Invente uma linguagem

Programol - Language Oriented Programming

Programol foi uma linguagem de programação desenvolvida na véspera de páscoa em 2017, pelos engenheiros arquitetos Márcia Regina e Wendel Ribeiro, com o objetivo de terminar as pressas o trabalho especificado em uma das disciplinas do curso de ciência da computação do Instituto federal de Brasília.

A linguagem de programação conta com técnicas avançadas das convenções universitárias de programação Orientada a gambiarra e outras técnicas utilizadas em projetos no qual os universitários não se preocupam em corrigir.

Especificações mínimas

- 1. Especifique os seguintes tokens:
 - Identificadores
 - Números reais (com expoente) *utilizar vírgula como separador da parte fracionária
 - Números naturais
 - Operadores aritméticos básicos
 - Operadores relacionais
 - Operadores lógicos
 - Delimitadores de bloco de comandos ;
 - Terminador de comando/linha
 - Comentários
 - Strings

Comandos:

- Um de repetição
- Um de seleção

Gramática

```
= \langle r | \langle n | \langle r \rangle n \rangle
LINHA
OPERADOR ARITMETICO = [ + | * | - | ]
             = [_|[a-zA-Z]]*
NUMEROS_NATURAIS = [0-9] +
                       = '.*'
TEXTO
NUMEROS_REIAS = [0-9]+, [0-9]+
EXPOENTE = \\ \\ \\ \\ \\ 
                        = ":|"
OU
Ε
                       = "&&"
IGUAL
                       = "=="
NAO IGUAL
                       = "!="
```

```
MENOR
                      = "<"
MAIOR
                      = ">"
MAIOR IGUAL
                      = ">="
MENOR IGUAL
                     = "<="
NAO
                      = "!"
INICIO BLOCO
                     = " { "
FIM BLOCO
                     = "}"
FIM LINHA
                     = ";"
ATRIBUICAO
                     = "="
BRANCO
                       = [ \t \f]
OPERADOR RELACIONAL =
 \{ \texttt{IGUAL} \} \mid \{ \texttt{NAO}\_\texttt{IGUAL} \} \mid \{ \texttt{MENOR} \} \mid \{ \texttt{MENOR}\_\texttt{IGUAL} \} \mid \{ \texttt{MAIOR}\_\texttt{IGUAL} \} 
OPERADOR LOGICO = \{OU\} | \{E\} | \{NAO\}
응응
"fim"
                            { lexeme = yytext(); return
FIM COMANDO; }
"then"
                     { lexeme = yytext(); return THEN; }
"if"
                     { lexeme = yytext(); return IF; }
"for"
                     { lexeme = yytext(); return FOR; }
"while"
                     { lexeme = yytext(); return WHILE; }
{ID}
                         { lexeme = yytext(); return ID; }
{OPERADOR_ARITMETICO} { lexeme = yytext(); return
OPERADOR ARITMETICO; }
{NUMEROS NATURAIS} { lexeme = yytext(); return NUMEROS NATURAIS;
{TEXTO}
                         { lexeme = yytext(); return TEXTO; }
{ NUMEROS_REIAS} { lexeme = yytext(); return NUMEROS_REIAS; }
{OPERADOR LOGICO} { lexeme = yytext(); return OPERADOR LOGICO;
{OPERADOR RELACIONAL} { lexeme = yytext(); return
OPERADOR RELACIONAL; }
{EXPOENTE}
                        { lexeme = yytext(); return EXPOENTE; }
{INICIO_BLOCO} { lexeme = yytext(); return INICIO_BLOCO;
{FIM_BLOCO}
                        { lexeme = yytext(); return FIM_BLOCO; }
{FIM LINHA}
                            { lexeme = yytext(); return FIM_LINHA;}
{ATRIBUICAO}
                            { lexeme = yytext(); return ATRIBUICAO;
{BRANCO}
                            { lexeme = yytext(); return BRANCO;}
{LINHA}
                                  { lexeme = yytext(); return
LINHA; }
(["$$"].*)
                            { lexeme = yytext(); return COMENTARIO;}
                      { lexeme = yytext(); return ERROR; }
```