Compiladores 2020.1 Avaliação: Comando return

Daniel Zambrano

Extensão da gramática de Imp

Adicionada definição do comando de return na ebnf como **return_cmd**:

```
return_cmd = op:"return" e:exp;
```

Por ser um comando, foi também adicionado ao conjunto de comandos atômicos em **atom_cmd.** Além disso, foi adicionado **call** ao conjunto de expressões em **exp**, para que seja possível fazer a operação de atribuição (assign) de uma chamada de função a uma variável:

```
ex.:  y := fib(5);
```

Ainda mais, foi adicionado ao conjunto de palavras reservadas a palavra *return* através da marcação

```
@@keyword :: 'return'
```

Pi denotações de Imp à Pi IR para a extensão da gramática

A única mudança nas Pi denotações foi a inclusão da identificação de *return_cmd* que chama a nova classe do Pi Framwork: *Return.* Ela passa para o framework um único argumento, que é a expressão que deve ser retornada.

Novas equações do Pi Framework

Classe nova:

Definida uma nova nova classe *Return* que estende Cmd. Sua inicialização recebe um argumento que pode ser uma instância de Exp ou ListInt.

Mudanças realizadas:

Assign(Cmd): Agora o assign também poderá receber uma instância de Call, além de Exp e ListInt. Isso foi feito para que uma variável possa receber o conteúdo do return de uma função, como exemplificado mais acima.

CmdKW: Adicionada a chave de retorno #RETURN para identificação na pilha de controle.

DecCmdKW: Adicionadas as chaves #BLKCALL para indicar que um bloco pertence a uma *Call* e #BLKCMDCALL para indicar que um 'BLKCMD' veio de um bloco de *Call*. Essas duas chaves seguem a mesma lógica de construção e empilhamento das respectivas #BLKDEC e #BLKCMD, e servem somente para diferenciar na pilha de controle um bloco de comandos qualquer do bloco de comando que inicia o contexto da abstração.

__evalBlkDecKW: Agora a declaração é passada como argumento para dentro a função de avaliação, porque ela contém a informação sobre o tipo de bloco que deve ser inserido na pilha de controles (#BLKCMD ou #BLKCALL). o tipo da declaração é verificado e a chave correspondente é colocada no topo da pilha de controle.

eval (contexto de DecPiAut): Agora a função *__evalBlkDecKW* é chamada ao encontrar um #BLKDEC ou #BLKCALL, e a função __evalBlkCmdKW é chamada ao encontrar um #BLKCMD ou #BLKCMDCALL, para identificação do tipo de bloco explicado acima.

__evalCall: Agora é empilhado a nova chave #BLKCALL na pilha de controles em vez de #BLKCMD para identificar que a partir daquele momento começa uma abstração de uma função.

Funções de avaliação do *Return*:

__evalReturn: Ao ser chamada, o topo da pilha de controle é atualizado com a chave #RETURN e a instância da classe objeto que deve ser retornada.

__evalReturnKW: Quando chamada, o topo da pilha de valores contém o valor já calculado da expressão que deve ser retornada. Essa função deve então limpar a pilha de valores para sair do contexto de abstração e também ignorar todos os comandos que estão na pilha de controle dentro da mesma abstração.

Ao limpar a pilha de valores, três valores são importantes serem preservados: O valor calculado ‘v’, o ambiente ‘env’ e por último a expressão de associação do retorno da chamada ‘exp’, que ligará a variável de fora do escopo ao conteúdo de dentro. Para isso, ele limpa completamente a pilha de valores e insere novamente essas variáveis na nova pilha vazia.

Para limpar a pilha de controle, ele ignora todos os comandos até chegar em #BLKCALL, que é a chave usada ao iniciar uma chamada. A partir dela, tudo que entra na pilha de controle faz parte da abstração da função e pode ser ignorada quando o return for chamado.

Arquivo de testes

Foram criados 5 arquivos de testes:

cmd-test-factorial.imp2: Calcula o fatorial de um inteiro
cmd-test-fibonacci.imp2: Calcula o enésimo número da sequência de fibonacci
cmd-test-fibonacci-list.imp2: Retorna uma lista contendo os primeiros n números da sequência de fibonacci.
cmd-test-intervalo.imp2: Retorna 1,2,3 ou 4 dependendo do valor passado. Serve para mostrar que blocos de execução são ignorados quando um return é chamado.
cmd-test-return-list.imp2: Recebe uma lista e retorna uma nova lista com todos os elementos multiplicados por 2.