PCS2056 – Linguagens e Compiladores

Rogério Yuuki Motisuki - 8587052
Guilherme Mamprin - 8587584

Formalização de uma sintaxe - BIRL++

1. Descrição informal da linguagem

A nossa linguagem é inspirada na *Bambam's "It's show time" Recursive Language* (BIRL), que por sua vez é baseada na linguagem C. Dessa forma, denominamos nossa linguagem de: BIRL++

As terminações de linha são feitas através do caractere ponto-e-vírgula: **;**
As terminações de bloco são feitas através da sintaxe: BIRL

- Comentários

Comentários de uma linha são feitos através dos símbolos: // comentario
Um comentário multi-linha pode ser declarado com: /* comentario multi-linha */

- Declaração de variáveis

A linguagem possui os tipos tradicionais do C, com outro nome:

BIRL++	C
FRANGO	char
MONSTRINHO	short
MONSTRO	int
MONSTRAO	long
TRAPEZIO	float
TRAPEZIO DESCENDENTE	double

Os 4 primeiros tipos podem ser transformados em *unsigned* através do modificador “BICEPS”:

Exemplo: `BICEPS MONSTRO x = 13;`

- Declaração de tipo (struct)

A declaração de struct se dá através da sintaxe com a palavra BODYBUILDER:

```
BODYBUILDER nome_da_struct
      TRAPEZIO x;
      TRAPEZIO y;
BIRL
```

- Condicional

A estrutura usual de *if/else if/else* no BIRL++ é a seguinte, tendo apenas o primeiro bloco como obrigatório:

```
ELE QUE A GENTE QUER? (X > 2)
      //X > 2
QUE NAO VAI DAR O QUE? (X < 2)
      //X < 2
NAO VAI DAR NAO
      //X = 2
BIRL
```

Outra estrutura possível é o *switch/case/default*, onde a palavra *break* é substituida por SAI FDP:

```
MONSTRO x = 3;
DERRUBAR ARVORES (x)
      ARVORE 1:
            // x == 1
            SAI FDP;
      ARVORE 2:
            // x == 2
            SAI FDP;
      ARVORE DO PARQUE DO IBIRAPUERA:
            // default
BIRL
```

As constantes booleanas *true* e *false* são, respectivamente: FIBRA e AGUA
A negação utiliza a palavra: NEGATIVA

- Iteração

Há duas formas de iteração, via *for* ou via *while*:

for:
```
MONSTRO M;
MAIS QUERO MAIS (M = 0; M < 5; M++)
      // bloco
BIRL
```

while:
```
MONSTRO X = 5;
NEGATIVA BAMBAM (X > 2)
      X--;
BIRL
```

As palavras reservadas usadas na iteração *continue* e *break*, em BIRL++ são:

BIRL++	C
VAMO MONSTRO	continue
SAI FDP	break

- Declaração de função

A declaração de função segue a seguinte sintaxe, onde *return* é substituido por `BORA CUMPADE`:

```
OH O HOME AI PO (MONSTRO soma(MONSTRO x, MONSTRO y))
      BORA CUMPADE x + y;
BIRL
```

- Execução de função

A execução de função é realizada através da expressão `AJUDA O MALUCO TA DOENTE`:

```
MONSTRO A = 5;
MONSTRO B = 8;
MONSTRO C = AJUDA O MALUCO TA DOENTE soma(A, B);
```

- Função principal (main)

  A *main* é uma função especial, declarada com a seguinte sintaxe:

```
HORA DO SHOW
      //código aqui
BIRL
```

2. **Exemplo de programa escrito na linguagem definida**

```
OH O HOME AI PO (MONSTRO soma(MONSTRO x, MONSTRO y))
      BORA CUMPADE x + y;
BIRL

HORA DO SHOW
      MONSTRO i, x = 0, y = 0;

      MAIS QUERO MAIS (i = 0; i < 100; i++)
            x = AJUDA O MALUCO TA DOENTE soma(x,i);
      BIRL

      ELE QUE A GENTE QUER? (x % 2 == 0)
            y = x / 2;
      NAO VAI DAR NAO
            y = x;
      BIRL

      BORA CUMPADE 1;
BIRL
```

3. Descrição da linguagem em BNF

```
<PROGRAMA> ::= <DEFINIÇÃO> <MAIN> <DEFINIÇÃO>
             | <DEFINIÇÃO> <MAIN>
             | <MAIN> <DEFINIÇÃO>

<DEFINIÇÃO> ::= <STRUCT> <DEFINIÇÃO>
              | <FUNÇÃO> <DEFINIÇÃO>
              | <STRUCT>
              | <FUNÇÃO>

<STRUCT> ::= BODYBUILDER <IDENTIFICADOR> <LISTA-DECLARAÇÃO-TIPOS> BIRL

<LISTA-DECLARAÇÃO-TIPOS> ::= <DECLARAÇÃO-TIPO> <LISTA-DECLARAÇÃO-TIPO>
                           | <DECLARAÇÃO-TIPO>

<DECLARAÇÃO-TIPO> ::= <TIPO> <IDENTIFICADOR> ;

<TIPO> ::= FRANGO | MONSTRINHO | MONSTRO | MONSTRAO | TRAPEZIO | TRAPEZIO DESCENDENTE
         | BICEPS FRANGO | BICEPS MONSTRINHO | BICEPS MONSTRO | BICEPS MONSTRAO

<IDENTIFICADOR> ::= <LETRA> <LETRA-DIGITO>
<LETRADIGITO> ::= <LETRA> <LETRA-DIGITO>
                | <DIGITO> <LETRADIGITO>
                | <LETRA>
                | <DIGITO>
<LETRA> ::= a | b | c | d | e | f | ... | A | B | C | ... | Z
<DIGITO> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<FUNÇÃO> ::= OH O HOME AI PO ( <TIPO> <IDENTIFICADOR> ( <LISTA-PARÂMETROS> ) )
<BLOCO-CÓDIGO> BIRL

<LISTA-PARÂMETROS> ::= <PARÂMETRO> , <LISTA-PARÂMETROS> | <PARÂMETRO>
<PARÂMETRO> ::= <TIPO> <IDENTIFICADOR>

<MAIN> ::= HORA DO SHOW <BLOCO-CÓDIGO> BIRL

<BLOCO-CÓDIGO> ::= <COMANDO> <BLOCO-CÓDIGO> | <COMANDO>

<COMANDO> ::= <COMANDO-ATRIBUIÇÃO> | <CONDICIONAL> | <ITERATIVO> | <PULO>

<COMANDO-ATRIBUIÇÃO> ::= <TIPO> <LISTA-ATRIBUIÇÃO>;
                       | <LISTA-ATRIBUIÇÃO>;
<LISTA-ATRIBUIÇÃO> ::= <ATRIBUIÇÃO> , <LISTA-ATRIBUIÇÃO> ;
                     | <ATRIBUIÇÃO> ;
<ATRIBUIÇÃO> ::= <IDENTIFICADOR> = <EXPRESSÃO> | <IDENTIFICADOR>
```

```
<EXPRESSÃO> ::= <EXPRESSÃO-BOOL> | <EXPRESSÃO-ARITM>
<EXPRESSÃO-BOOL> ::= <EXPRESSÃO-BOOL> <OPERADOR-BOOL> <EXPRESSÃO-BOOL>
                   | <EXPRESSÃO-ARITM> <OPERADOR-COMP> <EXPRESSÃO-ARITM>
                   | <IDENTIFICADOR> | <CHAMADA-FUNÇÃO>
                   | FIBRA | AGUA | NEGATIVA <EXPRESSÃO-BOOL>
                   | ( <EXPRESSÃO-BOOL> )
<EXPRESSÃO-ARITM> ::= <EXPRESSÃO-ARITM> <OPERADOR-ARITM> <EXPRESSÃO-ARITM>
                    | <IDENTIFICADOR> | <CHAMADA-FUNÇÃO>
                    | <NÚMERO>
                    | ( <EXPRESSÃO-ARITM> )
<OPERADOR-BOOL> ::= && | “||”
<OPERADOR-ARITM> ::= + | - | * | / | %
<OPERADOR-COMP> ::= == | != | > | >= | < | <=
<NÚMERO> ::= <DIGITO><NÚMERO>

<CHAMADA-FUNÇÃO> ::= AJUDA O MALUCO TA DOENTE <IDENTIFICADOR> (<LISTA-ARGUMENTOS>)
<LISTA-ARGUMENTOS> ::= <EXPRESSÃO> | <EXPRESSÃO> , <LISTA-ARGUMENTOS>

<CONDICIONAL> ::= <IF> | <SWITCH>
<IF> ::= ELE QUE A GENTE QUER? ( <EXPRESSÃO> ) <BLOCO-CÓDIGO> BIRL
       | ELE QUE A GENTE QUER? ( <EXPRESSÃO> ) <BLOCO-CÓDIGO> <ELSE-IF> BIRL
<ELSE-IF> ::= QUE NAO VAI DAR O QUE? ( <EXPRESSÃO> ) <BLOCO-CÓDIGO>
            | QUE NAO VAI DAR O QUE? ( <EXPRESSÃO> ) <BLOCO-CÓDIGO> <ELSE-IF>
            | NAO VAI DAR NAO <BLOCO-CÓDIGO>

<SWITCH> ::= DERRUBAR ARVORES ( <EXPRESSÃO> ) <CASE> BIRL
<CASE> ::= ARVORE <EXPRESSÃO-CONSTANTE>: <BLOCO-CÓDIGO>
         | ARVORE <EXPRESSÃO-CONSTANTE>: <BLOCO-CÓDIGO> <CASE>
         | ARVORE DO PARQUE DO IBIRAPUERA: <BLOCO-CÓDIGO>

<EXPRESSÃO-CONSTANTE> ::= <EXPRESSÃO-BOOL-CONST> | <EXPRESSÃO-ARITM-CONST>
<EXPRESSÃO-BOOL-CONST> ::= <EXPRESSÃO-BOOL-CONST> <OPERADOR-BOOL>
<EXPRESSÃO-BOOL-CONST>
                   | <EXPRESSÃO-ARITM-CONST> <OPERADOR-COMP> <EXPRESSÃO-ARITM-CONST>
                   | FIBRA | AGUA
                   | ( <EXPRESSÃO-BOOL-CONST> )
<EXPRESSÃO-ARITM-CONST> ::= <EXPRESSÃO-ARITM-CONST> <OPERADOR-ARITM>
<EXPRESSÃO-ARITM-CONST>
                    | <NÚMERO>
                    | ( <EXPRESSÃO-ARITM-CONST> )

<ITERATIVO> ::= <FOR> | <WHILE>
<WHILE> ::= NEGATIVA BAMBAM ( <EXPRESSÃO> ) <BLOCO-CÓDIGO> BIRL
<FOR> ::= MAIS QUERO MAIS <FOR-EXPR> <BLOCO-CÓDIGO> BIRL
<FOR-EXPR> ::= ( <FOR-EXPR2> ; <EXPRESSÃO> ) | ( <FOR-EXPR2> ; )
<FOR-EXPR2> ::= <EXPRESSÃO> ; <EXPRESSÃO> | ; <EXPRESSÃO> | <EXPRESSÃO> ; | ;

<PULO> ::= BORA CUMPADE <EXPRESSÃO>; | SAI FDP; | VAMO MONSTRO;
```

4. Descrição da linguagem em Wirth

Disponível no anexo “WIRTH.txt”.