**Manual Do Programador**

1. **Gramática Utilizada:**

<programa> ::= <decl\_list>

<decl\_list> ::= <decl>

<decl> ::= <var\_decl><decl>|<fun\_decl><decl>|<vazio>

<var\_decl> ::= <type\_spec>ID<decl\_1>

<type\_spec> ::= "void"|"float"|"char"|"int"|"double"

<decl\_1> ::= ";"|"[""]";"|"="<exp>";"

<fun\_decl> ::= <type\_spec>ID"("<params>")"<fun\_decl\_1>

<fun\_decl\_1> ::= ";"|<com\_stmt>

<params> ::= <param\_list\_2>|"void"|")"

<param\_list\_2> ::= <param><param\_list\_1>|<vazio>

<param\_list\_1> ::= ","<param><param\_list\_1>|<vazio>

<param> ::= <type\_spec>ID<param\_1>

<param\_1> ::= "[""]"|<vazio>

<stmt\_list> ::= <stmt><stmt\_list>|<vazio>

<stmt> ::= <exp\_stmt>|<com\_stmt>|<if\_stmt>|<while\_stmt>|<return\_stmt>|<break\_stmt>|for\_stmt|printf\_stmt|scanf\_stmt

<exp\_stmt> ::= ";"|<exp>";"

<while\_stmt> ::= "while""("<exp>")"<stmt>

<com\_stmt> ::= "{"<local\_decls><stmt\_list>"}"

<local\_decls> ::= <type\_spec>ID<local\_decl\_1>|<vazio>

<local\_decl\_1> ::= ";"<local\_decls>|"[""]";"<local\_decls>|"="<exp>";"<local\_decls>

<if\_stmt> ::= "if""("<exp>")"<stmt><if\_stmt\_1>

<if\_stmt\_1> ::= <vazio>|"else"<stmt>

<for\_stmt> ::= "for""("<for\_stmt\_1>";"<exp>";"<exp>")"<stmt>

<for\_stmt\_1> ::= <type\_spec>ID=<exp>|<exp>

<return\_stmt> ::= "return"<return\_stmt\_1>

<return\_stmt\_1> ::= ";"|<exp>";"

<break\_stmt> ::= "break"";"

<exp> ::= ID<exp\_1><exp\_3>|"+"<exp><exp\_3>|"-"<exp><exp\_3>|"!"<exp><exp\_3>|FLOAT\_LIT<exp\_3>|CHAR\_LIT<exp\_3>|INT\_LIT<exp\_3>

<exp\_3> ::= "||"<exp><exp\_3>|"="<exp><exp\_3>|"=="<exp><exp\_3>|"<="<exp><exp\_3>|"<"<exp><exp\_3>|">"<exp><exp\_3>|">="<exp><exp\_3>|"&&"<exp><exp\_3>|"+"<exp><exp\_3>|"-"<exp><exp\_3>|"%"<exp><exp\_3>|"\*"<exp><exp\_3>|"/"<exp><exp\_3>|"+="|"-="|<vazio>

<exp\_1> ::= "["<exp>"]"<exp\_2>|".""sizeof"|"("<args>")"|"++"|"--"|<vazio>

<exp\_2> ::= "="<exp>|<vazio>

<arg\_list> ::= <exp><arg\_list\_1>

<arg\_list\_1> ::= ","<exp><arg\_list\_1>|<vazio>

<args> ::= <arg\_list>|<vazio>

**2. Conversão da Gramática em Código:**

package entities;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

public class SintaticSemanticAnalex {

    int pointer = 0;

    private String error = "";

    private Lexema readedLexema;

    private ArrayList<Lexema> lexema;

    private final List<String> TYPE\_SPEC\_ARRAY = Arrays.asList("void", "float", "char", "int", "double");

    public String getError() {

        return this.error;

    }

    public Lexema readLexema() {

        if (this.pointer >= 0 && lexema.size() > 0 && this.pointer < lexema.size() && this.error.equals("")) {

            this.readedLexema = lexema.get(this.pointer);

            this.pointer = this.pointer + 1;

            return this.readedLexema;

        }

        this.pointer = -1;

        return new Lexema("", "", -1);

    }

    public void printError(String text) {

        if (this.error.equals("") && this.readedLexema != null) {

            this.error = text + " na linha " + this.readedLexema.getLine();

        }

    }

    public ArrayList<Lexema> filterDirectives(ArrayList<Lexema> lexema) {

        ArrayList<Lexema> newLexema = new ArrayList<Lexema>();

        int index = 0;

        for (Lexema item : lexema) {

            if (!item.getToken().equals(Token.TK\_DIRECTIV) &&

                    !item.getToken().equals(Token.TK\_CARD) &&

                    !item.getToken().equals(Token.TK\_CARDC)) {

                item.setIndex(index);

                newLexema.add(item);

            }

            index++;

        }

        return newLexema;

    }

    public SintaticSemanticAnalex(ArrayList<Lexema> lexema) {

        this.lexema = filterDirectives(lexema);

        program();

    }

    public void program() {

        decl\_list();

    }

    public void decl\_list() {

        this.decl();

    }

    public void decl() {

        if (this.var\_decl()) {

            // System.out.println("Declaraco de variavel valida");

            if (this.pointer < this.lexema.size()) {

                this.decl();

            }

        } else if (this.fun\_decl()) {

            // System.out.println("Declaracao de funcao valida");

            if (this.pointer < this.lexema.size()) {

                this.decl();

            }

        } else {

            // System.out.println("\*\*\*Fim de programa\*\*\*");

        }

    }

    public boolean var\_decl() {

        if (this.type\_spec()) {

            return true;

        } else {

            this.pointer -= 1;

            return false;

        }

    }

    public boolean type\_spec() {

        Lexema item = readLexema();

        if (TYPE\_SPEC\_ARRAY.contains(item.getLexema())) {

            if (this.type\_spec\_id()) {

                return true;

            } else {

                this.pointer -= 1;

                return false;

            }

        } else {

            this.pointer -= 1;

            this.printError("[type\_spec] Esperava receber um tipo de dados int, float, char... ");

            return false;

        }

    }

    public boolean type\_spec\_id() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_ID) || item.getToken().equals(Token.TK\_VAR)) {

            item = readLexema();

            if (item.getToken().equals(Token.TK\_FDI)) {

                return true;

            } else if (item.getToken().equals(Token.TK\_APR)) {

                item = readLexema();

                if (item.getToken().equals(Token.TK\_FPR)) {

                    item = readLexema();

                    if (item.getToken().equals(Token.TK\_FDI)) {

                        return true;

                    } else {

                        this.pointer -= 1;

                        this.printError("[type\_spec\_id] Esperava receber ;");

                        return false;

                    }

                } else {

                    this.pointer -= 1;

                    this.printError("[type\_spec\_id] Esperava receber ]");

                    return false;

                }

            } else if (item.getToken().equals(Token.TK\_IG)) {

                if (this.exp()) {

                    item = readLexema();

                    if (item.getToken().equals(Token.TK\_FDI)) {

                        return true;

                    } else {

                        this.pointer -= 1;

                        this.printError("[type\_spec\_id-exp] Esperava receber ;");

                        return false;

                    }

                } else {

                    this.printError("[type\_spec\_id-exp] Expressao invalida");

                    return false;

                }

            } else {

                this.pointer -= 1;

                return false;

            }

        } else {

            this.pointer -= 1;

            this.printError("[type\_spec\_id] Esperava receber um identificador ou uma variavel ");

            return false;

        }

    }

    public boolean fun\_decl() {

        Lexema item = readLexema();

        if (TYPE\_SPEC\_ARRAY.contains(item.getLexema())) {

            return this.fun\_decl\_id();

        } else {

            this.printError("[fun\_decl] Esperava receber um tipo de dados ");

            pointer--;

            return false;

        }

    }

    public boolean fun\_decl\_id() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_ID) || item.getToken().equals(Token.TK\_VAR)) {

            return this.fun\_decl\_id\_open\_parent();

        } else {

            this.printError("[fun\_decl\_id] Esperava receber um ID ");

            this.pointer -= 1;

            return false;

        }

    }

    public boolean fun\_decl\_id\_open\_parent() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_AP)) {

            if (this.fun\_decl\_id\_open\_parent\_params()) {

                if (this.fun\_decl\_id\_close\_parent()) {

                    if (this.fun\_decl\_1()) {

                        return true;

                    } else {

                        return false;

                    }

                } else {

                    this.pointer -= 1;

                    return false;

                }

            } else {

                this.pointer -= 1;

                return false;

            }

        }

        this.printError("[fun\_decl\_id\_open\_parent] Esperava receber ( ");

        this.pointer -= 1;

        return false;

    }

    public boolean fun\_decl\_id\_close\_parent() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_FP)) {

            return true;

        } else {

            this.printError("[fun\_decl\_id\_close\_parent] Esperava receber ) ");

            this.pointer -= 1;

            return false;

        }

    }

    public boolean fun\_decl\_id\_open\_parent\_params() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_VOID)) {

            return true;

        } else {

            this.pointer -= 1;

        }

        if (this.param\_list\_2()) {

            return true;

        } else {

            this.printError("[fun\_decl\_id\_open\_parent\_params]  Espera receber um parametro ou void ");

            this.pointer -= 1;

            return false;

        }

    }

    public boolean fun\_decl\_1() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_FDI)) {

            return true;

        } else {

            this.pointer -= 1;

            return this.com\_stmt(false);

        }

    }

    public boolean com\_stmt(boolean isOptional) {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_ABC)) {

            if (this.local\_decls()) {

                if (this.stmt\_list()) {

                    item = readLexema();

                    if (item.getToken().equals(Token.TK\_FCH)) {

                        return true;

                    } else {

                        if (!isOptional)

                            this.printError("[com\_stmt] Esperava receber } ");

                        this.pointer -= 1;

                        return false;

                    }

                } else {

                    this.pointer -= 1;

                    return false;

                }

            } else {

                this.pointer -= 1;

                return false;

            }

        } else {

            this.pointer -= 1;

            return false;

        }

    }

    public boolean if\_stmt() {

        Lexema item = readLexema();

        if (item.getLexema().equals("if")) {

            item = readLexema();

            if (item.getToken().equals(Token.TK\_AP)) {

                if (this.exp()) {

                    item = readLexema();

                    if (item.getToken().equals(Token.TK\_FP)) {

                        if (this.stmt()) {

                            if (this.if\_stmt\_1()) {

                                return true;

                            } else {

                                this.printError("[if\_stmt] Expressão inválida");

                                return false;

                            }

                        } else {

                            this.printError("[if\_stmt] Expressão inválida");

                            return false;

                        }

                    } else {

                        this.printError("[if\_stmt] Esperava receber )");

                        this.pointer -= 1;

                        return false;

                    }

                } else {

                    this.printError("[if\_stmt] Expresão invalida");

                    return false;

                }

            } else {

                this.printError("[if\_stmt] Esperava receber (");

                this.pointer -= 1;

                return false;

            }

        } else {

            this.pointer -= 1;

            return false;

        }

    }

    public boolean if\_stmt\_1() {

        Lexema item = readLexema();

        if (item.getLexema().equals("else")) {

            if (this.stmt()) {

                return true;

            } else {

                return false;

            }

        } else {

            this.pointer -= 1;

            return true;

        }

    }

    public boolean while\_stmt() {

        Lexema item = readLexema();

        if (item.getLexema().equals("while")) {

            item = readLexema();

            if (item.getToken().equals(Token.TK\_AP)) {

                if (this.exp()) {

                    item = readLexema();

                    if (item.getToken().equals(Token.TK\_FP)) {

                        if (this.stmt()) {

                            return true;

                        } else {

                            this.printError("[while\_stmt-stmt] Expressão invalida");

                            return false;

                        }

                    } else {

                        this.printError("[while\_stmt] Esperava receber ) ");

                        return false;

                    }

                } else {

                    this.printError("[while\_stmt-exp] Expressão invalida");

                    return false;

                }

            } else {

                this.printError("[while\_stmt] Esperava receber (");

                this.pointer -= 1;

                return false;

            }

        } else {

            this.pointer -= 1;

            return false;

        }

    }

    public boolean stmt() {

        if (this.exp\_stmt()) {

            return true;

        } else if (this.com\_stmt(true)) {

            return true;

        } else if (this.if\_stmt()) {

            return true;

        } else if (this.while\_stmt()) {

            return true;

        } else if (this.return\_stmt()) {

            return true;

        } else if (this.break\_stmt()) {

            return true;

        } else if (this.for\_stmt()) {

            return true;

        } else {

            return false;

        }

    }

    public boolean for\_stmt() {

        Lexema item = this.readLexema();

        if (!item.getLexema().equals("for")) {

            this.pointer -= 1;

            return false;

        }

        item = this.readLexema();

        if (!item.getToken().equals(Token.TK\_AP)) {

            this.pointer -= 1;

            this.printError("[for\_stmt] Esperava receber ( ");

            return false;

        }

        if (!this.for\_stmt\_1()) {

            this.printError("[exp] Expresão inválida ");

            return false;

        }

        item = this.readLexema();

        if (!item.getToken().equals(Token.TK\_FDI)) {

            this.pointer -= 1;

            this.printError("[for\_stmt] Esperava receber ; ");

            return false;

        }

        if (!this.exp()) {

            this.printError("[for\_stmt-exp] Expressão invalida");

            return false;

        }

        item = this.readLexema();

        if (!item.getToken().equals(Token.TK\_FDI)) {

            this.pointer -= 1;

            this.printError("[for\_stmt] Esperava receber ; ");

            return false;

        }

        if (!this.exp()) {

            this.printError("[for\_stmt-exp] Expressão invalida");

            return false;

        }

        item = this.readLexema();

        if (!item.getToken().equals(Token.TK\_FP)) {

            this.pointer -= 1;

            this.printError("[for\_stmt] Esperava receber ) ");

            return false;

        }

        if (!this.stmt()) {

            this.printError("[for\_stmt-stmt] Expressão invalida");

            return false;

        }

        return true;

    }

    public boolean for\_stmt\_1() {

        Lexema item = this.readLexema();

        if (TYPE\_SPEC\_ARRAY.contains(item.getLexema())) {

            item = this.readLexema();

            if (!item.getToken().equals(Token.TK\_ID)) {

                this.pointer -= 1;

                this.printError("Esperava receber um ID ");

                return false;

            }

            item = this.readLexema();

            if (!item.getToken().equals(Token.TK\_IG)) {

                this.pointer -= 1;

                this.printError("Esperava receber um = ");

                return false;

            }

            if (this.exp()) {

                return true;

            } else {

                return false;

            }

        } else {

            this.pointer -= 1;

        }

        if (this.exp()) {

            return true;

        } else {

            return false;

        }

    }

    public boolean break\_stmt() {

        Lexema item = this.readLexema();

        if (item.getToken().equals("break")) {

            item = this.readLexema();

            if (item.getLexema().equals(";")) {

                return true;

            } else {

                this.pointer -= 1;

                this.printError("[break\_stmt] Esperava receber ;");

                return false;

            }

        } else {

            this.pointer -= 1;

            return false;

        }

    }

    public boolean return\_stmt() {

        Lexema item = this.readLexema();

        if (item.getLexema().equals("return")) {

            if (this.return\_stmt\_1()) {

                return true;

            } else {

                this.printError("[return\_stmt\_1] Expressão invalida");

                return false;

            }

        } else {

            this.pointer -= 1;

            return false;

        }

    }

    public boolean return\_stmt\_1() {

        Lexema item = this.readLexema();

        if (item.getLexema().equals(";")) {

            return true;

        } else {

            this.pointer -= 1;

        }

        if (this.exp()) {

            item = this.readLexema();

            if (item.getLexema().equals(";")) {

                return true;

            } else {

                this.pointer -= 1;

                return false;

            }

        } else {

            return false;

        }

    }

    public boolean exp\_stmt() {

        Lexema item = this.readLexema();

        if (item.getToken().equals(Token.TK\_FDI)) {

            return true;

        } else {

            this.pointer -= 1;

        }

        if (this.exp()) {

            item = this.readLexema();

            if (item.getToken().equals(Token.TK\_FDI)) {

                return true;

            } else {

                this.pointer -= 1;

                return false;

            }

        } else {

            return false;

        }

    }

    public boolean stmt\_list() {

        if (this.stmt()) {

            return this.stmt\_list();

        } else {

            return true;

        }

    }

    public boolean exp() {

        Lexema item = this.readLexema();

        if (item.getToken().equals(Token.TK\_ID)) {

            if (this.exp\_1()) {

                if (this.exp\_3()) {

                    return true;

                } else {

                    this.printError("[exp-exp\_3] Expressão invalida");

                    return false;

                }

            } else {

                this.printError("[exp-exp\_1] Expressão invalida");

                return false;

            }

        } else if (item.getLexema().equals("+") || item.getLexema().equals("-") || item.getLexema().equals("!")) {

            if (this.exp()) {

                if (this.exp\_3()) {

                    return true;

                } else {

                    this.printError("[exp-exp-exp\_3] Expressão invalida");

                    return false;

                }

            } else {

                this.printError("[exp-exp] Expressão invalida");

                return false;

            }

        } else if (item.getToken().equals(Token.TK\_NF) || item.getToken().equals(Token.TK\_NI)

                || item.getToken().equals(Token.TK\_CH)) {

            if (this.exp\_3()) {

                return true;

            } else {

                this.printError("[exp-exp\_3] Expressão invalida");

                return false;

            }

        } else {

            this.pointer -= 1;

            return false;

        }

    }

    public boolean exp\_3() {

        Lexema item = this.readLexema();

        if (item.getToken().equals(Token.TK\_OR) || item.getToken().equals(Token.TK\_IG)

                || item.getToken().equals(Token.TK\_IGIG) || item.getToken().equals(Token.TK\_MENRI)

                || item.getToken().equals(Token.TK\_MENR) || item.getToken().equals(Token.TK\_MAIOR)

                || item.getToken().equals(Token.TK\_MAIOI) || item.getToken().equals(Token.TK\_AND)

                || item.getToken().equals(Token.TK\_M) || item.getToken().equals(Token.TK\_MEN)

                || item.getToken().equals(Token.TK\_PERCENT) || item.getToken().equals(Token.TK\_AST)

                || item.getToken().equals(Token.TK\_DIV)

                || item.getToken().equals(Token.TK\_MI) || item.getToken().equals(Token.TK\_MEMI)) {

            if (this.exp()) {

                if (this.exp\_3()) {

                    return true;

                } else {

                    this.printError("[exp\_3] Expressão inválida");

                    return false;

                }

            } else {

                this.printError("[exp\_3] Expressão inválida");

                return false;

            }

        } else {

            this.pointer -= 1;

            return true;

        }

    }

    public boolean exp\_1() {

        Lexema item = this.readLexema();

        if (item.getToken().equals(Token.TK\_APR)) {

            if (this.exp()) {

                item = this.readLexema();

                if (item.getToken().equals(Token.TK\_FPR)) {

                    if (this.exp\_2()) {

                        return true;

                    } else {

                        this.printError("[exp\_1-exp-exp\_2] Expressão invalida");

                        return false;

                    }

                } else {

                    this.printError("[exp\_1-exp-exp\_2] Esperava receber ]");

                    this.pointer -= 1;

                    return false;

                }

            } else {

                this.printError("[exp\_1-exp] Esperava receber ]");

                return false;

            }

        } else if (item.getToken().equals(Token.TK\_PONT)) {

            item = this.readLexema();

            if (item.getLexema().equals("sizeof")) {

                return true;

            } else {

                this.printError("[exp\_1] Esperava receber sizeof");

                this.pointer -= 1;

                return true;

            }

        } else if (item.getToken().equals(Token.TK\_AP)) {

            if (this.args()) {

                item = this.readLexema();

                if (item.getToken().equals(Token.TK\_FP)) {

                    return true;

                } else {

                    this.pointer -= 1;

                    this.printError("[exp\_1-args] Esperava receber ]");

                    return false;

                }

            } else {

                this.printError("[exp\_1-args] Expressão inválida");

                return false;

            }

        } else if (item.getToken().equals(Token.TK\_MA)) {

            return true;

        } else if (item.getToken().equals(Token.TK\_MEM)) {

            return true;

        }

        this.pointer -= 1;

        return true;

    }

    public boolean exp\_2() {

        Lexema item = this.readLexema();

        if (item.getToken().equals(Token.TK\_IG)) {

            if (this.exp()) {

                return true;

            } else {

                this.printError("[exp\_2] Expressão invalida!");

                return false;

            }

        } else {

            this.pointer -= 1;

            return true;

        }

    }

    public boolean args() {

        if (this.arg\_list()) {

            return true;

        } else {

            return true;

        }

    }

    public boolean arg\_list() {

        if (this.exp()) {

            if (this.arg\_list\_1()) {

                return true;

            } else {

                this.printError("[arg\_list-arg\_list\_1] Expressão invalida!");

                return false;

            }

        } else {

            return false;

        }

    }

    public boolean arg\_list\_1() {

        Lexema item = this.readLexema();

        if (item.getToken().equals(Token.TK\_VIR)) {

            if (this.exp()) {

                if (this.arg\_list\_1()) {

                    return true;

                } else {

                    this.printError("[arg\_list\_1] Argumento Inválida!");

                    return false;

                }

            } else {

                this.printError("[arg\_list\_1-exp] Expressão Inválida!");

                return false;

            }

        } else {

            this.pointer -= 1;

            return true;

        }

    }

    public boolean local\_decls() {

        Lexema item = readLexema();

        if (TYPE\_SPEC\_ARRAY.contains(item.getLexema())) {

            item = readLexema();

            if (item.getToken().equals(Token.TK\_ID)) {

                if (this.local\_decl\_1()) {

                    return true;

                } else {

                    this.pointer -= 1;

                    return false;

                }

            } else {

                this.printError("[local\_decls] Esperava receber um ID");

                this.pointer -= 1;

                return false;

            }

        } else {

            this.pointer -= 1;

            return true;

        }

    }

    public boolean local\_decl\_1() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_FDI)) {

            return this.local\_decls();

        } else if (item.getToken().equals(Token.TK\_APR)) {

            item = readLexema();

            if (item.getToken().equals(Token.TK\_FPR)) {

                item = readLexema();

                if (item.getToken().equals(Token.TK\_FDI)) {

                    return this.local\_decls();

                } else {

                    this.printError("Esperava receber ;");

                    this.pointer -= 1;

                    return false;

                }

            } else {

                this.printError("Esperava receber ]");

                this.pointer -= 1;

                return false;

            }

        } else if (item.getToken().equals(Token.TK\_IG)) {

            if (this.exp()) {

                item = readLexema();

                if (item.getToken().equals(Token.TK\_FDI)) {

                    return this.local\_decls();

                } else {

                    this.pointer -= 1;

                    this.printError("[local\_decl\_1-exp] Esperava receber ;");

                    return false;

                }

            } else {

                this.printError("[local\_decl\_1-exp] Expressao invalida");

                return false;

            }

        } else {

            this.pointer -= 1;

            this.printError("[local\_decl\_1] Esperava receber ; ou [ ");

            return false;

        }

    }

    public boolean param\_list\_2() {

        if (this.param()) {

            return this.param\_list\_1();

        } else {

            return true;

        }

    }

    public boolean param() {

        Lexema item = readLexema();

        if (TYPE\_SPEC\_ARRAY.contains(item.getLexema())) {

            if (param\_type\_spec\_id()) {

                return true;

            } else {

                this.pointer -= 1;

                return false;

            }

        } else {

            this.pointer -= 1;

            return false;

        }

    }

    public boolean param\_type\_spec\_id() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_ID)) {

            if (this.param\_type\_spec\_id\_param\_1()) {

                return true;

            } else {

                this.pointer -= 1;

                return false;

            }

        } else {

            this.pointer -= 1;

            return false;

        }

    }

    public boolean param\_type\_spec\_id\_param\_1() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_APR)) {

            return this.param\_type\_spec\_id\_param\_1\_close\_parent();

        } else {

            this.pointer -= 1;

            return true;

        }

    }

    public boolean param\_type\_spec\_id\_param\_1\_close\_parent() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_FPR)) {

            return true;

        } else {

            this.printError("[param\_type\_spec\_id\_param\_1\_close\_parent] Esperava receber ] ");

            this.pointer -= 1;

            return false;

        }

    }

    public boolean param\_list\_1() {

        Lexema item = readLexema();

        if (item.getToken().equals(Token.TK\_VIR)) {

            if (this.param()) {

                return this.param\_list\_1();

            } else {

                if (this.error.equals("")) {

                    this.printError("[param\_list\_1] Esperava receber um parametro ");

                }

                return false;

            }

        } else {

            this.pointer -= 1;

            return true;

        }

    }

}