『プログラム言語論』 第3週 演習の略解
2014.5.7

課題 1.
MiniC 処理系は、「実行時のモード」という概念をもっている。モードは (現時点では)、0,1,2,3,4,5,6,7 の 8 つがあり、それぞれ、実行方法が異なっている。

- 動的束縛か静的束縛か。
- 値呼びか、名前呼びか、必要呼びか。
- 値呼びにおいて、複数の引数を持つ関数の呼び出しで、引数を左から右に順番に計算 (評価) するか、右から左に計算するか。(値呼び以外のときは、左からとか右からとかの区別はない。)

このためのプログラム例としては、ex4.c, ex6.c, ex14.c, ex15.c などを使うことができるが、自分でいろいろ試してほしい。(例題ファイル exN.c については、簡単な説明が EX-FILES というファイルに書いてある。)

略解
例題ファイルを実行する.

- ファイル “ex4.c”: 25 が印刷されれば動的束縛であり，30 が印刷されれば静的束縛である.
- ファイル “ex6.c”: 値呼びの場合に．1,2,3 の順に印刷されれば「引数を左から右の順番で処理」3,2,1 の順に印刷されれば「引数を右から左の順番で処理」である．(名前呼びや必要呼びの場合は，1,2,3 と印刷されるが，これは，引数を処理する順番ではなく，それらの引数を実際に使う順番で処理しているということである．)
- ファイル “ex14.c”: プログラムが停止しなければ値呼び，停止すれば名前呼びか必要呼びである．(名前呼びか必要呼びかは区別できない．)
- ファイル “ex15.c”: 10 と 20 が 1 回ずつ印刷されれば値呼び，10 が 5 回印刷されれば名前呼び，10 が 1 回だけ印刷されれば必要呼びである.

最終的な答えは以下の通り：

モード	静的/動的	左から右	値呼び/名前呼び/必要呼び
0	動的	左から	値呼び
1	静的	右から	値呼び
2	動的	右から	値呼び
3	静的	左から	値呼び
4	動的	–	名前呼び
5	静的	–	名前呼び
6	動的	–	必要呼び
7	静的	–	必要呼び

課題 2. (発展課題; 余力がある人のみ)
Fortran,C,C++,C#,Java,JavaScript,Scala, Ruby,Python,Perl,OCaml,F#,Haskell,Scheme,Lisp などの言語から、なるべく多くのものを取りあげ、

- 動的束縛か静的束縛か。
- 値呼びか、名前呼びか、必要呼びか、

- 関数の引数を左から評価するか右からか

について調べなさい。

答え (の一部):

これは，基本的には，各自で調べてもらいた課題であるので，一般的なことをいくつか述べるに留める．

- そもそも「関数呼びだし」がない言語もある．たとえば，オブジェクト指向言語では，それぞれのクラス (あるいはオブジェクト) において関数のように定義しているものは「メソッド」等と呼び，関数呼び出しのように見えているものは，メソッド呼び出しという名前で区別している．その理由はこの授業の後の方で述べる．

- 多くのプログラム言語において，関数呼び出し (あるいはメソッド呼び出し) における引数の処理については，静的呼び出し，かつ，値呼びである．

  「多くの」と書いた以上，そうでない言語もある．たとえば，Haskell は「必要呼び」のプログラム言語として有名なものである。これにより、たとえば、「無限に長いリスト」を生成してから、その先頭から 10 要素だけを取り出す、といったプログラムを実行しても、Haskell ではちゃんと停止するプログラムとなる。

- 値呼びの言語において，複数の引数をもつ関数の呼び出しで，左から処理されるか右から処理されるかは，プログラム言語ごとにばらばらである

  授業で言及したように C 言語の仕様では，左からとも右からとも決めていないどころか，もっと複雑な処理 (引数 1 の処理を途中まで行い，次に引数 2 の処理を行い，そのあと引数 1 の処理の残りをやる，など) でも良いようになっている．

  OCaml 言語の標準的な処理系は「右から左」に引数を処理しているが，OCaml の言語仕様としてはどちらであるかを定めていない．よって,「左から右」に処理する OCaml 言語処理系があっても構わない．