

プログラミング演習B ML編 第1回

2011/5/10 (コミ)
2011/5/11 (情報・知能)
住井

[http://www.kb.ecei.tohoku.ac.jp/
~sumii/class/proenb2011/ml1/](http://www.kb.ecei.tohoku.ac.jp/~sumii/class/proenb2011/ml1/)

今日のポイント

1. MLって何？
2. 式を「評価」すると値になる
3. 式や値には「型」がある

レポートについて

電気・情報系内のマシンから

<http://130.34.188.208/> (情報・知能)

<http://130.34.188.209/> (コミ)

にアクセスし、画面にしたがって提出せよ。締め切りは**一週間後厳守**。

- 初回は画面にしたがい自分のアカウントを作成すること。
- 「プログラム」のテキストボックスがある課題では、プログラムとしてsmlに**入力**した文字列のみを**過不足なく正確に**コピー&ペーストして提出せよ。
(smlの**出力**は「プログラム」ではなく考察に含めて書くこと。)
- プログラムの課題でも必ず考察を書くこと。
- 提出したレポートやプログラムの実行結果は「提出状況」から確認できる。
 - 質問はm1-enshu@kb.ecei.tohoku.ac.jpにメールせよ。
 - レポートの不正は試験の不正と同様に処置する。

ポイント1

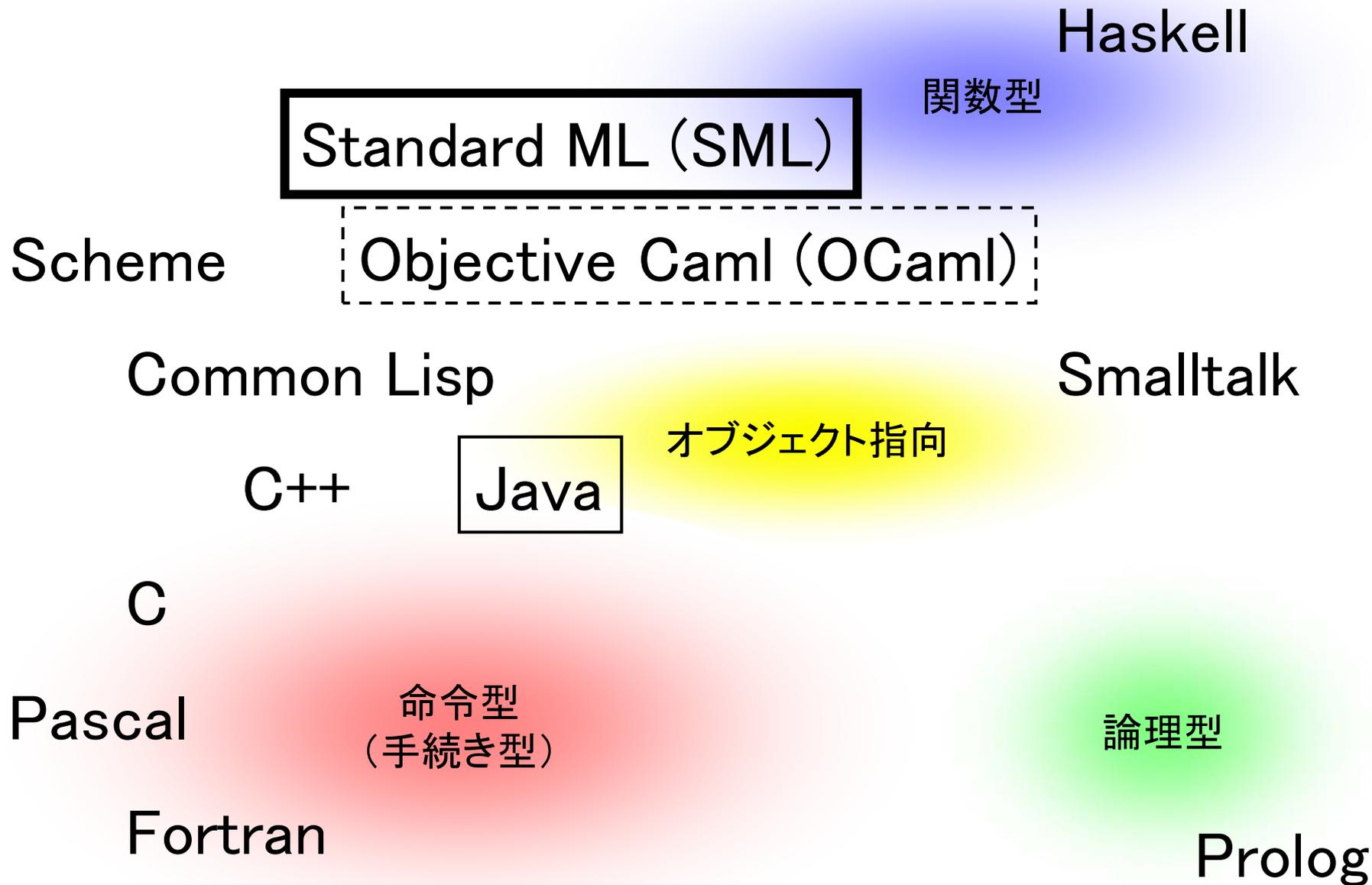
Q:

MLって何？

A:

**「簡単・安全・高速」な
プログラミング言語の名前
（「関数型言語」の一種）**

- ここでは" Mailing List" や" Markup Language" ではない
 - MatLab (科学技術計算ソフトウェア) のことでもありません



なぜMLを学ぶのか？

- 1960年代(Lisp)以来、関数型言語は**プログラミング言語の最先端**
 - ガベージコレクション、高階関数／クロージャ、多相型／総称型、型推論等
 - 数十年遅れでJava, C++等に導入
 - ここ数年、国内外で再注目
 - 各種書籍、雑誌記事、コンファレンス等
- MLは**代表的関数型言語**の一つ
 - 他にHaskell, Scheme等

課題 1.1

Wikipedia (<http://ja.wikipedia.org/>)で「プログラミング言語一覧」の項目を調べる等して、何か一つの言語 (C, Java, SML, OCaml以外) について、どのような言語か数行程度で述べよ。

- (特に日本語の) インターネット上の情報は不正確なことも多いので、一つないし少数の記述を鵜呑みにしないこと。
- 他人の文章を (一部でも) 剽竊しないこと。

剽窃は不正行為です

剽窃：引用であることを明示せず、他人の文章の一部または全体を少し変えただけで（あるいはまったく変えずに）用いること

剽窃になる例：C言語は、AT&Tベル研究所のデニス・リッチーが主体となって1972年に作られたプログラム言語。UNIXのために開発された経緯から、OSカーネル向けの低レベルな記述が可能。文書や文脈によっては単にCと呼ばれる。

<http://ja.wikipedia.org/wiki/C言語>
(2010年6月19日(土) 06:46版) に酷似

剽窃をしないために

- **内容を理解し自分の言葉で書き直す**
- **参考文献は明記する**
 - **たとえ参考文献として明記しても剽窃は不正行為**
 - **議論のために引用する場合は、引用記号（「」や"”）をつけ、引用元の書籍名・著者名・ページ番号等を明記する**
- **引用と剽窃の違い：**
 - 引用であることを明示しているかどうか

剽窃にならない例

C言語は手続き型プログラミング言語の一つ。1972年頃にベル研究所のテニス・リッチーらにより、UNIXオペレーティングシステムを記述するために開発された。ポインタ演算など、アセンブリ言語に近い比較的低水準の記述が容易な反面、欠陥のあるプログラムも記述しやすい。

(参考文献 <http://ja.wikipedia.org/wiki/C言語>)

- ただし（特に日本語の）インターネット上の情報は玉石混濁なので要注意（Wikipediaを含む）

レポート採点方針

文章や論理の正しさを含む
「内容」を見ます

「ちゃんとわかっているのに評価されないのは不当ではないか」

→ 実はわかっていない and/or 自分の考えを論理的に表現できない

レポートの問題点の指摘は提出者の人格を否定するものではありません

課題 1.2

電気系教育用計算機システムなどで
次の操作をし、結果を述べよ。

1. ktermなどにおいてコマンド`ocaml`を起動
2. `ocaml`に`#load "graphics.cma" ;;`
と入力してEnterキー（`#`も入力すること）
3. さらに`Graphics.open_graph "" ;;`
4. `Graphics.draw_circle 100 100 50 ;;`
5. `exit 0 ;;`

（わからなくなったらControlキーを押しながら
`c`や`d`を連打すれば終了するのでやりなおす）

課題 1.3

(1/3)

近くの人とペアを組んで
次の操作をし、結果を述べよ。

1. 1人目は `ifconfig -a` を実行し、
マシンのIPアドレスを確認する
(130.34.195.11~18のはず)
2. さらに、1人目は10000以上30000以
下の適当な整数（ポート番号）を、
他の人と重ならないように決める

課題 1.3

(2/3)

3. 1人目はocamlを起動し、
以下のプログラムを実行して待機する

```
#load "unix.cma" ;;  
open Unix ;;  
establish_server  
  (fun ic oc ->  
    print_endline (input_line ic))  
(ADDR_INET  
  (inet_addr_any, ポート番号)) ;;
```

課題 1.3

(3/3)

4. 2人目はocamlを起動し、以下のプログラムを実行する

```
#load "unix.cma" ;;  
open Unix ;;  
let (ic, oc) =  
    open_connection  
    (ADDR_INET  
     (inet_addr_of_string "IPアドレス",  
      ポート番号)) ;;  
output_string oc "Hello, world!¥n" ;;  
close_out oc ;;
```

ポイント1 おわり

ここからは、OCamlではなく
SMLをやります

(6セメの授業「コンパイラ」でSMLを使用するので)

OCamlについて、もっと知りたい人は...

- 「プログラミング in OCaml」 (ISBN 978-4-7741-3264-8)
- 「プログラミングの基礎」 (ISBN 978-4-7819-1160-1)
- 「入門OCaml」 (ISBN 978-4-8399-2311-2)
- <http://www.google.co.jp/search?q=ocaml>

参考書

「プログラミング言語
Standard ML入門」

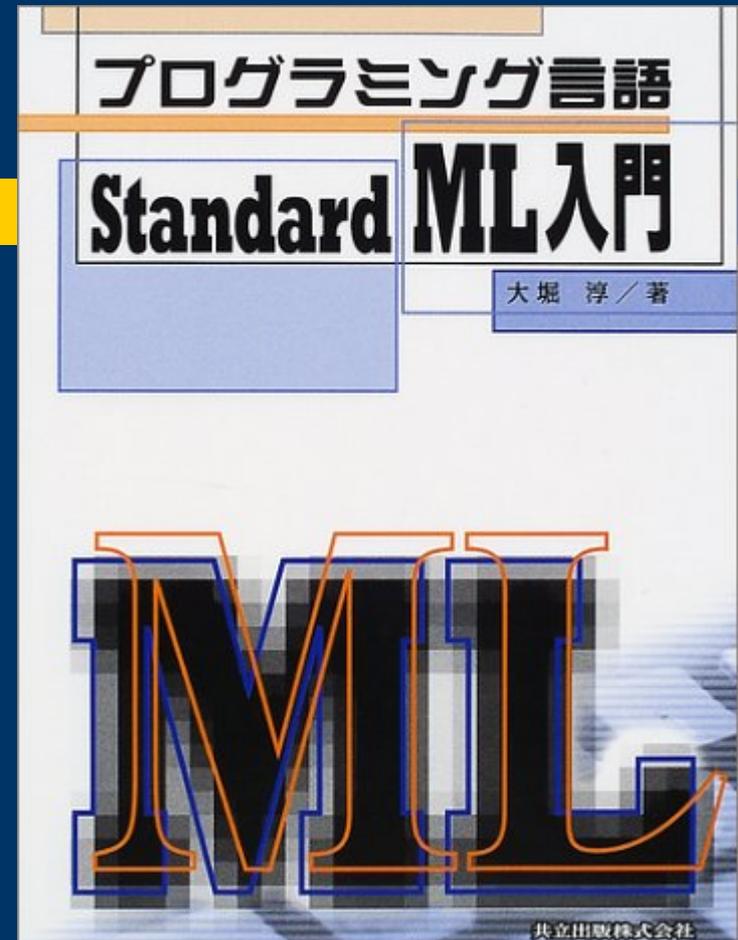
大堀淳

共立出版

ISBN

978-4-320-12024-2

<http://www.pllab.riec.tohoku.ac.jp/~ohori/texts/mltext.html>



その他の本

- 「プログラミング言語ML」
(ISBN 978-4-7561-1641-3)
- "ML for the Working Programmer"
(ISBN 978-0-5215-6543-1)
- "The Definition of Standard ML -
Revised"
(ISBN 978-0-262-63181-5)

SMLの起動と終了

起動：

ktermなどで **sm1** コマンドを実行

- または emacs で

Esc x **run-sm1** Enter Enter

終了：**C-d**

(Controlキーを押しながらd)

入力や計算の中断：**C-c**

ポイント2

式を「評価」すると値になる

- 式の後に;`;`を入力してEnterを押すと式の値が計算される

```
> sm1
```

```
Standard ML of New Jersey, Version 110.0.7,  
September 28, 2000 [CM; autoload enabled]
```

```
- 1+2;
```

```
val it = 3 : int
```

```
-
```

- このように式の値を計算することを「評価」(evaluation)という

いろいろな式と値 (1/2)

- 整数: $0, 123, \sim 456$ など
- 浮動小数点数: $0.0, 1.23, \sim 4.56$ など
注: SMLでは負の数は $-$ ではなく \sim で書く
- 算術演算: $\text{式}_1 + \text{式}_2, \text{式}_1 - \text{式}_2, \text{式}_1 * \text{式}_2$
- 商と余り: $\text{式}_1 \text{ div } \text{式}_2, \text{式}_1 \text{ mod } \text{式}_2$
- 浮動小数点数の割り算: $\text{式}_1 / \text{式}_2$
- 切り下げ, 切り上げ, 切り捨て, 偶数丸め:
 $\text{floor } \text{式}, \text{ceil } \text{式}, \text{trunc } \text{式}, \text{round } \text{式}$
- 整数から浮動小数点数への変換: $\text{real } \text{式}$

いろいろな式と値 (2/2)

- 論理値: `true`, `false`
- 論理演算:
`not 式`, `式1 andalso 式2`, `式1 orelse 式2`
- 比較: `式1 = 式2`, `式1 <> 式2`, `式1 >= 式2` など
- 文字列: `"abcde"`, `"Hello, world!¥n"` など
- 文字列の連結: `式1 ^ 式2`
- カッコつき式: `(式)`
- スペースのところには、空白やタブや改行やコメントを好きなだけ入れてよい
 - コメントは `(*と*)` で囲む

課題 1.4

次の式を評価してみて、結果を考察せよ。

1. $123 + 456$

2. $1 + 2 * 3$

3. $(1 + 2) * 3$

4. $7 - -8$

5. $7 - \sim 8$

6. $10 \text{ div } 3$

7. $\sim 10 \text{ div } 3$

8. $\sim 10 \text{ mod } 3$

9. $10.0 / 3.0$

ポイント3

式や値には「型」がある

型 = 式や値の種類

- 整数型 `int`, 浮動小数点数型 `real`,
論理値型 `bool`, 文字列型 `string`,
etc.

静的型検査と型エラー

- MLは評価（実行）の前に型をチェックする
- 型が合わなければ評価せずエラーとする

```
- 1.2 / 3.0 ;
```

```
val it = 0.4 : real
```

```
- 1.2 / 3 ;
```

```
stdIn:18.1-18.8 Error: operator and operand  
don't agree [literal]
```

```
operator domain: real * real
```

```
operand:          real * int
```

```
in expression:
```

```
  1.2 / 3
```

```
- 1.2 / real 3 ;
```

```
val it = 0.4 : real
```

課題 1.5

今までの例と課題以外の、様々な式の評価を試みよ（エラーになる例も試せ）。できるだけ自明でない例を少なくとも3つ考察せよ。