# 2005年度「計算機ソフトウェア工学」
# λ計算入門 練習問題

授業で定義した純粋なλ計算において、次の式の簡約結果を、1ステップごとの簡約過程と共に示せ。ただし、どう簡約しても停止しないときは、そう述べた上で、理由を説明せよ。また、`let`式・自然数・論理値・組・再帰関数などは、授業で定義したλ式による表現を使用せよ。

例1. $(\lambda a.b)cd$

$$(\lambda a.b)cd \quad \to \quad bd$$

例2. $(\lambda x.\lambda y.xy)y$

$$(\lambda x.\lambda y.xy)y \quad \to \quad \lambda y'.yy'$$

例3. $1+1$

$$\begin{aligned}
1+1 &= \lambda s.\lambda z.1s(1sz) \\
&= \lambda s.\lambda z.1s((\lambda s.\lambda z.sz)sz) \\
&\to \lambda s.\lambda z.1s((\lambda z.sz)z) \\
&\to \lambda s.\lambda z.1s(sz) \\
&= \lambda s.\lambda z.(\lambda s.\lambda z.sz)s(sz) \\
&\to \lambda s.\lambda z.(\lambda z.sz)(sz) \\
&\to \lambda s.\lambda z.s(sz)
\end{aligned}$$

例4. $(\lambda x.xx)(\lambda x.xx)$

どう簡約しても停止しない。

理由：与式を $e$ とおく。$e$ は関数適用であって $\lambda$ 抽象ではないから、規則 (R-Abs) による簡約はできない。また、$e$ の関数部分 $\lambda x.xx$ および引数部分 $\lambda x.xx$ は簡約できないので、(R-App1) および (R-App2) も適用できない。したがって、(R-Beta) による

$$\begin{aligned} e &= (\lambda x.xx)(\lambda x.xx) \\ &\to (\lambda x.xx)(\lambda x.xx) \\ &= e \end{aligned}$$

という簡約のみが可能である。すなわち、任意の $e'$ について、$e \to e'$ ならば $e = e'$ である。すると「任意の $e'$ について、$e \to^* e'$ ならば $e = e'$ である」ことが $e \to^* e'$ の導出に関する帰納法よりいえる。したがって、もし $e \to^* e' \not\to$ ならば $e = e' \not\to$ のはずだが、これは $e \to e$ と矛盾する。よって、$e \to^* e' \not\to$ となる $e'$ は存在しない、つまり $e$ はどう簡約しても停止しないことがわかる。 □

1. $(\lambda x.x)y$
2. $(\lambda x.x)x$
3. $(\lambda x.x)(\lambda x.x)$
4. $(\lambda x.y)(\lambda z.z)$
5. $(\lambda x.y)x$
6. $(\lambda x.y)y$
7. $(\lambda x.y)xy$
8. $(\lambda x.y)xx$
9. $(\lambda a.a)(\lambda b.b)((\lambda c.c)(\lambda d.d))$
10. $(\lambda a.a)(\lambda b.b)(\lambda c.c)(\lambda d.d)$
11. $(\lambda x.x)(\lambda x.x)((\lambda x.x)(\lambda x.x))$
12. $(\lambda x.xx)(\lambda y.yy)$

13. $(\lambda y.z)((\lambda x.xx)(\lambda x.xx))$

14. $(\lambda y.y)((\lambda x.xx)(\lambda x.xx))$

15. $(\lambda x.xx)(\lambda x.xx)(\lambda y.z)$

16. $(\lambda x.xx)(\lambda x.x)$

17. `let` $f = \lambda x.x$ `in` $fy$

18. `if false then` $x$ `else` $y$

19. `let` $\text{my\_not} = \lambda x.$(`if` $x$ `then false else true`) `in` $\text{my\_not}$ `false`

20. `let` $\text{my\_and} = \lambda x.\lambda y.$(`if` $x$ `then` $y$ `else false`) `in` $\text{my\_and}$ `true false`

21. `match` $((\lambda x.x), y)$ `with` $(g, z)$ `->` $gz$

22. $2 \times 2$

23. `let` $\text{is\_zero} = \lambda n.n(\lambda m.$`false`$)$`true in` $\text{is\_zero}\ 2$

24. `let rec` $fx =$ (`if` $x$ `then` $f$ `false else` $y$) `in` $f$ `true`

25. `let rec` $fx =$ (`if` $x$ `then` $f$ `false else` $f$ `true`) `in` $f$ `true`