Министерство образования и науки Российской Федерации

Федеральное Государственное Бюджетное Образовательное Учреждение

Высшего Профессионального Образования

Тамбовский Государственный Технический Университет

Кафедра

Отчёт по лабораторной работе №3

дисциплина «Лингвистические средства вычислительных систем»

Вариант 7

Выполнил студент гр.

.

Проверил:.

Тамбов 20

**Задание:***Разработать программу синтаксического анализа методом операторного предшествования. В качестве исходных данных использовать последовательность лексем, полученных в результате работы лексического анализатора (Л.Р №1).*

**Отрывок программы:**

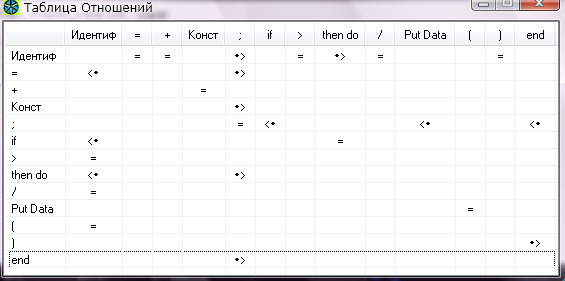
z=z+1;

if z>n then do z=z/n;

Put Data(z)

end;

**Таблица предшествования**

****

Работа программы основана на изначально подготовленной таблице приоритетов. Смысл метода в постепенном сворачивании составляющих исходного текста, например есть A+B кусочек текста, внем знак плюс будет главнее А и В, следовательно возможна сборка в более короткую строку. Так же и в коде, слево на право – ищем первый закрывающий знак ‘\*>’, далее, справо на лево ищем знак открытия ‘<\*’, делаем сборку.

**Грамматика**

<Список операторов> -> <оп. присваивания> /; / <оп. условия>

<оп. присваивания> -> <идентификатор>/ = / <арифм. выражение>

<идентификатор> -> z/ n

<арифм. выражение> -> <слагаемое>/ <частное>

<слагаемое> -> <идентификатор>/ + / <константа>

<константа> -> 1

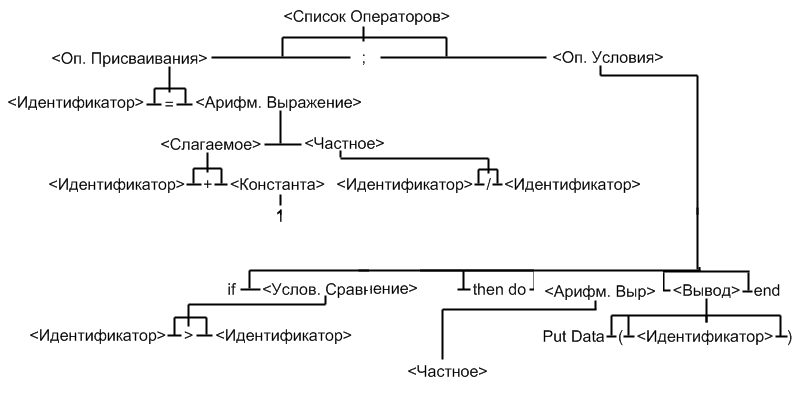
<частное> -> <идентификатор>/ ***/*** / <идентификатор>

<оп. условия> -> if / <услов. сравнение>/ then do / <частное> / <вывод>/end

<услов. сравнение> -> <идентификатор>/ > / <идентификатор>

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Идентиф | = | + | Конст | ; | if | > | then do | / | Put Data | ( | ) | end |
| Идентиф |  | **=** | **=** |  | **>** |  | **=** | **>** | **=** |  |  | **=** |  |
| = | **<** |  |  |  | **>** |  |  |  |  |  |  |  |  |
| + |  |  |  | **=** |  |  |  |  |  |  |  |  |  |
| Конст |  |  |  |  | **>** |  |  |  |  |  |  |  |  |
| ; |  |  |  |  | **=** | **<** |  |  |  | **<** |  |  | **<** |
| if | **<** |  |  |  |  |  |  | **=** |  |  |  |  |  |
| > | **=** |  |  |  |  |  |  |  |  |  |  |  |  |
| then do | **<** |  |  |  | **>** |  |  |  |  |  |  |  |  |
| / | **=** |  |  |  |  |  |  |  |  |  |  |  |  |
| Put Data |  |  |  |  |  |  |  |  |  |  | **=** |  |  |
| ( | **=** |  |  |  |  |  |  |  |  |  |  |  |  |
| ) |  |  |  |  |  |  |  |  |  |  |  |  | **>** |
| end |  |  |  |  | **>** |  |  |  |  |  |  |  |  |

Грамматическое дерево



**Исходный код программы:**

fVosxodParser.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Windows.Forms;

//LPO-3 Восходящий парсер

namespace Карась\_Project

{

class fVosxodParser

{

private string outMessage;//Переменная для хранения конечного результа работы парсера

private string[] arrPriority = {"="/\*0\*/, "<•"/\*1\*/, "="/\*2\*/, "="/\*3\*/, "•>"/\*4\*/, "<•"/\*5\*/, "<•"/\*6\*/,

"="/\*7\*/, "="/\*8\*/, "•>"/\*9\*/, "<•"/\*10\*/,"="/\*11\*/,"<•"/\*12\*/,"="/\*13\*/,

"="/\*14\*/,"•>"/\*15\*/,"<•"/\*16\*/,"="/\*17\*/, "="/\*18\*/,"="/\*19\*/,"•>"/\*!20\*/,"•>"/\*!21\*/

};//Первоначальная растановка приоритетов

private string[] arrZapas = { "•>"/\*= ;\*/, "="/\*IF THEN\_DO\*/, "<•"/\*; END\*/ };//ДопЗначения при колизии на 4-5-6-7 проходах

/\*Z"="="<•"Z"="+"="1"•>";

"<•"IF"<•"Z"=">"="N"•>"THEN\_DO"<•"Z"="="<•"Z"="/"="N"•>";

"<•"PUT\_DATA"="("="Z"=")"•>"END"•>";\*/

public static string[] words;//масив для слов

string text;//для строки

void BufferWords()

{//Вытаскиваем из буфера Текст "как есть" и переводим в массив

text = FormMain.BuffWords.DataBuffer;//Пишем в сторку весь текст из буфера

char[] razdelenie = { ' ', '\n' };//Символы переноса строки выкидываем

words = text.Split(razdelenie); //Делим строку на ПОДСТРОКИ

for (int i = 0; i < words.Length; i++)

{//Поиск then и do

if (words[i] == "then")

{

if (words[i + 1] == "do\r" || words[i + 1] == "do")

{//Обьединить then и do в одну ячейку масива

words[i] += " do";//then do

for (int j = i + 1; j < words.Length - 1; j++)

{//Делаем сдвиг в лево оставшейся части после обьединения

words[j] = words[j + 1];

}

words[words.Length - 1] = null;//Последнюю занулим

}

}

if (words[i] == "Put")//i=[18]

{

if (words[i + 1] == "Data")

{//Обьединить Put и Data в одну ячейку

words[i] += " Data";//Put Data

}

for (int j = i + 1; j < words.Length-1; j++)

{//Сдвиг влево на 1 от 18 до конца тк обьединили ячейку выше

words[j] = words[j + 1];

}

outMessage += 0 + "] ";//абзац 0]

foreach (string word in words)

{//ПРОВЕРКА ВЫВОДА

outMessage += word + " ";//Вывод Эталона строки

} outMessage += Environment.NewLine;//Делаем переход на новую строку

}

}

}

void OutRes()

{//Три прохода с ноля и плюс еще четыре по массиву страховки

outMessage += 1 + "] "; perebor(); outMessage += Environment.NewLine;//Строка Z=Z+1;

outMessage += 2 + "] "; PereborTwo(); outMessage += Environment.NewLine;//Строка if...;

outMessage += 3 + "] "; PereborFree(); outMessage += Environment.NewLine;//Строка PutData();

}

void perebor()

{//z=z+1;

for (int i = 0; i < 9; i++)

{//перебираем элементы

if (arrPriority[i] == "•>")

{//i=[4]

//MessageBox.Show(i.ToString());

for (int j = i; j > 0; j--)

{//перебор в обратную сторону

if (arrPriority[j] == "<•")

{//j=[1]

//MessageBox.Show(j.ToString());

for (; j+1 < i+1; j++)//с 2 до 4 включительно

{//j по приоритетам, а для текста необходим j+1 сдвиг

words[j+1] = null; //Нулим уже использованые значения текстов

arrPriority[j] = null; //Нулим уже использованые значения приоритетов

}

foreach (string word in words)//Перебор всего массива

{//Вывод результата обработки первой строки

if (word != null)//Нулы пропускаем

outMessage += word + " ";//Вывод

}

if (words[7]==null && words[9]==null)

{//Направляем на запасную таблицу

arrPriority[6] = arrZapas[1];

//arrPriority[]

}

}

continue;

}

}

}

}

void PereborTwo()

{//if[6]

for (int i = 12; i < 16; i++)

{//перебираем элементы

if (arrPriority[i] == "•>")

{//i=[4]

MessageBox.Show(i.ToString());

for (int j = i; j > 0; j--)

{//перебор в обратную сторону

if (arrPriority[j] == "<•")

{//j=[1]

MessageBox.Show(j.ToString());

for (; j + 1 < i + 1; j++)//с 2 до 4 включительно

{//j по приоритетам, а для текста необходим j+1 сдвиг

words[j + 1] = null; //Нулим уже использованые значения текстов

arrPriority[j] = null; //Нулим уже использованые значения приоритетов

}

foreach (string word in words)//Перебор всего массива

{//Вывод результата обработки первой строки

if (word != null)//Нулы пропускаем

outMessage += word + " ";//Вывод

}

}

else

{//Иначе нету отбатной скобки - необходим запасной вариант

if(words[13]==null && words[14]==null && words[15]==null)

{//z=z/n;

arrPriority[12] = arrZapas[0];//Востанавлииваем \*> меж z= && ;

}

if(words[11]==null && words[13]==null && words[14]==null && words[15]==null)

if(arrPriority[15]==null)

{//Востанавливаем отношение между then do и ;

arrPriority[6] = arrZapas[1];

arrPriority[15] = arrZapas[0];//then do \*> ;

}

}

continue;

}

}

}

}

void PereborFree()

{//Закрываем End

for (int i = 17; i < arrPriority.Length; i++)

{//перебираем элементы

if (arrPriority[i] == "•>")

{//i=[4]

MessageBox.Show(i.ToString());

for (int j = i; j > 0; j--)

{//перебор в обратную сторону

if (arrPriority[j] == "<•")

{//j=[1]

MessageBox.Show(j.ToString());

for (; j + 1 < i + 1; j++)//с 2 до 4 включительно

{//j по приоритетам, а для текста необходим j+1 сдвиг

words[j + 1] = null; //Нулим уже использованые значения текстов

arrPriority[j] = null; //Нулим уже использованые значения приоритетов

}

foreach (string word in words)//Перебор всего массива

{//Вывод результата обработки первой строки

if (word != null)//Нулы пропускаем

outMessage += word + " ";//Вывод

}

if (arrPriority[19] == null && arrPriority[20] != null && words[21]!=null)

{//Востановим ) <\* end;

arrPriority[20] = arrZapas[2];//) <\* end;

}

}

continue;

}

}

}

}

public string inOutTextBox()

{//Выводим результаты в textBoxOut

BufferWords();//ПЕРВЫЙ ШАГ - Обработаем текст с файла

//=================================================//

FormMain formMain = new FormMain();//Создали ссылку на главное Окно

OutRes();//Отработа метод Парсера

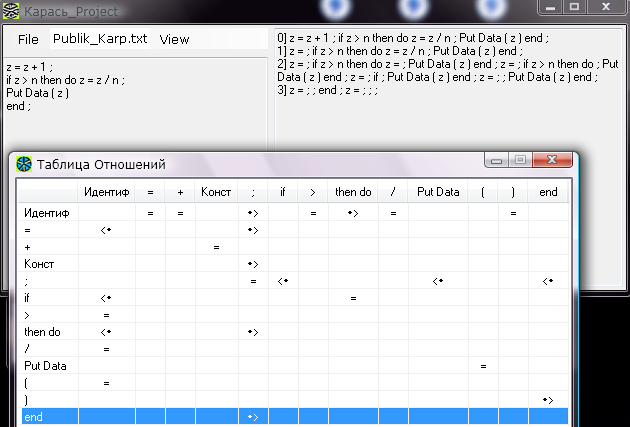
return formMain.textBoxOut.Text = outMessage;//Вывод результатов работы класса в текстовую область справо

}

}//End Class

}

**Результат работы:**

****