『プログラム言語論』 期末試験

解答用紙は 2 枚であり、表と裏の両方を使用できる。すべての解答用紙の上部に、学籍番号と氏名を明記すること．問題の順番通りに解答する必要はないが、必ず、問題番号 (「1-a」など) を記載してから解答を書くこと．

問 1. (配点 30 点)

次のコードは、あるプログラム言語 (Simple と呼ぶことにする) の構文と意味 (評価器,evaluator) を、OCaml 言語で記述したものである。

なお、これらは、演習で学習したものと、lookup 関数の定義を省略した以外は、同一である。問 1-a 以外の解答は、ここで与えた構文通りでなくてよく、環境の中身がわかる形の記法を使えばよい。(実質的な中身が正しければよい。)

```
type expr =
  | CstI of int
  | Var of string
  | Let of string * expr * expr
  | Prim of string * expr * expr ;;
let rec lookup env x = (* omitted *) ;;
let rec eval e (env : (string * int) list) : int =
  match e with
  | CstI i            -> i
  | Var x             -> lookup env x
  | Let(x, erhs, ebody) ->
    let xval = eval erhs env in
    let env1 = (x, xval) :: env in
       eval ebody env1
  | Prim("+", e1, e2) -> (eval e1 env) + (eval e2 env)
  | Prim("*", e1, e2) -> (eval e1 env) * (eval e2 env)
  | Prim _            -> failwith "unknown" ;;
```

1-a. C 言語や OCaml 言語における (2+4) に相当する式を、上記の expr 型の要素として表しなさい。

1-b. 式 e を、上記の eval で評価したい。eval は e と env を引数として取るが、この env として (eval の最初の呼び出しで) 何を与えればよいか答えなさい。

1-c. 次の expr 型の式を、上記の eval で評価するとき、関数 eval は 再帰呼び出しにより何度か呼ばれるはずである。そこで、eval が呼ばれるごとに引数 e と env がどの値を取るかをすべて (実行の順序に従って) 書きなさい。

```
Let("x", CstI(3), Prim("+", Var("x"), CstI(5)))
```

1-d. 前問と同様のことを、次の式に対して書きなさい。

```
Let("x", CstI(7),
    Prim("+",
         Let("y", CstI(9),
             Prim("*", Var("x"), Var("y"))),
         Let("z", CstI(2),
             Prim("+", Var("z"), CstI(10)))))
```

1-e. Simple 言語に print 文を追加した言語では、a+b の形の式 (を Simple の構文に直したもの) の評価で、a から評価するか b から評価するかで、印刷される結果が異なることがある。上記の eval を適宜修正して、「a+b の形の式は a から評価し、a*b の形の式は b から評価する」ようにするためには、どうしたらよいかを述べよ。(OCaml のコードが書ける人は具体的に書けばよい。また、C 言語や Java 言語に似た構文をつかった擬似コードでも構わない。)

問 2. (配点 35 点)

今度は、関数をデータとして扱うことのできる言語 (Simple 言語の拡張) を想定して、以下のコードについて考える。

```
Let("x", CstI(3),
 Letfun("f", "y", Prim("+", Var("x"), Var("y")),
   Let("x", CstI(5),
      Letfun("g", "y",
             Prim("+", Prim("*", Var("x"), Var("y")),
                       Var("y")),
        Call(Var("g"), Call(Var("f"),Var("x")))))))
```

参考までに、これは、以下の OCaml コードと同等のコードである。こちらについて考えてもよい。

```
let x = 3 in
let f y = x + y in
let x = 5 in
let g y = x * y + y in
   g (f x)
```

上記のプログラムを以下のそれぞれの方式で実行したとき、プログラムが返す値を示しなさい．また、それぞれの方式での実行において、環境 (前問における env 引数) がどのように変化するか、実行の順序に従って書きなさい。ただし、関数クロージャ等の表記は、自分で設定した表記でかまわない。

2-a. 静的束縛かつ値呼び

2-b. 静的束縛かつ名前呼び

2-c. 動的束縛かつ値呼び

2-d. 関数呼び出しの方式の 1 つで、必要呼びとは何かを説明しなさい。(その特徴を 2 点以上とりあげ、それぞれ簡潔に説明しなさい。)

問 3. (配点 25 点)

Java 言語で、2 つのクラス ClassA, ClassB を、以下のように定義する。

```
class ClassA {
  public String toString () { return "ClassA";       }
  public String method1 ()  { return "Method1";      }
  ClassA ()                 {}
}
class ClassB extends ClassA {
  public String toString () { return "ClassB";       }
  public String method2 ()          { return "Method2-1"; }
  public String method2 (String s)  { return "Method2-2"; }
  ClassB ()                 {}
}
```

ClassB は ClassA を継承している。toString, method1, method2 はメソッドである。

このとき、以下のプログラムを実行したとする。ただし、いくつかの行はコンパイルエラーを起こす可能性があり、その場合は、他の行を試すときは、その行をコメントにしているものとする。

また、new ClassA() は、ClassA というクラスのオブジェクトを新たに 1 つ作り、x.method1() は、変数 x に格納されたオブジェクトへのメソッド method1 の呼び出し、System.out.println は引数の値を印刷する。

```
class Test1 {
  public static void main(String args[]) {
  ClassA  x;  ClassB  y;
  y = new ClassA();                 // test1
  System.out.println(y.toString()); // test2
  System.out.println(y.method2());  // test3
  y = new ClassB();                 // test4
  System.out.println(y.toString()); // test5
  System.out.println(y.method1());  // test6
  System.out.println(y.method2());  // test7
  System.out.println(y.method2("a"));  // test8
  x = y;                            // test9
  System.out.println(x.toString()); // test10
  System.out.println(x.method1());  // test11
  System.out.println(x.method2());  // test12
} }
```

3-a. 上記の test1 から test12 について、(1) コンパイルエラーか、(2) コンパイルエラーではなく、実行時エラーか、(3) エラーがなく実行すると何かが印刷されるものはその文字列を答えなさい。また、たとえば、test1 がコンパイルエラーの場合、test2 から test3 まではテストできないので、それも (1) に分類せよ。

3-b. 上記の例を参考に、Java における override と overload について簡潔に (1-2 行で) 説明しなさい。

問 4. (配点 15 点)

以下の設問に全て答えなさい。解答の分量の目安は、問題 1 つあたり 5 行程度とする。

4-a. コンパイラが最適化 (高速化) のために取得・活用する「プログラムの静的情報」には、どんなものがあるが、具体例をあげて説明せよ。(A という情報は静的であり、これをこういう風に利用すると、コードが高速になる、という形で 2-3 個の例について述べよ。)

4-b. 静的型付けとは何か、また、静的型付けが、動的型付けに比べて優れている点を (なるべく多く) 述べよ。

4-c. オブジェクト指向言語を特徴付ける 4 つの性質のうち、動的ルックアップ、情報隠蔽、継承とは何かを簡単に説明した上で、なぜそれらがプログラムを書く上で有用な概念であるかを説明せよ。

以上.