

Alunos: Alexandre Diniz de Souza e Leandro José Gonçalves Pereira

RELATÓRIO - T1 COMPILADORES

Neste trabalho desenvolvemos um analisador **léxico** e **sintático** com a funcionalidade de recuperação de erros (especificada posteriormente) para a linguagem mini java a qual foi fornecida a gramática na descrição do trabalho. A partir da gramática fornecida foi necessário realizar modificações a fim de torná-la em uma gramática LL(1) para a criação de um analisador sintático descendente recursivo. A partir da gramática inicial realizamos modificações para retirar recursão à esquerda e não determinismos encontrados. Com isso chegamos a gramática abaixo descrita.

GRAMÁTICA MODIFICADA

Program → *MainClass* *ClassDeclaration*' EOF

MainClass → **class** **ID** { **public static void main** (**String**[] **ID**) { *Statement* } }

ClassDeclaration → **class** **ID** *ClassDeclaration*'' { *VarDeclaration*' *MethodDeclaration*' }

ClassDeclaration' → *ClassDeclaration* *ClassDeclaration*'

| ε

ClassDeclaration'' → **extends** **ID**

| ε

VarDeclaration → **int** *Type*' **ID** ;

| **boolean** **ID** ;

| **ID** **ID** ;

VarDeclaration' → **int** *Type*' **ID** ; *VarDeclaration*'

| **boolean** **ID** ; *VarDeclaration*'

| **ID** **ID** ; *VarDeclaration*'

| ε

MethodDeclaration → **public** *Type* **ID** (*Type*'') { *VarDeclarationStatement* **return**

Expression ; }

MethodDeclaration' → *MethodDeclaration* *MethodDeclaration*'

| ε

VarDeclarationStatement → **int** *Type*' **ID** ; *VarDeclarationStatement*

- | **boolean ID ; VarDeclarationStatement**
- | **ID VarDeclarationStatement''**
- | **{ Statement'' } Statement''**
- | **if (Expression) Statement else Statement Statement''**
- | **while (Expression) Statement Statement''**
- | **System.out.println (Expression) ; Statement''**
- | ϵ

VarDeclarationStatement' \rightarrow **ID ; VarDeclarationStatement**
 | **Statement' Statement''**

Type \rightarrow **int Type'**
 | **boolean**
 | **ID**

Type' \rightarrow []
 | ϵ

Type'' \rightarrow , *Type* **ID** *Type''*
 | ϵ

Type''' \rightarrow *Type* **ID** *Type''*
 | ϵ

Statement \rightarrow { *Statement''* }
 | **if (Expression) Statement else Statement**
 | **while (Expression) Statement**
 | **System.out.println (Expression) ;**
 | **ID Statement'**

Statement' \rightarrow = *Expression* ;
 | [*Expression*] = *Expression* ;

Statement'' \rightarrow { *Statement''* } *Statement''*
 | **if (Expression) Statement else Statement Statement''**
 | **while (Expression) Statement Statement''**
 | **System.out.println (Expression) ; Statement''**
 | **ID Statement' Statement''**
 | ϵ

$Expression \rightarrow \text{INTEGER_LITERAL } Expression'$

$| \text{true } Expression'$

$| \text{false } Expression'$

$| \text{ID } Expression'$

$| \text{this } Expression'$

$| \text{new } Expression'' Expression'$

$| ! Expression Expression'$

$| (Expression) Expression'$

$Expression' \rightarrow Op Expression Expression'$

$| [Expression] Expression'$

$| . Expression'''$

$| \epsilon$

$Expression'' \rightarrow \text{int } [Expression]$

$| \text{ID } ()$

$Expression''' \rightarrow \text{length } Expression'$

$| \text{ID } (Expression'''') Expression'$

$Expression'''' \rightarrow , Expression Expression''''$

$| \epsilon$

$Expression''''' \rightarrow Expression Expression''''$

$| \epsilon$

$Op \rightarrow \&\&$

$| <$

$| >$

$| ==$

$| !=$

$| +$

$| -$

$| *$

$| /$

RECUPERAÇÃO DE ERROS

A estratégia de recuperação de erro que utilizamos foi a “Panic-mode” (Modalidade do Desespero), que ao encontrar um erro, descarta os símbolos da entrada até encontrar um token de sincronização. Consideramos o conjunto Follow da produção em que o erro ocorreu como conjunto de sincronização. Outra característica da nossa abordagem surgiu da necessidade de minimizar a quantidade de tokens descartados a fim de maximizar a porção de código que será verificada. Para isso, ao se deparar com um token que não casa com o esperado na chamada da função *match()* mantemos o lookahead no mesmo token e implementamos uma lógica para que a próxima chamada ao método *match()* verifique o token atual (que gerou o erro inicialmente) e, caso não case, avance para o próximo token e o verifique.

OBS: Utilizamos o Sistema Operacional Linux para desenvolvimento e testes do compilador.