プログラミング言語（2006年度）**特**論・テスト問題用紙

('07年2月8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リストに 80 以上の数が 1 つでも存在すれば `True`、1 つもなければ `False` を返す関数 `foo` を定義せよ。

例えば、`foo [24,79,68]` は `False`、`foo [44,89]` は `True` となる。

この問では `map`, `filter` などのライブラリ関数や内包表記を使わず、`if`〜`then`〜`else`〜式や論理演算子、不等号、パターンマッチ、再帰などのみを使って定義せよ。

(2) 引数として与えられるリストの中で、互いに素（つまり、最大公約数が 1）で、$x < y$ がなりたつ 2 つの要素の組 (x, y) を列挙する関数 `bar` を（リストの内包表記を用いて）定義せよ。

ただし、2 つの引数の最大公約数を求める関数 `gcd` は Haskell の標準ライブラリに用意されているので、利用して良い。

```
Prelude> gcd 12 18
6
Prelude> gcd 11 30
1
```

例えば、`bar [2,3,4,5]` は `[(2,3),(2,5),(3,4),(3,5),(4,5)]`、`bar [2,3,6,10]` は `[(2,3),(3,10)]` 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、正規形が存在しない式については、それが判別できる時点（ただし少なくとも 3 回以上 β 変換したあと）に $\cdots$ と記入せよ。

記入例:

$$
\begin{array}{ll}
 & (\lambda fx.f(fx))((\lambda fx.f(fx))g)y \\
\xrightarrow{\beta} & (\lambda x.((\lambda fx.f(fx))g)(((\lambda fx.f(fx))g)x))y \\
\xrightarrow{\beta} & ((\lambda fx.f(fx))g)(((\lambda fx.f(fx))g)y) \\
\xrightarrow{\beta} & (\lambda x.g(gx))(((\lambda fx.f(fx))g)y) \\
\xrightarrow{\beta} & g(g(((\lambda fx.f(fx))g)y)) \\
\xrightarrow{\beta} & g(g((\lambda x.g(gx))y)) \\
\xrightarrow{\beta} & g(g(g(gy)))
\end{array}
\qquad
\begin{array}{ll}
 & (\lambda x.xx)(\lambda x.xx) \\
\xrightarrow{\beta} & (\lambda x.xx)(\lambda x.xx) \\
\xrightarrow{\beta} & (\lambda x.xx)(\lambda x.xx) \\
\xrightarrow{\beta} & (\lambda x.xx)(\lambda x.xx) \\
\xrightarrow{\beta} & \cdots
\end{array}
$$

(1) $(\lambda xy.yx)(\lambda xy.y)(\lambda zw.z)$

(2) $(\lambda nfx.n(\lambda gh.h(gf))(\lambda w.z)(\lambda x.x))(\lambda fx.f(fx))$

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)` ⇒ **答:** `[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) `foldr (\ x y -> 2 * x + y) 0 [1,2,3]`

この問で使用されている関数 `foldr` の定義は次のとおりである。

```
foldr :: (a -> b -> b) -> b -> [a] -> b;
foldr f z []     = z;
foldr f z (x:xs) = f x (foldr f z xs);
```

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

（この問に関してはリスト内の順番の間違いの減点は 1 点のみとする。）

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）
- 環境（environment）
- 非決定性（nondeterminism）
- 多重定義（overloading）

V. （自由記述 — ボーナス問題） （最高 20 点）

プログラミング言語特論の講義では、リストを反転する関数の 2 種類の実装 `rev` と `reverse` が等価である（任意の有限リストに対し、等しい結果を返す）ことの証明を取り上げた。

これまでにいろいろなプログラムを作成した経験から、等価性以外にプログラムを 実行せずに 証明したいプログラムの性質の例を挙げ、できるだけ 具体的に 説明せよ。

プログラミング言語**特**論（2006年度）・テスト解答用紙('07年2月8日)

学籍番号		氏名	

学籍番号		氏名	