工学部専門科目
「プログラミング言語」
(その1: 導入)

五十嵐 淳

igarashi@kuis.kyoto-u.ac.jp

京都大学 大学院情報学研究科
通信情報システム専攻

April 14, 2015

事務的情報

- **名前**: **五十嵐 淳 (いがらし あつし)**
- **所属**: **情報学研究科 通信情報システム専攻 計算機ソフトウェア分野**
- **オフィス**: **総合研究 7 号館 224 号室**
  - **オフィスアワー**: **月曜** 17:00〜18:30
- **よろず相談窓口**: `pl15@fos.kuis....`
  - TA **の大元 武君，樹下 稔君と五十嵐に連絡できます**
- **講義** WWW **ページ**: `http://www.fos.kuis.kyoto-u.ac.jp/~igarashi/class/pl/`

成績評価

- **宿題** 30%
  - **宿題提出システムへの登録が必要 (後述)**
- **期末試験** 70%
- **随意課題によりさらに加点**

講義内容

シラバスより

プログラミング言語についてコンピュータサイエンスの立場から論じる．使用するプログラミング言語はScheme であり，高度なプログラミングの概念について学ぶとともに，実際にプログラミングに適用することを通じて，プログラミングの本質を習得する．
教科書の前半（第１章および第２章）は「アルゴリズムとデータ構造入門」（第 1 学年後期配当，91150）で取り上げ，本講義では第３章および第４章を取り上げる．

第3章: Modularity, Objects, and State

状態変化を伴う部品を使ったシステム記述について

- Modularity(**モジュール性**)…**システム全体を，より小さく，独立して開発/保守可能な部分システム(モジュール)に「自然に」分割可能であること**
- Objects(**オブジェクト**)…**(時間を経て変化する)内部状態を持つシステム部品**
- State…**状態**

第4章: Metalinguistic Abstraction

超言語的抽象について

- **プログラミング言語自体 (の意味) の変更による抽象化の実現**
  - Scheme **処理系を** Scheme **自身で記述**
  - **処理系を変更して「**Scheme **の亜種」を作る，特殊形式を追加する**

注意!!(特に３回生)

- この内容 (SICP3，4 章) での「プログラミング言語」の講義は**今年度が最後**です．
  - 計算機科学コースのカリキュラム改定による
- 来年度からは２回生**後期配当**
- ４回生後期に必修科目で内容も初めての科目が残っている，なんてことのないように

SICP 第1〜2章の復習

問題1: 正整数 n に対し，階乗 n! を返す手続 factorial を定義せよ

```
(define (factorial n)



)
```

問題2: 問題1 で定義した factorial を使って5! を計算する過程を示せ

```
(factorial 5)
```

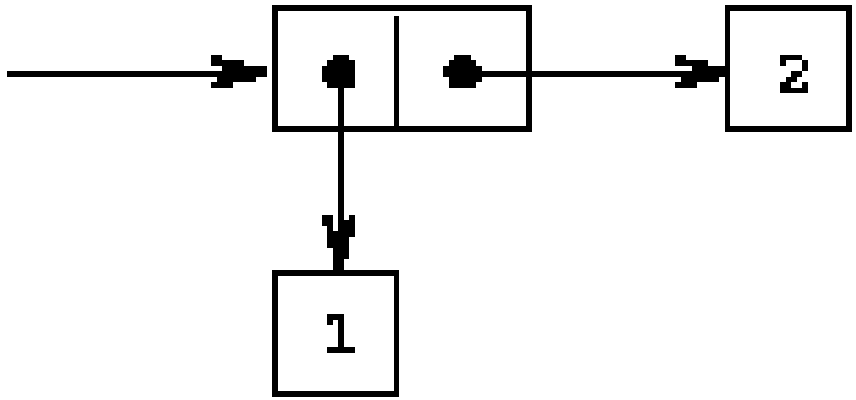
$\longrightarrow$

問題3: 以下の入力列を与えた時(最後の入力に対する) Scheme 処理系からの応答はどうなるか答えよ.

```
(define (foo x)
  (lambda (y) (* (+ 2 x) y)))
(define x 6)
(define bar (foo 5))
(define x 1)
(bar 4)
```

問題4: 以下の式を評価した結果，表示される結果と作られる構造を図示せよ

- **例**: `(cons 1 2)` $\Longrightarrow$ `(1 . 2)`

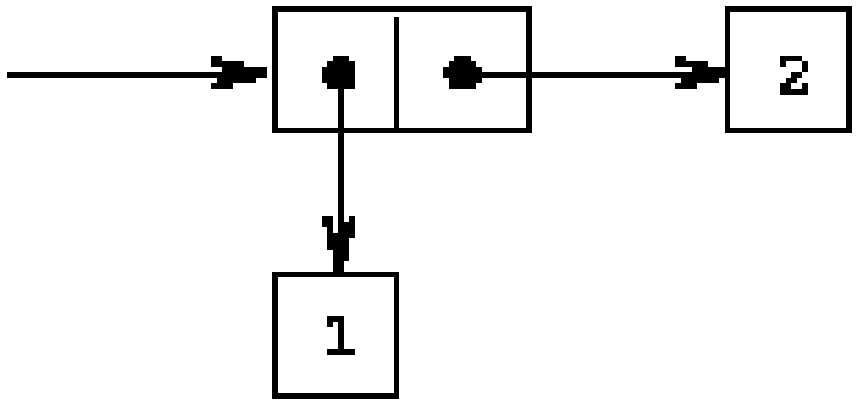


- `(cons 1 nil)`
- `(list 1 2)`
- `(cons (cons 1 2) 3)`
- `(cons 1 (list 2 3))`
- `(list (list 2 3) 4)`

- `(cons (cons 1 2) (cons 1 2))`
- `(let ((z (cons 1 2))) (cons z z))`

問題5: x を評価した結果，以下が表示される時，x の指す構造を図示せよ．また x に car, cdr を使って 2 を得る方法を示せ．

- **例**: (1 . 2)　　**答**: (cdr x)



- (4 3 2)
- (7 (3 . 4) (5 2))
- (((2)))

問題6: 手続き length を以下のように定義する．この時，以下の式の値は何か

```
(define (length xs)
  (if (null? xs) 0
      (+ 1 (length (cdr xs)))))
```

- `(length '(4 8 2))`
- `(length '((1 2) 3 (4 5 (6))))`

宿題(1)： 締切 4/21 10:30

問題 1〜6 を解いて、紙で提出できるようにしてくること。

宿題(2)： 4/21 午前10:30 締切

- 教科書 3.1 までを読み，以下の問いについて考えておくこと
  - 「銀行口座」における状態とは何か？
  - 手続き `withdraw` が今までの手続きとは根本的に異なる挙動を示している，という説明があるが，それはどういうことか？
  - `withdraw` と `new-withdraw` の違いは何か？
  - (擬似)乱数を状態を使って実現することのメリットは何か？
  - 破壊的代入 (assignment) による状態変化を導入することによって何が複雑になるといっているか？