プログラム言語論

亀山幸義

筑波大学 情報科学類

No. 5:

変数の束縛 (復習)

変数の束縛: 変数の「宣言 (定義)」と「使用 (変数の値の参照)」の対応関係

- 静的束縛
- 動的束縛

関数呼び出しでの引数の渡し方

関数型言語での主要な関数呼び出し方法

- 値呼び call by value
- 名前呼び call by name
- 必要呼び call by need

命令型言語での主要な関数呼び出し方法 (後日)

- 値呼び call by value
- 参照呼び call by reference
- ... call by value return

実引数と仮引数:

```
let foo  x y = x + x in
  foo 20 (3 + 5)
```

x や y が仮引数 (formal parameter)
20 や 3+5 が実引数 (actual parameter)

関数型言語における引数の渡し方

値呼び (call by value):

```
let rec loop x   = loop x in
let show x   = print x; x in
let foo  x y = x + x in
  foo 20 (loop 10);
  foo (show 10) 20;
```

foo ... の呼び出し

- 実引数を計算してから、その値を関数に渡す。(仮引数に束縛する。)

関数型言語における引数の渡し方

名前呼び (call by name):

```
let rec loop x   = loop x in
let show x   = print x; x in
let foo  x y = x + x in
  foo 20 (loop 10);
  foo (show 10) 20;
```

foo ... の呼び出し

- 実引数を計算せずに、その式 (あるいは式へのポインタ) を関数に渡す。(仮引数に束縛する。)
- 関数本体の計算において、その値が必要になったときに、計算する。

関数型言語における引数の渡し方

ある人が、講演の最初に以下のように言った。

- 何か質問はありますか？(lazy evaluation)
- 私は同じ質問には１度しか回答しません。(only once)

関数型言語における引数の渡し方

必要呼び (call by need):

```
let rec loop x   = loop x in
let show x   = print x; x in
let foo  x y = x + x in
  foo 20 (loop 10);
  foo (show 10) 20;
```

foo ... の呼び出し

- 実引数を計算せずに、その式 (あるいは式へのポインタ) を関数に渡す。
- 関数本体の計算において、その値が初めて使われるときに計算する。
- その値が 2 回目に必要なときは、最初に計算した値を再利用する。

演習問題

以下のプログラムを 3 通りの関数呼び出しで評価した結果 (プリントされるもの) を示しなさい。

```
let rec loop x   = loop x in
let show x   = print x; x in
let foo  x y = x + x in
  foo 20 (loop 10);
  foo (show 10) 20;
```

関数型プログラム言語における引数の渡し方

call by value	Lisp, Scheme, ML,...
call by name	–
call by need	Haskell

なお、参照 (C 言語のポインタ相当のもの) を使うなどをして、他の呼び出し方をある程度シミュレートすることは可能。