Linguagem Futebolística

APS DE LÓGICA DA COMPUTAÇÃO

Guilherme Fontana
Prof. Raul Ideka

Objetivos do Projeto

- Estruturar a linguagem segundo o padrão EBNF.
- Utilizar as ferramentas Flex e Bison para realizar as etapas de Análise Léxica e Sintática. A saída deve ser um arquivo C ou CPP compilado pelo Flex/Bison.
- Utilizar alguma VM (LLVM, JVM, .net, etc) para interpretar um programa da sua linguagem.
- Criar um exemplo de testes que demonstre as características da sua Linguagem.

Detalhes da Linguagem

- A linguagem é baseada em golang, alterando certos comandos para termos futebolísticos.
- EBNF:

```
<sentence> ::= ( "λ" | <assign> | <conditional> | <loop> )
<assign> ::= "var", <identifier>, "=", <boolean expression>
<conditional> ::= "If", <boolean expression>, <block>, [ "Else:", <block> ]
<loop> ::= "For", <boolean expression>, <block>
<block> ::= "{", <sentence>, "}"
<boolean expression> ::= <boolean clause>, { "or", <boolean clause> }
<boolean clause> ::= <relational expression>, { "and", <relational expression> }
<relational expression> ::= <expression>, { ( "draw" | "win" | "loss" ), <expression> }
<expression> ::= <term>, { ( "guardiola" | "mourinho" | "ferguson" ), <term> }
<term> ::= <factor>, { ( "bellingham" | "camavinga" ), <factor> }
<factor> ::= <number> | <string> | <identifier> | ( ( "OnSide" | "Offside" | "not" ), <factor> )
<identifier> ::= <letter>, { <letter> | <digit> }*
<number> ::= <digit>+
<string> ::= '"' { "λ" | <letter> | <digit> }* '"'
<letter> ::= "a" | "b" | ... | "z" | "A" | "B" | ... | "Z"
<digit> ::= "0" | "1" | "2" | ... | "9"
```

Detalhes da Linguagem

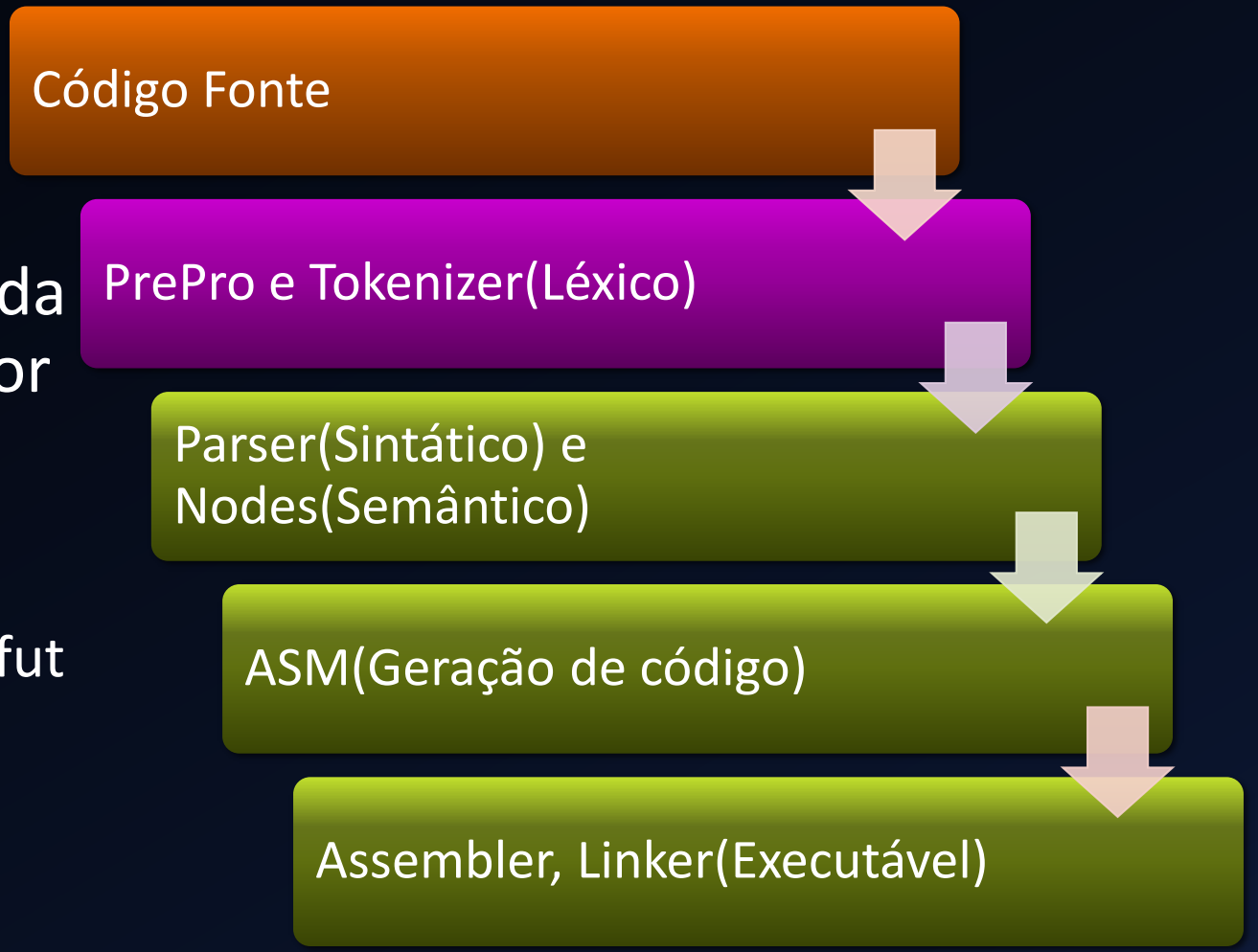
- Legenda dos termos futebolísticos e justificativas:
  - Bellingham: Multiplicação * (Porque ele se multiplica em campo)
  - Camavinga: Divisão / (Se divide para qualquer lugar do campo)
  - Guardiola: Soma + (A soma do tiki-taka)
  - Mourinho: Subtração - (O oposto do Guardiola)
  - Draw: Igualdade == (O empate é a igualdade no futebol)
  - Win: Maior que > (A vitória só acontece quando o placar de um time é maior que o outro)
  - Loss: Menor que < (A derrota acontece quando o placar de um time é menor que o outro)

Análise Léxica e Sintática

- Foi utilizado Flex/Bison
- Na interpretação da Linguagem, foi utilizado o Compilador feito para a matéria
- Exemplos de teste disponíveis no github

Esquema de Funcionamento do Compilador

- A lógica segue a do compilador da matéria (golang)
- A mudança dos tokens foi tratada no Tokenizer, assim o compilador considera a linguagem .fut
- Comandos para teste:
  - Python3 main.py exemplos/testX.fut
  - nasm -f elf -o testX.o textX.asm
  - gcc -m32 -no-pie -o testX testX.o



Exemplo: Test 1

```
var x int
var y int
x = 3 Guardiola 1
y = x
if x Win 1 {
    x = 5 Mourinho 1
}
if (x Draw 3) {
} else {
    x = 3
}
for x = 3; x Loss 5; x = x Guardiola 1 {
    y = x Mourinho 1
}
Println(x)
```

- Declaração de variáveis e assign das mesmas
- A lógica de if else é a mesma de golang. Pode-se ver os operadores Guardiola(+) e Mourinho(-)
- Lógica do for: Na condição, X = 3, enquanto x < 5, x incrementando 1. Dentro do block, y = x – 1
- Print de x funcionando(5).

Exemplo: Test 2

```
var x int
var y int
x = 3 Bellingham 2
y = x
Println(y)
```

- Declaração de variáveis e assign das mesmas
- Assign usando o Bellingham(Multiplicação)
- Assign de y sendo o valor de x
- Print Correto de y, com valor 6

Exemplo: Test 3

```
var x int
var y int
x = 6 Draw 2
y = x
Println(y)
```

- Declaração de variáveis e assign das mesmas
- Assign usando a comparação de igualdade (Draw)
- Assign de y sendo o valor de x
- Print Correto de y, com valor 0(False)

Obrigado!