Fall, 1998 Name: _____

. My Section Day and Time : _____

Com S 342 — Principles of Programming Languages

Test on *EOPL* Chapters 6.1–3, 7.1

This test has 5 questions and pages numbered 1 through 9.

Reminders

For this test, you can use one (1) page (8.5 by 11 inches, one (1) side, no less than 9pt font) of notes. No photo-reduction is permitted. These notes are to be handed in at the end of the test. Use of other notes or failure to follow these instructions will be considered cheating.

During the test, if you need more space for an answer, use the back of a page. Note when you do that on the front.

This test is timed. We will not grade your test if you try to take more than the time allowed. Therefore, before you begin, please take a moment to look over the entire test so that you can budget your time.

Diagrams should be drawn as we discussed in class. If your diagrams do not follow the conventions used in class, or if they are sloppy or hard to read, you will lose points.

Briefly answer the following questions.

1. (5 points) In class we described how objects are like closures. What part of a closure is like the instance variables of an object?

2. (5 points) What is the array model used by Scheme and Java?

3. (20 points) For each of the following, list:
   (i) the array model, and
   (ii) all of the parameter passing mechanisms that we studied
   that correspond to each of the following groups of domains? (In some cases there may be more than one parameter passing mechanism that goes with each; to recieve full credit, name each one that is applicable.)

   (a)
   ```
   Denoted-Value = Cell(Expressed-Value)
   Expressed-Value = Number + Procedure + Array(Expressed-Value) + Void
   ```

   (b)
   ```
   Denoted-Value = Cell(Storable-Value) + Array(Storable-Value)
                 + Array-Element(Storable-Value)
   Expressed-Value = Number + Procedure + Array(Storable-Value) + Void
   Storable-Value = Number + Procedure
   ```

   (c)
   ```
   Denoted-Value = Cell(Expressed-Value) + Array-Element(Expressed-Value)
   Expressed-Value = Number + Procedure + Array(Expressed-Value) + Void
   ```

   (d)
   ```
   Denoted-Value = Cell(Storable-Value) + Array(Storable-Value)
   Expressed-Value = Number + Procedure + Array(Storable-Value) + Void
   Storable-Value = Number + Procedure
   ```

4. (15 points) Consider the following expression in the defined-langauge.

```
let a = 5;
    lst = emptylist;
    g = 0
in begin
    a := 5;
    g := proc(b,c)
             begin
               b := 7;
               lst := cons(c, emptylist);            %% this changes lst
               c := +(a,b);
               a := +(b,c);
               lst := cons(a, cons(b, cons(c, lst))) %% this builds on lst
             end;
     g(a, +(a,0));
     writeln(a);
     writeln(lst)
   end
```

Fill in the following table with the final values printed for `a` and `lst` by the above expression in each of the given parameter mechanisms. No diagrams are necessary for this problem.

(When doing call-by-value-result, copy results back in left-to-right order. Use the indirect array model, if necessary. If need be, you may use "?" to represent an undefined (i.e., unspecified) value.)

calling mechanism	value printed for `a`	value printed for `lst`
call-by-value		
call-by-reference		
call-by-value-result		

5. (90 points) This is a problem about parameter passing mechanisms and array models. Consider the following expression.

```
letarray a[2]; b[2]
in begin
     a[0] := 1; a[1] := 0; b[0] := 0; b[1] := 1;
     let i = 0; j = 1
     in let f = proc(v, w, m, n, v0, w1)
                  begin
                    i := v[n]; w := v; a[m] := w[1]; j := v0;
                    b[n] := 5; v0 := 10; w1 := v[i];
                    %%% draw a picture for this point
                    +(v0, w1)
                  end
        in let r = f(a, b, i, j, a[i], b[j])
           in list(a[0], a[1], b[0], b[1], i, j, r)
   end
```

For each of the following combinations of parameter passing mechanism and array model: (i) draw a picture of the execution (as discussed in class) for the point noted by the comment, and (ii) give the result of the expression. (If need be, you may use "?" to represent an undefined (i.e., unspecified) value.) The combinations you are to do are as follows (there are more on the following pages).

(a) Call-by-value with the indirect model.

Here is another copy of the expression, for your convenience.

```
letarray a[2]; b[2]
in begin
     a[0] := 1; a[1] := 0; b[0] := 0; b[1] := 1;
     let i = 0; j = 1
     in let f = proc(v, w, m, n, v0, w1)
                  begin
                    i := v[n]; w := v; a[m] := w[1]; j := v0;
                    b[n] := 5; v0 := 10; w1 := v[i];
                    %%% draw a picture for this point
                    +(v0, w1)
                  end
        in let r = f(a, b, i, j, a[i], b[j])
           in list(a[0], a[1], b[0], b[1], i, j, r)
   end
```

(b) Call-by-value with the direct model.

Here is another copy of the expression, for your convenience.

```
letarray a[2]; b[2]
in begin
     a[0] := 1; a[1] := 0; b[0] := 0; b[1] := 1;
     let i = 0; j = 1
     in let f = proc(v, w, m, n, v0, w1)
                  begin
                    i := v[n]; w := v; a[m] := w[1]; j := v0;
                    b[n] := 5; v0 := 10; w1 := v[i];
                    %%% draw a picture for this point
                    +(v0, w1)
                  end
        in let r = f(a, b, i, j, a[i], b[j])
           in list(a[0], a[1], b[0], b[1], i, j, r)
   end
```

(c) Call-by-reference with the indirect model

Here is another copy of the expression, for your convenience.

```
letarray a[2]; b[2]
in begin
     a[0] := 1; a[1] := 0; b[0] := 0; b[1] := 1;
     let i = 0; j = 1
     in let f = proc(v, w, m, n, v0, w1)
                  begin
                    i := v[n]; w := v; a[m] := w[1]; j := v0;
                    b[n] := 5; v0 := 10; w1 := v[i];
                    %%% draw a picture for this point
                    +(v0, w1)
                  end
        in let r = f(a, b, i, j, a[i], b[j])
           in list(a[0], a[1], b[0], b[1], i, j, r)
   end
```

(d) Call-by-reference with the direct model

Here is another copy of the expression, for your convenience.

```
letarray a[2]; b[2]
in begin
     a[0] := 1; a[1] := 0; b[0] := 0; b[1] := 1;
     let i = 0; j = 1
     in let f = proc(v, w, m, n, v0, w1)
                  begin
                    i := v[n]; w := v; a[m] := w[1]; j := v0;
                    b[n] := 5; v0 := 10; w1 := v[i];
                    %%% draw a picture for this point
                    +(v0, w1)
                  end
        in let r = f(a, b, i, j, a[i], b[j])
           in list(a[0], a[1], b[0], b[1], i, j, r)
   end
```

(e) Call-by-value-result with the indirect model (copy the results back in left-to-right order)

Here is another copy of the expression, for your convenience.

```
letarray a[2]; b[2]
in begin
     a[0] := 1; a[1] := 0; b[0] := 0; b[1] := 1;
     let i = 0; j = 1
     in let f = proc(v, w, m, n, v0, w1)
                  begin
                    i := v[n]; w := v; a[m] := w[1]; j := v0;
                    b[n] := 5; v0 := 10; w1 := v[i];
                    %%% draw a picture for this point
                    +(v0, w1)
                  end
        in let r = f(a, b, i, j, a[i], b[j])
           in list(a[0], a[1], b[0], b[1], i, j, r)
   end
```

(f) Call-by-value-result with the direct model (copy the results back in left-to-right order)