Especificação - Lista de Inteiros Pi Imp

Felipe de Brito Vieira

1. Especificação das novas equações adicionadas ao Pi framework que dão semântica às novas construções em Pi IR que dão suporte à criação e manipulação de listas de inteiros. (pi.py)

As classes utilizadas foram as seguintes:
Array(Statement), **Array_val**(Exp), **Append**(Exp), **Array_concat**(Exp), **Array_len**(Exp), **Array_atrib**(Cmd).
As funções eval foram:

__evalArray -> Responsável por adicionar a lista no pilha de valores.

__evalArrayConcat -> Responsável por adicionar o comando de Concatenação a pilha de controle.

__evalArrayConcatKW -> Faz a concatenação de duas listas e adiciona a pilha de valores.

__evalArrayLen -> Responsável por recupar a lista a partir do id e adicionar seu tamanho a pilha de valores.

__evalArrayAppend -> Responsável por adicionar o comando de Append a pilha de controle com seus valores.

__evalArrayAppendKW -> Responsável por criar uma cópia da lista com o append do novo valor numérico.

__evalArrayVal -> Responsável por adicionar o comando de recuperar valor de um array junto do id e posição do valor que deseja na pilha de controle.

__evalArrayValKW -> Retorna o valor da lista na posição desejada e coloca na pilha de valor.

__evalArrayAssign -> Responsável por pegar os valores necessários para atribuição em uma posição de array e adiciona a pilha de controle.

__evalArrayAssignKW -> Faz a atribuição do valor numérico passado ao vetor e índice selecionado. Verificar instâncias do valor, para caso seja variável ou valores numéricos.

2. Especificação da extensão da gramática de Imp para listas de inteiros. (imp2.ebnf)

exp = array_len | bin_exp | array_exp | array_concat | array_val | paren_exp | un_exp | @:atom ; -> Adicionados array_len (Tamanho do array), array_exp (Identifica arrays), array_concat (Concatenação de array) e array_val (Retorna o valor do array no índice indicado) - Adicionado ao exp para poder ser usado em outros comandos.

array_val = idn:identifier "["e:exp"]" ; -> Identifica o acesso a um valor do array.

array_exp = "["e:exp {','e:exp}*"]" | "["e:exp"]" ; -> Identifica um array.

array_atrib = idn:identifier "["v:exp"]" op:":=" e:exp ; -> Identifica atribuição a um valor do array.

array_len = "@" idn:identifier ; -> Identifica o pedido de tamanho do array(# teve problemas na sua execução).

array_concat = l:array_exp "++" r:array_exp | l:array_exp "++" r:exp | l:exp "++" r:array_exp | l:exp "++" r:exp ; -> Concatenação de array, verifica para variáveis e para array por extenso.

atom_cmd = cond | loop | array_atrib | assign | print | call | skip ; -> Adição do array_atrib, verificando antes se é um array e evitando conflito com o assign.

3. Especificação das Pi denotações de Imp à Pi IR relativa às extensões realizadas. (impiler.py)

def array_exp(self, ast): -> Cria o objeto array.
return pi.Array(ast.e) -> O e contém o vetor por extenso.

def array_val(self, ast): -> Cria o objeto Array_val para acessar um valor do array.
return pi.Array_val(ast.idn, ast.e) -> Idn possui Id do array e ast.e possui índice.

def array_concat (self, ast): -> Cria o objeto para definir a concatenação de array.
return pi.Array_concat(ast.l, ast.r) -> L é operador da esquerda e R o da direita, podendo variar entre vetores por extenso ou usando Id.

def array_atrib(self, ast): -> Cria objeto para atribuição de valor em array
return pi.Array_atrib(ast.idn, ast.v, ast.e) -> Idn é o id do array, v é o índice a ser usado, ast.e é o valor a ser atribuído.

def array_len(self, ast): ->Cria objeto responsável por retornar o tamanho do array.
return pi.Array_len(ast.idn) -> Idn é o id do array