Declarações - Listas de inteiros

Professor:
Christiano Braga
Aluno:
Allan Patrick De Freitas Santana

Bnf -

⟨start⟩ ::= [ds:decSeq] [cs:cmd_seq]

⟨exp⟩ ::= ⟨arr⟩ | ⟨bin_exp⟩ | ⟨un_exp⟩ | ⟨paren_exp⟩ | ⟨atom⟩

⟨arr⟩ ::= '[' ','% ⟨exp⟩ * ']' | ⟨identifier⟩ '[' ⟨exp⟩ ']' | 'concat' '(' ⟨exp⟩ ',' ⟨exp⟩ ')' | 'append' '(' ⟨exp⟩ ',' ⟨ exp⟩ ')'

⟨paren_exp⟩ ::= '(' ⟨exp⟩ ')'

⟨bin_exp⟩ ::= ⟨exp⟩ ⟨binop⟩ ⟨exp⟩

⟨un_exp⟩ ::= 'len' '(' ⟨exp⟩ ')' | 'not' ⟨exp⟩

⟨atom⟩ ::= ⟨num_atom⟩ | ⟨truth_atom⟩ | ⟨id_atom⟩

⟨num_atom⟩ ::= ⟨number⟩

⟨truth_atom⟩ ::= ⟨truth⟩

⟨id_atom⟩ ::= ⟨identifier⟩

⟨binop⟩ ::= 'and' | 'or' | '==' | '<=' | '>=' | '<' | '>' | '+' | '-' | '*' | '/'

⟨truth⟩ ::= 'True' | 'False'

⟨identifier⟩ ::= /(?!\d)\w+/

⟨number⟩ ::= /\d+/

⟨blk⟩ ::= '{' ⟨decSeq⟩ ⟨cmd_seq⟩ '}'

⟨cmd_seq⟩ ::= ','% ⟨atom_cmd⟩ ⁺

⟨atom_cmd⟩ ::= ⟨cond⟩ | ⟨loop⟩ | ⟨assign⟩ | ⟨print⟩ | ⟨call⟩ | ⟨skip⟩

⟨skip⟩ ::= 'skip'

⟨assign⟩ ::= ⟨identifier⟩ ':=' ⟨exp⟩ | ⟨identifier⟩ '[' ⟨exp⟩ ']' ':=' ⟨exp⟩

⟨print⟩ ::= 'print' ⟨exp⟩

⟨cond⟩ ::= 'if' ⟨paren_exp⟩ ⟨blk⟩ ['else' ⟨blk⟩]

⟨loop⟩ ::= 'while' ⟨paren_exp⟩ ⟨blk⟩

⟨dec⟩ ::= ⟨fn⟩ | ⟨rec⟩ | ⟨var⟩ | ⟨const⟩

⟨decSeq⟩ ::= ⟨dec⟩ *

⟨var⟩ ::= 'var' ','%{ ⟨identifier⟩ '=' ⟨exp⟩ }+ ';'

⟨const⟩ ::= 'const' ⟨identifier⟩ '=' ⟨exp⟩ ';'

⟨fn⟩ ::= 'def' ⟨identifier⟩ '(' ⟨formal⟩ ')' ⟨blk⟩

⟨rec⟩ ::= 'rec' ⟨identifier⟩ '(' ⟨formal⟩ ')' ⟨blk⟩

⟨formal⟩ ::= ','% ⟨identifier⟩ *

⟨call⟩ ::= ⟨identifier⟩ '(' ⟨actual⟩ ')'

⟨actual⟩ ::= ','% ⟨exp⟩ *

Especificações ebnf

1. Foram feitas as seguintes alterações

- Foi adicionado as keywords concat e append.
- As seguintes produções foram alteradas exp, un_exp e assign.
- AS produções alteradas ficaram da seguinte forma:
    - exp = arr | bin_exp | un_exp | paren_exp | @:atom ;
    - un_exp = op:"len" "(" e:exp ")" | op:"not" e:exp ;
    - assign = idn:identifier op:":=" e:exp | idn:identifier["idx:exp"]" op:":=" e:exp ;
- Foi adicionado uma nova produção arr.
- A nova produção tem o seguinte formato:
    - arr = op:"[" ','%{e:exp}* "]" | idn:identifier["e:exp"]" | op:"concat" "(" e1:exp "," e2:exp ")" | op:"append" "(" e1:exp "," e2:exp ")";

BNF -> Impiler:

Conforme o impiler percorre a bnf, as expressões correspondentes terão suas classes no impiler chamadas. As funções a serem chamadas no impiler são mapeadas pelo nome, utilizando o termo a esquerda da expressão no BNF

arr - Quando ocorre a aparição de um array na ebnf a função arr é chamada e dentro dela podemos ter 4 caminhos distintos a partir da operação, os caminhos são:

- Index - Chama a função ArrIndex de pi para retornar o valor do array na posição do index;
- Append - Chama a função ArrAppend de pi para tratar a união de um array com um inteiro ou array;
- Concat - Chama a função ArrConcat de pi para tratar a concatenação de um array e um inteiro;
- Int- Chama a função ArrInt de pi para tratar os valores do array;

Especificações impiler

1. Foram feitas as seguintes alterações

- Alteração da definição de un_exp para busca de tamanho da lista;
- Criação da função arr para tratar todas as operações sobre listas;
- Alteração da Função assign, para atribuir o valor da lista a uma variável;

Impiler-> pi:

- **As funções a seguir são chamadas pelo impiler:**
- A classe ArrInt trata os valores do array e é chamada pela função arr do impiler;
- A classe ArrSize retorna o tamanho do array e é chamada pela função un_exp do impiler;
- A classe ArrIndex retorna o valor do array na posição do index e é chamada pela função arr do impiler;
- A classe ArrConcat trata a concatenação de um array e um inteiro e é chamada pela função arr do impiler;
- A classe ArrAppend trata a união de um array com um inteiro ou array e é chamada pela função arr do impiler;
- A classe ArrAssign trata a atribuição de um array para uma variável e é chamada pela função assign do impiler;

Especificações pi

1. Foram feitas as seguintes alterações

- Criação da classe ArrInt para tratar os valores do array;
- Criação da classe ArrSize para retornar o tamanho da array;
- Criação da classe ArrIndex para retornar o valor do array na posição do index;
- Criação da classe ArrConcat para tratar a concatenação de um array e um inteiro;
- Criação da classe ArrAppend para tratar a união de um array com um inteiro ou array;
- Criação das funções de avaliação para empilhar e desempilhar as novas instruções, as funções estão listadas a seguir:
    - __evalArrInt
    - __evalArrAppend
    - __evalArrAppendKW
    - __evalArrConcat
    - __evalArrConcatKW
    - __evalArrIndex
    - __evalArrIndexKW
    - __evalArrSize
    - __evalArrSizeKW
    - __evalArrAssign
    - __evalArrAssignKW
- Criação da classe ArrAssign para tratar a atribuição de um array;