プログラミング言語**特**論・テスト問題用紙

('03 年 2 月 6 日（木）・ 13:00 〜 14:30)

解答上、その他の注意事項

I. 問題は、問 I〜III 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. 解答欄を間違えないよう注意すること。

IV. 選択式でない問で解答欄がマス目になっている場合は、1 字に 1 マスを用いること。特に空白にも必ず 1 マスを用いること

V. 解答中の文字 (特に a と d) がはっきりと区別できるよう注意すること。

VI. 教科書・ノート・プリント・参考書・パソコンなどは持ち込み可である。ただし、パソコンのネットワーク機能は使用してはいけない。

VII. 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。

VIII. テストの配点は 50 点である。（第 1 回レポート 10 点・第 2 回レポート 20 点・第 3 回レポート 20 点）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. （λ 計算） 次の λ 式の正規形を書け。正規形が存在しないものについては、$\times$ を書け。

(1) $(\lambda fx.f(fx))((\lambda fx.f(fx))g)y$

(2) $(\lambda xyz.xz(yz))(\lambda x.x)(\lambda x.x)$

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

II. （Haskell）数列 $\{a_n\}$ をリストとして表す時、

(1) はじめて 100 を超える項、つまり $a_n > 100$ かつ $\forall i < n.\ a_i \leq 100$ となるような a_n を求める関数: `gt100`

(2) 前項との差がはじめて 10 を超える項、つまり $a_n - a_{n-1} > 10$ かつ $\forall i < n.\ a_i - a_{i-1} \leq 10$ となるような a_n を求める関数: `gt10d`

を Haskell で定義せよ。

ちなみに次のように定義される素数の列`primes`では、`gt100 primes`は`101`で、`gt10d primes`は`127`である。

```
from n = n : from (n+1)
sieve (x:xs) = filter (\ y -> y `mod` x /= 0) xs
primes = map head (iterate sieve (from 2))
```

III. （再帰の除去）次のように定義される関数sum, reverse を引数の数を増やすことにより末尾再帰に書き換え、さらに再帰を使用せずに**for** や**while** などの繰り返しの構文を利用する（C の）プログラムに書き換えよ。（解答には再帰を使用しない形のみ書けば良い。関数名は元のままのsum とreverse にせよ。）

```
list sum(list xs) {
  if (xs==NULL) {
    return 0;
  } else {
    return xs->car+sum(xs->cdr);
  }
}

list reverse(list xs)  {
  if (xs==NULL) {
    return xs;
  } else {
    return append(reverse(xs->cdr), cons(xs->car, NULL));
  }
}
```

ただし、このプログラムの中で使われている構造体や関数の定義は次のとおりである。

```
struct _list {
  int car;
  struct _list* cdr;
};

typedef struct _list* list;

list cons(int x, list xs) {
  list ret = (list)malloc(sizeof(struct _list));
  ret->car = x;
  ret->cdr = xs;
  return ret;
}

list append(list xs, list ys) {
  if (xs==NULL) {
    return ys;
  } else {
    return cons(xs->car, append(xs->cdr, ys));
  }
}
```

なお、次の Haskell のプログラムを再帰として末尾再帰のみ利用する（Haskell の）プログラムに書き直してもよい。この場合は、繰り返しに書き換える必要はない。

```
mySum []     = 0
mySum (x:xs) = x + mySum xs

myReverse []     = []
myReverse (x:xs) = myReverse xs ++ [x]
```

プログラミング言語**特**論・テスト解答用紙('03 年 2 月 6 日)

学籍番号		氏名	

I. (5 点 ×3)

(1).		(2).		(3).	

II. (8 点 ×2)

(1).	
(2).	

III. (9 点, 10 点)

(1).	
(2).	

テストの感想

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……