**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ**

**Федеральное государственное бюджетное образовательное**

**учреждение высшего профессионального образования**

**«Пензенский государственный университет»**

(**ФГБОУ ВПО «Пензенский государственный университет») \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Кафедра «Математическое обеспечение и применение ЭВМ»

**Отчёт по лабораторной работе №6**

**Отчёт по дисциплине**

**«Теория вычислительных процессов»**

Автор работы Серов Е. С.

Шалдыбин А. Ю.

Группа 15ВП1

Направление 09.03.04

Работа сдана «\_\_\_»\_\_\_\_\_\_\_\_2016г.

Работа защищена «\_\_\_»\_\_\_\_\_\_\_\_2016 г.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2016

Задание:

Разработать магазинный автомат для заданного языка, осуществить программную реализацию и тестирование.

Описание языка:

<Программа> ::= <Объявление переменных> <Описание вычислений>

<Описание вычислений> ::= BEGIN <cписок присваиваний> END

<Объявление переменных> ::= VAR <список переменных> : тип ;

<Список переменных> ::= <Идент>|<Идент>,<Список переменных>

<Список присваиваний> ::= <Присваивание>|<Присваивание> <Список присваиваний>

<Присваивание> ::= <Идент> = <Выражение>;

<Выражение> ::= <Ун.оп.> <Подвыражение> | <Подвыражение>

<Подвыражение> ::= (<Выражение>) | <Операнд> | <Подвыражение>

<Бин.оп.><Подвыражение>

<Ун.оп> ::= вид

<Бин.оп.> ::= вид

<Операнд> ::= <Идент>|<Конст>

<Идент> ::= <Буква><Идент>|<Буква>

<Конст> ::= вид

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № варианта | Тип переменных | Вид  <Ун. оп.> | Вид  <Бин. оп.> | Вид <конст.> | Макс. длина идентификатора | Распознаватель |
| 8 | LOGICAL | .NOT. | .AND. | .OR. | .EQU. | 0|1 | 9 | 1 |

Грамматика:

I - программа

A - объявление переменных

B - описание вычеслений

C - список присваиваний

D - список переменных

E - идентификатор

F - присваивание

G - выражение

H - ун. операция

J - подвыражение

K - операнд

L - бинарные операции

M - константа

N - буква

Vt = {a,b…,y,z,0,1,:,;,,,=,(,)}

Va ={I,A,B,V,D,E,F,G,H,J,K,L,M,N}

R={I→AB,

B→”BEGIN”C”END”,

A→”VAR”D”:LOGICAL;”,

D→E|ED,

C→F|FC,

F→E”=”G,

G→HJ|J,

J→”(”G”)”|K|J,

H→”.NOT.”,

L→”.AND.”|”.OR.”|”.EQU.”,

E→N|NE, E→0|1 }

Блок-схема программы:



Листинг программы:

***Класс Parser:***

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.util.Stack;

**public** **class** Laba5 {

**public** **static** **final** **int** ***START*** = 0,

***VARS\_INIT*** = 1,

***ID*** = 2,

***START\_PROCESS*** = 3,

***PROCESS*** = 4,

***EXPRESSION*** = 5,

***SUB\_EXPRESSION*** = 6;

**public** Stack<Integer> stack = **new** Stack<Integer>();

**public** Laba5() **throws** IOException{

BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(System.***in***));

String line;

stack.push(***START***);

**while** ((line = reader.readLine()).length() != 0){

**try**{

parsing(line);

} **catch**(ParsException e){

System.***out***.println("Ошибка в строке:");

System.***out***.println(line);

System.***out***.println("Стек:");

**for** (**int** i=0; i<stack.size(); i++){

String s = "";

**switch**(stack.get(i)){

**case** 0: s = "Программа"; **break**;

**case** 1: s = "Объявление переменных"; **break**;

**case** 2: s = "Идентификатор"; **break**;

**case** 4: s = "Описание вычислений"; **break**;

**case** 5: s = "Выражение"; **break**;

**case** 6: s = "Подвыражение"; **break**;

}

System.***out***.print(s + " -> ");

}

**break**;

}

}

reader.close();

}

**public** **void** parsing(String line) **throws** ParsException{

//System.out.println("Parsing: " + line);

//System.out.println(stack.toString());

//System.out.println();

**switch** (stack.peek()){

**case** ***START***: start(line); **break**;

**case** ***VARS\_INIT***: vars\_init(line); **break**;

**case** ***ID***: id(line); **break**;

**case** ***START\_PROCESS***: start\_process(line); **break**;

**case** ***PROCESS***: process(line); **break**;

**case** ***EXPRESSION***: expression(line); **break**;

**case** ***SUB\_EXPRESSION***: subExpression(line); **break**;

}

}

**public** **void** start(String line) **throws** ParsException{

stack.push(***VARS\_INIT***);

parsing(line);

stack.push(***START\_PROCESS***);

}

**public** **void** vars\_init(String line) **throws** ParsException{

**if** (!line.substring(0, 4).equals("VAR ")) **throw** **new** ParsException();

line = line.substring(4);

String[] str = line.split(":");

**if** (str.length != 2) **throw** **new** ParsException();

String[] ids = str[0].split(",");

**for** (**int** i=0; i<ids.length; i++){

stack.push(***ID***);

parsing(ids[i]);

}

**if** (!str[1].equals("LOGICAL;")) **throw** **new** ParsException();

stack.pop();

}

**public** **void** id(String line) **throws** ParsException{

String allow = "qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFGHJKLZXCVBNM";

**for** (**int** i=0; i<line.length(); i++){

**boolean** find = **false**;

**for** (**int** j=0; j<allow.length(); j++)

**if** (line.charAt(i) == allow.charAt(j)){

find = **true**;

**break**;

}

**if** (!find) **throw** **new** ParsException();

}

**if** (line.length() > 12) **throw** **new** ParsException();

stack.pop();

}

**public** **void** start\_process(String line) **throws** ParsException{

**if** (!line.equals("BEGIN")) **throw** **new** ParsException();

stack.push(***PROCESS***);

}

**public** **void** process(String line) **throws** ParsException{

**if** (line.equals("END")){

stack.pop();

**return**;

}

String[] str = line.split("=");

**if** (str.length != 2) **throw** **new** ParsException();

**for** (**int** i=0; i<str.length; i++) str[i].trim();

stack.push(***ID***);

parsing(str[0]);

**if** (str[1].charAt(str[1].length()-1) != ';') **throw** **new** ParsException();

stack.push(***EXPRESSION***);

parsing(str[1].substring(0, str[1].length()-1));

}

**public** **void** expression(String line) **throws** ParsException{

**if** ((line.length() >= 5) && (line.substring(0, 5).equals(".NOT."))) line = line.substring(5);

stack.push(***SUB\_EXPRESSION***);

parsing(line);

stack.pop();

}

**public** **void** subExpression(String line) **throws** ParsException{

**if** (line.charAt(0) == '('){

stack.push(***EXPRESSION***);

parsing(line.substring(1, line.length()-1));

} **else** **if**(line.indexOf('.') != -1){

**if** ((!line.substring(line.indexOf('.')+1, line.indexOf('.', line.indexOf('.')+1)).equals("AND")) &&

(!line.substring(line.indexOf('.')+1, line.indexOf('.', line.indexOf('.')+1)).equals("OR")) &&

(!line.substring(line.indexOf('.')+1, line.indexOf('.', line.indexOf('.')+1)).equals("EQU"))) **throw** **new** ParsException();

stack.push(***SUB\_EXPRESSION***);

parsing(line.substring(0, line.indexOf('.')));

stack.push(***SUB\_EXPRESSION***);

parsing(line.substring(line.indexOf('.', line.indexOf('.')+1)+1, line.length()));

} **else** {

**if** (line.length() == 1){

**if** (!line.equals("0") && !line.equals("1")) **throw** **new** ParsException();

} **else** {

stack.push(***ID***);

parsing(line);

}

}

stack.pop();

}

**public** **static** **void** main(String[] args) **throws** IOException{

**new** Laba5();

}

}

Тестирование:







