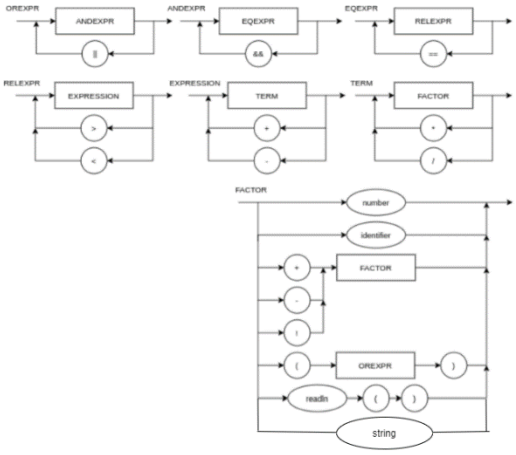
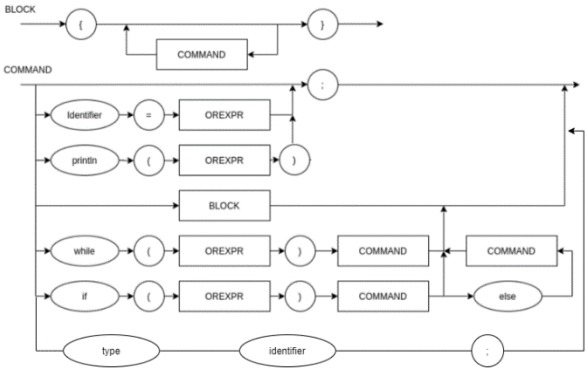
Roteiro 7 – Lógica da Computação

INSPER – 7° Semestre – Matheus Pellizzon

Problemas preliminares

1. Rascunhe as modificações no modelo **EBNF** e no **Diagrama Sintático** baseado nos novos elementos.





```
BLOCK = "{", { COMMAND, } "}" ;
COMMAND = ( λ | ASSIGNMENT | PRINT | READ), ";" | BLOCK | WHILESTMT | IFSMT | DECLARE
;
ASSIGNMENT = IDENTIFIER, "=", OREXPR ;
PRINT = "println", "(", OREXPR, ")" ;
READ = "readln", "(", ")" ;
DECLARE = TYPE, IDENTIFIER, ";" ;
TYPE = ( "int" | "strign" | "bool" ) ;
IFSTMT = "if", "(", OREXPR, ")", COMMAND ["else", COMMAND] ;
WHILESTMT = "while", "(", OREXPR, ")", COMMAND ;
OREXPR = ANDEXPR ["||", ANDEXPR] ;
ANDEXPR = EQEXPR ["&&", EQEXPR] ;
EQEXPR = RELEXPR ["==", EQEXPR] ;
RELEXPR = EXPRESSION [(">" | "<"), EXPRESSION] ;
EXPRESSION = TERM, { ("+" | "-"), TERM } ;
TERM = FACTOR, { ("*" | "/"), FACTOR } ;
FACTOR = (("+" | "-"| "!"), FACTOR) | NUMBER | "(", OREXPR, ")" | IDENTIFIER | READ |
STRING;
IDENTIFIER = LETTER, { LETTER | DIGIT | "_" } ;
NUMBER = DIGIT, { DIGIT } ;
LETTER = ( a | ... | z | A | ... | Z ) ;
DIGIT = ( 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 ) ;
STRING = QUOT_MARK, {(DIGIT | LETTER | SPECIAL_CHARACTERS)}, QUOT_MARK ;
QUOT_MARK = '"' ;
```

Liste e pense como serão os novos elementos da AST (value, children e Evaluate).

Todos os elementos da AST foram modificados para retornarem (valor, tipo) de alguma operação ou para funcionarem adequadamente com essas modificações (caso do if e do while, por exemplo).

Foram restringidos os tipos de operações dependendo do tipo de variável (não podemos fazer operações binárias ou unárias com strings, nem lógicas. A única permitida é ==)

Além dessas modificações, foi adicionado um elemento BoolVal, responsável por tratar os casos em que temos true e false e StringVal para tratar strings. Semelhante ao IntVal, nenhum dos dois novos elementos possui filhos, apenas value. O método Evaluate de cada um retorna (0 ou 1, “TYPE_BOOL”) e (valor string, “TYPE_STRING”), respectivamente.

O print foi modificado para printar true ou false no terminal (ao invés de 0 e 1) nos casos da variável ser booleana. Também é responsável por remover as aspas de strings e printar somente seu conteúdo, uma vez que tokenizo a string inteira, incluindo aspas.

A última modificação feita foi na symbol table que, além de guardar o valor de uma variável, guarda seu valor e tipo em uma tupla. Foram implementados dois métodos novos, declare, para declarar uma variável com valor None e tipo designado no código (se a variável for redeclarada, lança um erro). O método contains serve para validação da variável declarada no node Assign.

Base de Testes:
Proponha um programa de testes, com os seguintes elementos:

• usar de variáveis de todos os tipos.

• verificar se programas antigos ainda funcionam.

• operar tipos incorretos.

• testar um if/while com uma string de entrada.Teste utilizado:

```
{
    int x;
    bool y;
    string z;
    x = 0;
    y = 3 > 2;
    z = "b";
    println(x);
    println(x + y);
    println(z);
    println(z == "a");

    while (x < 10 && y)
    {
        x = x + 1;
        println(x + y);
        if (x == 4 && y)
        {
            y = !y;
            println("y inverteu valor");
        }
    }
    println("saiu do while");

    /*ERRO:*/
    println(z > x);
    println(z == y);
    while (z)
    {
        println(z);
    }
}
```

Questionário 1. Como você modificaria o seu compilador para gerar código assembly do código fonte? Não precisa fazer, apenas descreva como faria

Poderia adicionar um Assembler. Ao chamar os métodos evaluate de cada nó, traduzimos esse trecho de código para assembly, que pode ser escrito em algum arquivo ou variável. No final, teríamos o código completo em assembly.