プログラミング言語**特**論（2011年度）・テスト問題用紙

(2012年2月9日（木）・ 13:00 〜 14:30)

解答上、その他の注意事項

I. 問題は、問I〜Vまである。

II. 解答用紙の右上の欄に学籍番号・名前を記入すること。

III. ノート・プリント・参考書などは持ち込み可である。

IV. 携帯電話などの通信機能を持つものは持ち込み不可である。

V. 問Iを解答するときのみ、ノートPCを使用して良い。ネットワークに接続してWWWを閲覧しても良いが、掲示板、チャット、メールなどで生身の人間と通信することは禁じる。

VI. テストの配点は50点（+ボーナス20点）である。合格はレポートの得点を加点して、100点満点中60点以上とする。

I. （Haskell 実習問題） （6 点 × 2）

(1) 引数として与えられる整数の組のリスト中の `x+y==0` を満たす要素 `(x, y)` の個数を返す関数

```
foo :: [(Integer, Integer)] -> Integer
```

を定義せよ。

例えば、`foo [(1, -1), (0, 0), (2, -3)]`は 2 であり、`foo [(1,-2), (5, -4), (2, 2)]` は `0` である。

この問では `map`, `filter`, `foldl`, `foldr` などのリストに関するライブラリ関数や内包表記を使わず、`if` 〜 `then` 〜 `else` 〜 式や四則演算子、論理演算子、等号・不等号、パターンマッチング、再帰などを使って定義せよ。

(2) 整数 `n` を引数として受け取り、正の整数の組 (i, j) で、$i^2 \leq j$ かつ $j \leq n$ が成り立つものを列挙する関数:

```
bar :: Integer -> [(Integer,Integer)]
```

を（リストの内包表記を用いて）定義せよ。

例えば、`bar 4` は `[(1,1),(1,2),(1,3),(1,4),(2,4)]` で、
`bar 6` は `[(1,1),(1,2),(1,3),(1,4),(1,5),(1,6),(2,4),(2,5),(2,6)]` となる。
（リストの要素の順番はこのとおりでなくても良い。）

II. （ラムダ計算）（6 点 × 2）

次の λ 式が正規形に到達するまでの、**最左変換**による 1 ステップずつの β 変換の列を書け。ただし、10 回以内の最左変換で正規形に到達しない式については、それが判別できる時点（以前と同じ式が出現した時点）、または 10 回最左変換した時点で止めてよい。

解答例 1:

$$
\begin{aligned}
& (\lambda f x.f(fx))((\lambda f x.f(fx))g)y \\
\xrightarrow{\beta}\ & (\lambda x.((\lambda f x.f(fx))g)(((\lambda f x.f(fx))g)x))y \\
\xrightarrow{\beta}\ & ((\lambda f x.f(fx))g)(((\lambda f x.f(fx))g)y) \\
\xrightarrow{\beta}\ & (\lambda x.g(gx))(((\lambda f x.f(fx))g)y) \\
\xrightarrow{\beta}\ & g(g(((\lambda f x.f(fx))g)y)) \\
\xrightarrow{\beta}\ & g(g((\lambda x.g(gx))y)) \\
\xrightarrow{\beta}\ & g(g(g(gy)))
\end{aligned}
$$

解答例 2:

$$
\begin{aligned}
& (\lambda x.xx)(\lambda x.xx) \\
\xrightarrow{\beta}\ & (\lambda x.xx)(\lambda x.xx) \\
\xrightarrow{\beta}\ & \text{（停止しない）}
\end{aligned}
$$

(1) $(\lambda abc.abc)(\lambda xy.x)(\lambda f x.x)(\lambda f x.fx)$

(2) $(\lambda xy.x(\lambda uv.u)y)(\lambda xy.y)(\lambda xy.x)$

なお、必要に応じて $I \equiv \lambda x.x$ など適宜、定数を定義しても良い。

III. （Haskell）　（7 点 ×2）

次の例にならって、下の Haskell のプログラム (1)〜(2) を評価した結果を書け。

例: `take 5 (from 1)` ⇒ **評価結果:** `[1,2,3,4,5]`

ただし、`take` と `from` は講義プリントに定義されているとおりの関数である。

```
from :: Integer -> [Integer]
from n = n : from (n+1)

take :: Integer -> [a] -> [a]
take 0 _      = []
take _ []     = []
take n (x:xs) = x : take (n-1) xs
```

(1) `take 6 (iterate (\ (x,y) -> (2*x+y, x)) (1,0))`

この問で使用されている関数 `iterate` の定義は次のとおりである。

```
iterate     :: (a -> a) -> a -> [a]
iterate f x = x : iterate f (f x)
```

(2) `[ (x,y) | x <- [1,3,5], y <- [2,4,6], y<=2*x ]`

（この問に関してはリスト内の順番のみの間違いは、減点はしない。）

IV. （語句） （6 点 × 2）

プログラミング言語（やその処理系）で用いられる次の 6 つの語句のうち 2 つを選択し、具体的な例を挙げて説明せよ。ただし、講義プリントにのっている例ではなく、オリジナル の例を考えること。

- 抽象構文 (abstract syntax)
- 動的束縛 (dynamic binding)
- 高階関数 (higher-order function)
- 遅延評価 (lazy evaluation)
- 接続（あるいは継続） (continuation)
- CPS (continuation passing style)

V. （自由記述 — ボーナス問題） （最高 20 点）

小学生５年生のいとこに「大学で何を勉強しているの？」と訊かれました。小学生５年生にわかるように「コンピューター」・「プログラミング」とは何か、情報系の学科で何を勉強するか、説明してみよ。

プログラミング言語**特**論（2011 年度）・テスト解答用紙 (’12 年 2 月 9 日)

学籍番号		氏名	

学籍番号		氏名	