1 Primeira Fase

Nesta primeira fase vocês realizarão a análise sintática para a linguagem descrita pela gramática da Seção 3.

Esta gramática está parcialmente representada e será completada conforme as próximas etapas do trabalho, podendo sofrer alterações em certas regras de produção.

2 Entrega

Data: **04/04/16**

3 Gramática

A seguir estão listados alguns elementos da notação de descrição da gramática:

- {} A regra de produção pode ser repetida 0 ou mais vezes.
- [] A regra de produção é opcional.
- | Separa regras de produção alternativas.

As regras de produção aparecem com as letras inicias maiúsculas.

Note que os terminais da linguagens estão descritos entre aspas simples, como por exemplo ';' na regra de produção VariableDecl e os terminais que são palavras-chave da linguagem estão, também, em negrito.

Não confunda os terminais '{' '}' com o símbolo de notação { }, pois este determina o número de repetições de certa regra de produção e aquele marca o início e final de bloco. O mesmo vale para '[' ']'.

A gramática da linguagem está detalhada abaixo:

Program	::= Decl
Decl	::= '**v**' '**m**' '(' ')' StmtBlock
StmtBlock	::= '{' { VariableDecl } { Stmt } '}'
VariableDecl	::= Variable ';'
Variable	::= Type Ident
Type	::= StdType | ArrayType
StdType	::= '**i**' | '**d**' | '**c**'
ArrayType	::= StdType '[' ']'
Stmt	::= Expr ';' | ifStmt | WhileStmt | BreakStmt | PrintStmt
IfStmt	::= '**f**' '(' Expr ')' '{' { Stmt } '}' ['**e**' '{' { Stmt } '}']
WhileStmt	::= '**w**' '(' Expr ')' '{' { Stmt } '}'
BreakStmt	::= '**b**' ';'
PrintStmt	::= '**p**' '(' Expr { ',' Expr }')'

```
Expr      ::= SimExpr [ RelOp Expr]
SimExpr   ::= [Unary] Term { AddOp Term }
Term      ::= Factor { MulOp Factor }
Factor    ::= LValue '=' Expr
            | LValue
            | '(' Expr ')'
            | 'r' '(' ')' | 's' '(' ')' | 't' '(' ')'
LValue    ::= Ident | Ident '[' Expr ']'
Ident     ::= Letter { Letter | Digit }
RelOp     ::= '=' | '#' | '<' | '>'
AddOp     ::= '+' | '-'
MulOp     ::= '*' | '/' | '%'
Unary     ::= '+' | '-' | '!'
Digit     ::= '0' | '1' | ... | '9'
Letter    ::= 'A' | 'B' | ... | 'Z' | 'a' | 'b' | ... | 'z'
```