
                            {
                                termType = TermType.Int;
                                IndexError = i;
                                Term += ligne;
                                state = State.S9;
                            }
                            else
                            {
                                error = new ExceptionResult("S3 ошибка.ожидались буква, знак
подчёркивания, либо любая цифра", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S4:
                        {
                            if (ligne == ' ')
                            {
                                termType = TermType.Id;
                                Term += ligne;
                                FinishTerm = true;
                                state = State.S7;
```

```
                            }
                            else if (char.IsLetterOrDigit(ligne) || (ligne == '_'))
                            {
                                termType = TermType.Id;
                                Term += ligne;
                                state = State.S4;
                            }
                            else if (ligne == ']')
                            {
                                termType = TermType.Id;
                                state = State.S5;
                                FinishTerm = true;
                            }
                            else if (ligne == ',')
                            {
                                termType = TermType.Id;
                                state = State.S3;
                                FinishTerm = true;
                            }
                            else
                            {
                                error = new ExceptionResult("S4 ошибка.ожидался пробел, запятая,
], либо идентификатор", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S5:
                        {
                            if (ligne == ' ')
                            {
                                state = State.S5;
                            }
                            else if (ligne == ':')
                            {
                                state = State.S6;
                            }
                            else
                            {
                                error = new ExceptionResult("S5 ошибка.ожидалось двоеточие или
пробел", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S6:
                        {
                            if (ligne == '=')
                            {
                                state = State.S11;
                            }
                            else
                            {
                                error = new ExceptionResult("S6 ошибка.ожидался знак =", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S7:
                        {
                            if (ligne == ' ')
                            {
                                state = State.S7;
                            }
                            else if (ligne == ']')
```

```
                        {
                            state = State.S5;
                        }
                        else if (ligne == ',')
                        {
                            state = State.S3;
                        }
                        else
                        {
                            error = new ExceptionResult("S7 ошибка. ожидался пробел, запятая
или ] ", i);
                            state = State.E;
                        }
                        break;
                    }
                case State.S8:
                    {
                        if (char.IsDigit(ligne) && (ligne != '0'))
                        {
                            termType = TermType.Int;
                            Term += ligne;
                            state = State.S9;
                        }
                        else
                        {
                            error = new ExceptionResult("S8 ошибка. ожидалась цифра от 1 до
9 ", i);
                            state = State.E;
                        }
                        break;
                    }
                case State.S9:
                    {
                        if (char.IsDigit(ligne))
                        {
                            termType = TermType.Int;
                            Term += ligne;
                            state = State.S9;
                        }
                        else if (ligne == ' ')
                        {
                            termType = TermType.Int;
                            FinishTerm = true;
                            state = State.S7;
                        }
                        else if (ligne == ']')
                        {
                            termType = TermType.Int;
                            FinishTerm = true;
                            state = State.S5;
                        }
                        else if (ligne == ',')
                        {
                            termType = TermType.Int;
                            FinishTerm = true;
                            state = State.S3;
                        }
                        else
                        {
                            error = new ExceptionResult("S9 ошибка.ожидалась цифра, пробел,
], запятая", i);
                            state = State.E;
                        }
                        break;
                    }
```

```
                    case State.S10:
                        {
                            if (ligne == ' ')
                            {
                                state = State.S7;
                                FinishTerm = true;
                            }
                            else if (ligne == ']')
                            {
                                state = State.S5;
                                FinishTerm = true;
                            }
                            else if (ligne == ',')
                            {
                                state = State.S3;
                                FinishTerm = true;
                            }
                            else
                            {
                                error = new ExceptionResult("S10 ошибка.ожидался пробел, ], либо
запятая", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S11:
                        {
                            if (ligne == '0')
                            {
                                IndexError = i;
                                Term += ligne;
                                state = State.S12;
                            }
                            else if (ligne == '-')
                            {
                                IndexError = i;
                                Term += ligne;
                                state = State.S13;
                            }
                            else if (char.IsDigit(ligne) && (ligne != '0'))
                            {
                                IndexError = i;
                                Term += ligne;
                                state = State.S14;
                            }
                            else if (ligne == ' ')
                            {
                                state = State.S11;
                            }
                            else if (char.IsLetter(ligne) || ligne == '_')
                            {
                                termType = TermType.Id;
                                IndexError = i;
                                Term += ligne;
                                state = State.S28;
                            }
                            else
                            {
                                error = new ExceptionResult("S11 ошибка.ожидался пробел, -,
любая цифра или идентификатор", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S12:
```

```
                        {
                            if (MathOpers.Contains(ligne))
                            {
                                termType = TermType.Int;
                                FinishTerm = true;
                                state = State.S11;
                            }
                            else if (ligne == '.')
                            {
                                termType = TermType.Double;
                                Term += ligne;
                                state = State.S15;
                            }
                            else if (ligne == ' ')
                            {
                                termType = TermType.Int;
                                FinishTerm = true;
                                state = State.S18;
                            }
                            else
                            {
                                error = new ExceptionResult("S12 ошибка.ожидался пробел, точка
или + - / * # = ", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S13:
                        {
                            if (ligne == '0')
                            {
                                termType = TermType.Double;
                                Term += ligne;
                                state = State.S16;
                            }
                            else if (char.IsDigit(ligne) && (ligne != '0'))
                            {
                                Term += ligne;
                                state = State.S14;
                            }
                            else
                            {
                                error = new ExceptionResult("S13 ошибка.ожидалась цифра", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S14:
                        {
                            if (MathOpers.Contains(ligne))
                            {
                                termType = TermType.Int;
                                FinishTerm = true;
                                state = State.S11;
                            }
                            else if (char.IsDigit(ligne))
                            {
                                Term += ligne;
                                state = State.S14;
                            }
                            else if (ligne == '.')
                            {
                                termType = TermType.Double;
                                Term += ligne;
                                state = State.S15;
```

```
                }
                else if (ligne == ' ')
                {
                    termType = TermType.Int;
                    FinishTerm = true;
                    state = State.S18;
                }
                else if (ligne == '>' || ligne == '<')
                {
                    termType = TermType.Int;
                    FinishTerm = true;
                    state = State.S27;
                }
                else if (ligne == ';')
                {
                    termType = TermType.Int;
                    FinishTerm = true;
                    state = State.F;
                }
                else
                {
                    error = new ExceptionResult("S14 ошибка.ожидался пробел, ;, .,
цифра или + - / * # =", i);
                    state = State.E;
                }
                break;
            }
        case State.S15:
            {
                if (char.IsDigit(ligne) || ligne != '0')
                {
                    termType = TermType.Double;
                    Term += ligne;
                    state = State.S17;
                }
                else
                {
                    error = new ExceptionResult("S15 ошибка.ожидалась цифра от 1 до
9", i);
                    state = State.E;
                }
                break;
            }
        case State.S16:
            {
                if (ligne == '.')
                {
                    termType = TermType.Double;
                    Term += ligne;
                    state = State.S15;
                }
                else
                {
                    error = new ExceptionResult("S16 ошибка.ожидалась точка", i);
                    state = State.E;
                }
                break;
            }
        case State.S17:
            {
                if (MathOpers.Contains(ligne))
                {
                    termType = TermType.Double;
                    FinishTerm = true;
                    state = State.S11;
```

```
                        }
                        else if (ligne == ';')
                        {
                            termType = TermType.Double;
                            FinishTerm = true;
                            state = State.F;
                        }
                        else if (char.IsDigit(ligne))
                        {
                            termType = TermType.Double;
                            Term += ligne;
                            state = State.S17;
                        }
                        else if (ligne == ' ')
                        {
                            termType = TermType.Double;
                            FinishTerm = true;
                            state = State.S18;
                        }
                        else if (ligne == '<' || ligne == '>')
                        {
                            termType = TermType.Double;
                            FinishTerm = true;
                            state = State.S27;
                        }
                        else if (ligne == 'e')
                        {
                            termType = TermType.Double;
                            Term += ligne;
                            state = State.S19;
                        }
                        else
                        {
                            error = new ExceptionResult("S17 ошибка.ожидался пробел, ; или +
- / * # =", i);
                            state = State.E;
                        }
                        break;
                    }
                case State.S18:
                    {
                        if (ligne == ' ')
                        {
                            state = State.S18;
                        }
                        else if (ligne == '>' || ligne == '<')
                        {
                            state = State.S27;
                        }
                        else if (ligne == 'm')
                        {
                            state = State.S22;
                        }
                        else if (ligne == 'd')
                        {
                            state = State.S23;
                        }
                        else if (ligne == ';')
                        {
                            state = State.F;
                        }
                        else if (MathOpers.Contains(ligne))
                        {
                            state = State.S11;
                        }
```

```
                            else
                            {
                                state = State.E;
                                error = new ExceptionResult("S18 ошибка.ожидался пробел, mod,
div, ;, + - / * # = >= <=", i);
                            }
                            break;
                        }
                    case State.S19:
                        {
                            if (ligne == '-')
                            {
                                termType = TermType.Double;
                                Term += ligne;
                                state = State.S20;
                            }
                            else if (char.IsDigit(ligne))
                            {
                                termType = TermType.Double;
                                Term += ligne;
                                state = State.S21;
                            }
                            else
                            {
                                error = new ExceptionResult("S19 ошибка.ожидалась цифра или знак
минус", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S20:
                        {
                            if (char.IsDigit(ligne) && ligne != '0')
                            {
                                termType = TermType.Double;
                                Term += ligne;
                                state = State.S21;
                            }
                            else
                            {
                                error = new ExceptionResult("S20 ошибка.ожидалась цифра от 1 до
9", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S21:
                        {
                            if (ligne == ';')
                            {
                                termType = TermType.Double;
                                FinishTerm = true;
                                state = State.F;
                            }
                            else if (ligne == ' ')
                            {
                                termType = TermType.Double;
                                FinishTerm = true;
                                state = State.S18;
                            }
                            else if (ligne == '>' || ligne == '<')
                            {
                                termType = TermType.Double;
                                FinishTerm = true;
                                state = State.S27;
```

```
                            }
                            else if (char.IsDigit(ligne))
                            {
                                termType = TermType.Double;
                                Term += ligne;
                                state = State.S21;
                            }
                            else if (MathOpers.Contains(ligne))
                            {
                                termType = TermType.Double;
                                FinishTerm = true;
                                state = State.S11;
                            }
                            else
                            {
                                error = new ExceptionResult("S21 ошибка.ожидался пробел, ;,
цифра или + - / * # = <= >=", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S22:
                        {
                            if (ligne == 'o')
                            {
                                state = State.S24;
                            }
                            else
                            {
                                error = new ExceptionResult("S22 ошибка.ожидася mod", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S23:
                        {
                            if (ligne == 'i')
                            {
                                state = State.S25;
                            }
                            else
                            {
                                error = new ExceptionResult("S23 ошибка.ожидася div", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S24:
                        {
                            if (ligne == 'd')
                            {
                                state = State.S26;
                            }
                            else
                            {
                                error = new ExceptionResult("S24 ошибка.ожидася mod", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S25:
                        {
                            if (ligne == 'v')
                            {
                                state = State.S26;
```

```
                        }
                        else
                        {
                            error = new ExceptionResult("S25 ошибка.ожидася div", i);
                            state = State.E;
                        }
                        break;
                    }
                case State.S26:
                    {
                        if (ligne == ' ')
                        {
                            state = State.S11;
                        }
                        else
                        {
                            error = new ExceptionResult("S26 ошибка.ожидался пробел", i);
                            state = State.E;
                        }
                        break;
                    }
                case State.S27:
                    {
                        if (ligne == '=' || ligne == ' ')
                        {
                            state = State.S11;
                        }
                        else if (char.IsDigit(ligne) && ligne != '0')
                        {
                            IndexError = i;
                            Term += ligne;
                            state = State.S14;
                        }
                        else if (ligne == '-')
                        {
                            IndexError = i;
                            Term += ligne;
                            state = State.S13;
                        }
                        else if (ligne == '0')
                        {
                            IndexError = i;
                            Term += ligne;
                            state = State.S12;
                        }
                        else
                        {
                            error = new ExceptionResult("S27 ошибка.ожидался пробел, цифра
или =", i);
                            state = State.E;
                        }
                        break;
                    }
                case State.S28:
                    {
                        if (MathOpers.Contains(ligne))
                        {
                            termType = TermType.Id;
                            FinishTerm = true;
                            state = State.S11;
                        }
                        else if (ligne == ' ')
                        {
                            termType = TermType.Id;
                            FinishTerm = true;
```

```
                                state = State.S29;
                            }
                            else if (ligne == '[')
                            {
                                termType = TermType.Id;
                                FinishTerm = true;
                                state = State.S33;
                            }
                            else if (char.IsLetterOrDigit(ligne) || ligne == '_')
                            {
                                termType = TermType.Id;
                                Term += ligne;
                                state = State.S28;
                            }
                            else if (ligne == '>' || ligne == '<')
                            {
                                termType = TermType.Id;
                                FinishTerm = true;
                                state = State.S27;
                            }
                            else if (ligne == ';')
                            {
                                termType = TermType.Id;
                                FinishTerm = true;
                                state = State.F;
                            }
                            else
                            {
                                error = new ExceptionResult("S28 ошибка.ожидался прбел, [, _, ;,
+ - / * # = <= >=, буква или цифра", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S29:
                        {
                            if (MathOpers.Contains(ligne))
                            {
                                state = State.S11;
                            }
                            else if (ligne == ' ')
                            {
                                state = State.S29;
                            }
                            else if (ligne == '[')
                            {
                                state = State.S33;
                            }
                            else if (ligne == 'm')
                            {
                                state = State.S22;
                            }
                            else if (ligne == 'd')
                            {
                                state = State.S23;
                            }
                            else if (ligne == '>' || ligne == '<')
                            {
                                state = State.S27;
                            }
                            else if (ligne == ';')
                            {
                                state = State.F;
                            }
                            else
```

```
                            {
                                error = new ExceptionResult("S29 ошибка.ожидался пробел, [, mod,
div, ;, + - / * # = <= >=", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S30:
                        {
                            if (char.IsDigit(ligne) && ligne != '0')
                            {
                                termType = TermType.Int;
                                Term += ligne;
                                state = State.S31;
                            }
                            else
                            {
                                error = new ExceptionResult("S30 ошибка.ожидалась цифра от 1 до
9", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S31:
                        {
                            if (char.IsDigit(ligne))
                            {
                                termType = TermType.Int;
                                Term += ligne;
                                state = State.S31;
                            }
                            else if (ligne == ']')
                            {
                                termType = TermType.Int;
                                FinishTerm = true;
                                state = State.S36;
                            }
                            else if (ligne == ' ')
                            {
                                termType = TermType.Int;
                                FinishTerm = true;
                                state = State.S35;
                            }
                            else if (ligne == ',')
                            {
                                termType = TermType.Int;
                                FinishTerm = true;
                                state = State.S33;
                            }
                            else
                            {
                                error = new ExceptionResult("S31 ошибка.ожидалась цифра,
запятая, пробел или ]", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S32:
                        {
                            if (ligne == ']')
                            {
                                FinishTerm = true;
                                state = State.S36;
                            }
                            else if (ligne == ' ')
```

```
                            {
                                FinishTerm = true;
                                state = State.S35;
                            }
                            else if (ligne == ',')
                            {
                                FinishTerm = true;
                                state = State.S33;
                            }
                            else
                            {
                                error = new ExceptionResult("S32 ошибка.ожидался пробел, запятая
или ]", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S33:
                        {
                            if (ligne == '0')
                            {
                                termType = TermType.Int;
                                IndexError = i;
                                Term += ligne;
                                state = State.S32;
                            }
                            else if (char.IsDigit(ligne) && ligne != '0')
                            {
                                termType = TermType.Int;
                                IndexError = i;
                                Term += ligne;
                                state = State.S31;
                            }
                            else if (ligne == '-')
                            {
                                termType = TermType.Int;
                                IndexError = i;
                                Term += ligne;
                                state = State.S30;
                            }
                            else if (char.IsLetter(ligne) || ligne == '_')
                            {
                                termType = TermType.Id;
                                IndexError = i;
                                Term += ligne;
                                state = State.S34;
                            }
                            else
                            {
                                error = new ExceptionResult("S33 ошибка.ожидалась буква, цифра,
- или _", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S34:
                        {
                            if (ligne == ' ')
                            {
                                termType = TermType.Id;
                                FinishTerm = true;
                                state = State.S35;
                            }
                            else if (ligne == ']')
                            {
```

```
                                termType = TermType.Id;
                                FinishTerm = true;
                                state = State.S36;
                            }
                            else if (char.IsLetterOrDigit(ligne) || ligne == '_')
                            {
                                termType = TermType.Id;
                                Term += ligne;
                                state = State.S34;
                            }
                            else if (ligne == ',')
                            {
                                termType = TermType.Id;
                                FinishTerm = true;
                                state = State.S33;
                            }
                            else
                            {
                                error = new ExceptionResult("S34 ошибка.ожидалась буква, цифра,
запятая, пробел, знак подчёркивания или ]", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S35:
                        {
                            if (ligne == ' ')
                            {
                                state = State.S35;
                            }
                            else if (ligne == ',')
                            {
                                state = State.S33;
                            }
                            else if (ligne == ']')
                            {
                                state = State.S36;
                            }
                            else
                            {
                                error = new ExceptionResult("S35 ошибка.ожидался пробел, запятая
или ]", i);
                                state = State.E;
                            }
                            break;
                        }
                    case State.S36:
                        {
                            if (ligne == ';')
                            {
                                state = State.F;
                            }
                            else if (ligne == '>' || ligne == '<')
                            {
                                state = State.S27;
                            }
                            else if (MathOpers.Contains(ligne))
                            {
                                state = State.S11;
                            }
                            else if (ligne == ' ')
                            {
                                state = State.S18;
                            }
                            else
```

```
                            {
                                error = new ExceptionResult("S36 ошибка.ожидался пробел, ;, + -
/ * # = <= >=", i);
                                state = State.E;
                            }
                            break;
                        }
                }
                if (FinishTerm)
                {
                    if (termType == TermType.Int)
                    {
                        if (int.TryParse(Term.Replace(" ", ""), out int count))
                        {
                            if (!consts.Contains(Term.Replace("+", "")))
                            {
                                if ((count > 32767) || (count < -32767))
                                {
                                    error = new ExceptionResult("÷елое число должно принадлежать
интервалу [-32767, 32767]", IndexError);
                                    state = State.E;
                                }
                                else
                                {
                                    consts.Add(Term.Replace(" ", "").Replace("+", ""));
                                }
                            }
                            Term = "";
                        }
                    }
                    else if (termType == TermType.Double)
                    {
                        if (!consts.Contains(Term.Replace("+", "")))
                        {
                            consts.Add(Term.Replace(" ", "").Replace("+", ""));
                        }
                        int z = 0;
                        while (Term[z] != '.') z++;
                        if (Term.Length - z > 15)
                        {
                            error = new ExceptionResult("ћантисса не должна превышать 15
символов!", IndexError);
                            state = State.E;
                        }
                        Term = "";
                    }
                    else
                    {
                        if (ReserveWords.Contains(Term))
                        {
                            error = new ExceptionResult("Оельзя использовать зарезервированные
слова в качестве идентификатора", IndexError);
                            state = State.E;
                        }
                        else if (Term.Length > 8)
                        {
                            error = new ExceptionResult("»дентификатор не далжен быть больше 8
символов", IndexError);
                            state = State.E;
                        }
                        else
                        {
                            if (!vars.Contains(Term))
                            {
                                vars.Add(Term);
```

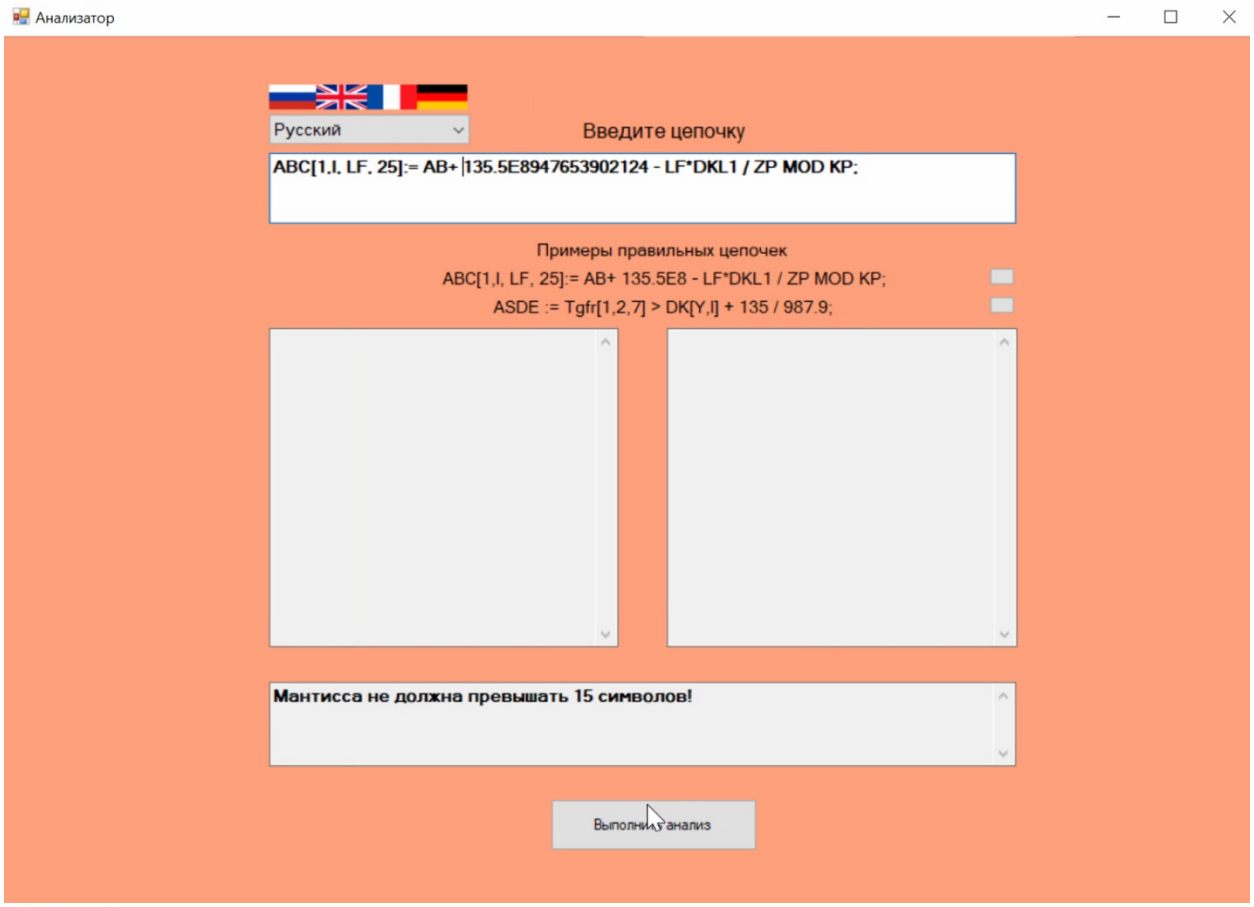
```
                            }
                            Term = "";
                        }
                    }
                }
                i++;
            }
            return new CheckingResult(consts, vars, error);
        }
    }
}

class CheckingResult
{
    ExceptionResult error;
    List<String> consts;
    List<String> vars;
    public CheckingResult(List<String> consts, List<String> vars, ExceptionResult error)
    {
        this.vars = vars;
        this.consts = consts;
        this.error = error;
    }

    public List<string> Consts
    {
        get { return consts; }
        set { consts = value; }
    }
    public List<string> Vars
    {
        get { return vars; }
        set { vars = value; }
    }
    public ExceptionResult Error
    {
        get { return error; }
        set { error = value; }
    }
}

class ExceptionResult
{
    string description;
    int pos;
    public ExceptionResult(string description, int pos)
    {
        this.description = description;
        this.pos = pos;
    }
    public int Pos { get { return pos; } }
    public string Description
    {
        get { return description; }
    }
}
```

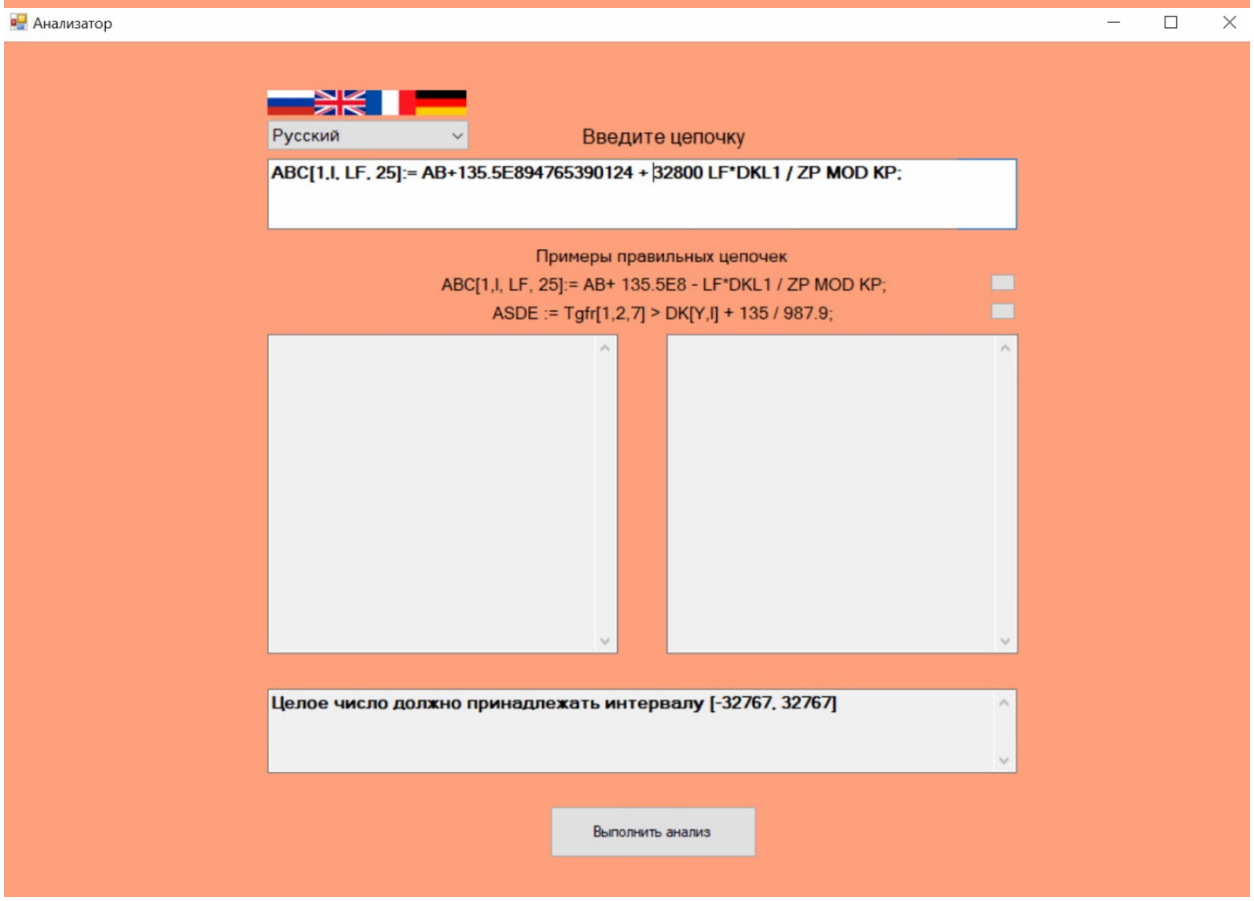
Класс пользовательского интерфейса

```
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.Reflection;
using System.Resources;
using System.Globalization;

namespace Analizator
{
    public partial class ananlyzer_19 : Form
    {
        public ananlyzer_19()
        {
            InitializeComponent();
            comboBox1.SelectedIndex = 0;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            CheckingResult result = Controller.CheckString(textBox1.Text);
            textBox2.Text = null;
            textBox3.Text = null;
            textBox4.Text = null;
            if (result.Error == null)
            {
                textBox2.Text = "—писок идентификаторов:\r\n\r\n";
                for (int i = 0; i < result.Vars.Count; ++i)
                {
                    textBox2.Text += result.Vars.ElementAt(i) + "\r\n";
                }
                textBox3.Text += "—писок констант:\r\n\r\n";
                for (int i = 0; i < result.Consts.Count; ++i)
                {
                    textBox3.Text += result.Consts.ElementAt(i) + "\r\n";
                }
            }
            else
            {
                textBox4.Text = result.Error.Description;
                textBox1.Focus();
                textBox1.SelectionStart = result.Error.Pos;
            }

        }

        private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (comboBox1.SelectedIndex == 0)
            {
                CultureInfo ci = new CultureInfo("ru-RU");
                Assembly a = Assembly.Load("Analizator8");
                ResourceManager rm = new ResourceManager("Analizator8.Lang-Ru", a);
                label1.Text = rm.GetString("String1", ci);
                label2.Text = rm.GetString("String2", ci);
                button1.Text = rm.GetString("String3", ci);
            }
            if (comboBox1.SelectedIndex == 1)
            {
```

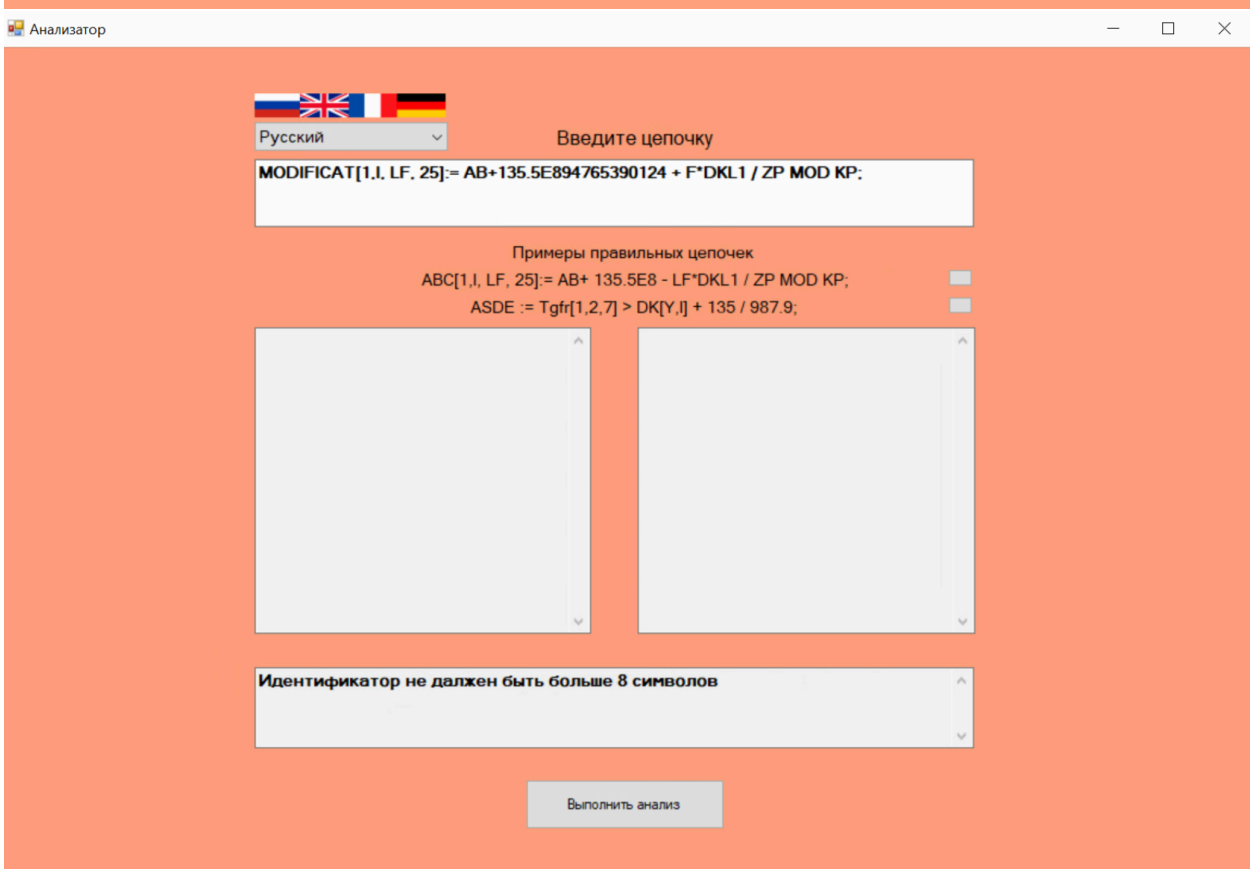
```
                CultureInfo ci = new CultureInfo("en-US");
                Assembly a = Assembly.Load("Analizator8");
                ResourceManager rm = new ResourceManager("Analizator8.Lang-En", a);
                label1.Text = rm.GetString("String1", ci);
                label2.Text = rm.GetString("String2", ci);
                button1.Text = rm.GetString("String3", ci);
            }
            if (comboBox1.SelectedIndex == 2)
            {
                CultureInfo ci = new CultureInfo("fr-FR");
                Assembly a = Assembly.Load("Analizator8");
                ResourceManager rm = new ResourceManager("Analizator8.Lang-Fr", a);
                label1.Text = rm.GetString("String1", ci);
                label2.Text = rm.GetString("String2", ci);
                button1.Text = rm.GetString("String3", ci);
            }
            if (comboBox1.SelectedIndex == 3)
            {
                CultureInfo ci = new CultureInfo("de-DE");
                Assembly a = Assembly.Load("Analizator8");
                ResourceManager rm = new ResourceManager("Analizator8.Lang-De", a);
                label1.Text = rm.GetString("String1", ci);
                label2.Text = rm.GetString("String2", ci);
                button1.Text = rm.GetString("String3", ci);
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            textBox1.Text = "ABC[1,I, LF, 25]:= AB+ 135.5E8 - LF*DKL1 / ZP MOD KP;";
        }

        private void button3_Click(object sender, EventArgs e)
        {
            textBox1.Text = "ASDE := Tgfr[1,2,7] > DK[Y,I] + 135 / 987.9;";
        }
    }
}
```

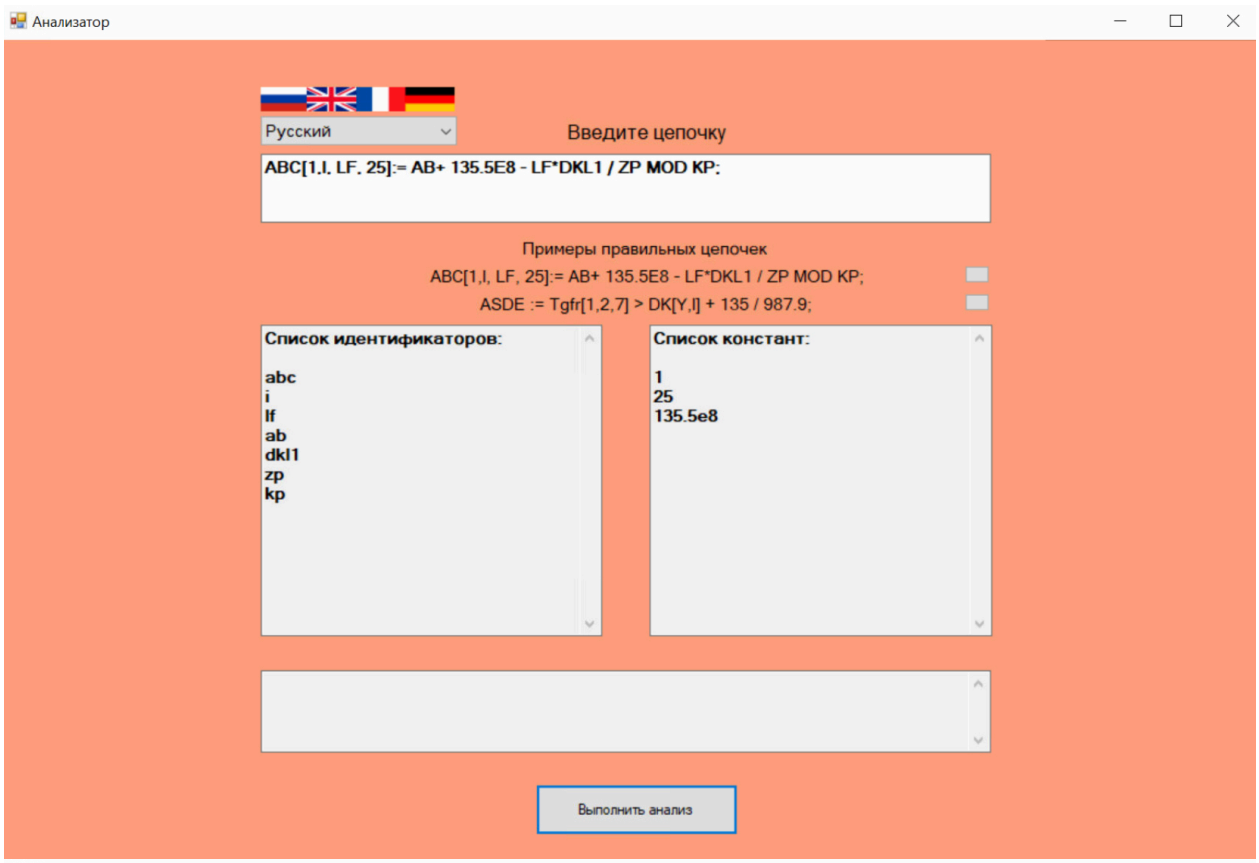
Результаты работы программы:

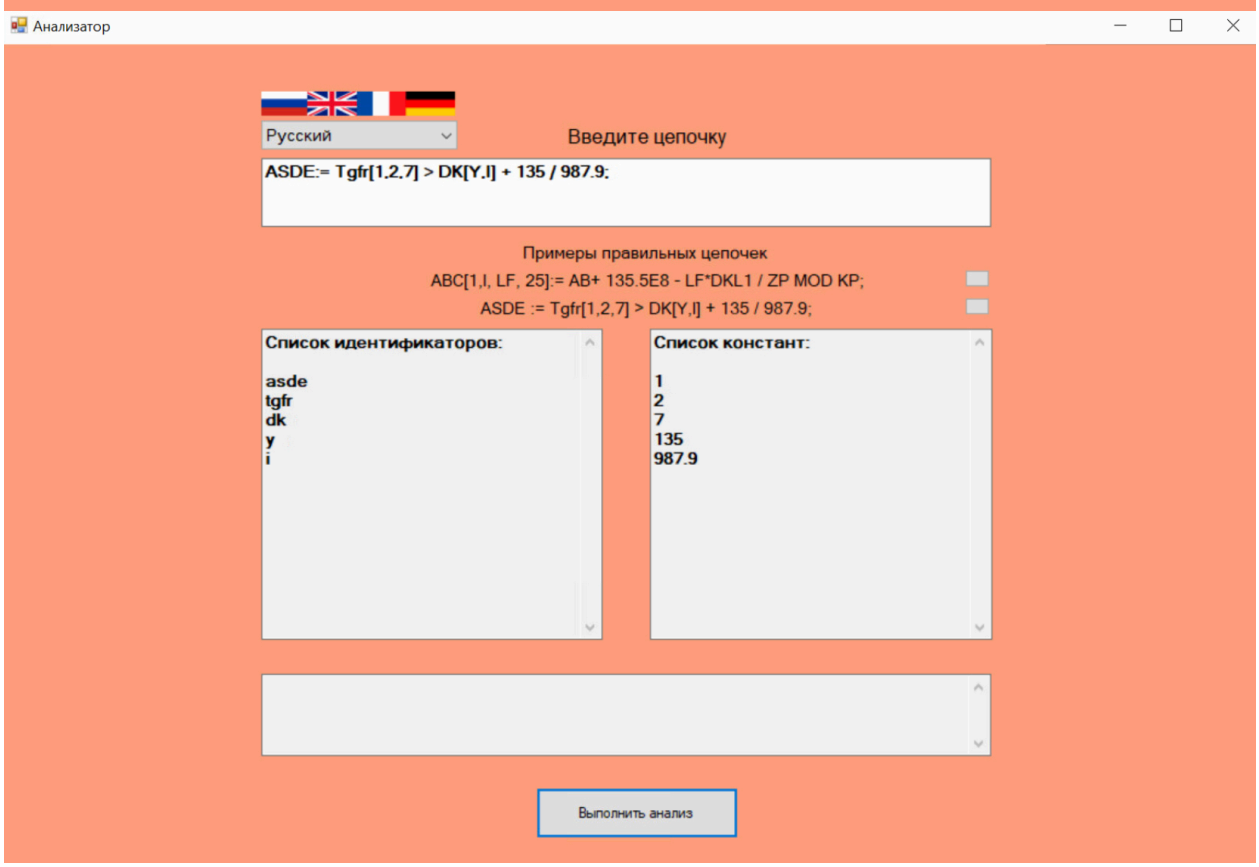





Анализатор
Русский
Введите цепочку
MOD[1,I, LF, 25]:= AB+135.5E894765390124 + 3800 +LF*DKL1 / ZP MOD KP;
Примеры правильных цепочек
ABC[1,I, LF, 25]:= AB+ 135.5E8 - LF*DKL1 / ZP MOD KP;
ASDE := Tgfr[1,2,7] > DK[Y,I] + 135 / 987.9;
Нельзя использовать зарезервированные слова в качестве идентификатора
Выполнить анализ


Анализатор
Русский
Введите цепочку
MODIFICAT[1,I, LF, 25]:= AB+135.5E894765390124 + F*DKL1 / ZP MOD KP;
Примеры правильных цепочек
ABC[1,I, LF, 25]:= AB+ 135.5E8 - LF*DKL1 / ZP MOD KP;
ASDE := Tgfr[1,2,7] > DK[Y,I] + 135 / 987.9;
Идентификатор не далжен быть больше 8 символов
Выполнить анализ


Анализатор
Русский
Введите цепочку
ABC[1,I, LF, 25]:= AB+ 135.5E8 - LF*DKL1 / ZP MOD KP;
Примеры правильных цепочек
ABC[1,I, LF, 25]:= AB+ 135.5E8 - LF*DKL1 / ZP MOD KP;
ASDE := Tgfr[1,2,7] > DK[Y,I] + 135 / 987.9;
Список идентификаторов:
abc
i
lf
ab
dkl1
zp
kp
Список констант:
1
25
135.5e8
Выполнить анализ


Анализатор
Русский
Введите цепочку
ASDE:= Tgfr[1,2,7] > DK[Y,I] + 135 / 987.9;
Примеры правильных цепочек
ABC[1,I, LF, 25]:= AB+ 135.5E8 - LF*DKL1 / ZP MOD KP;
ASDE := Tgfr[1,2,7] > DK[Y,I] + 135 / 987.9;
Список идентификаторов:
asde
tgfr
dk
y
i
Список констант:
1
2
7
135
987.9
Выполнить анализ