

---

**Final Exam (40 pts)**

## 1 Interpreter for Object-Oriented Language (8 + 8 pts)

Consider the interpreter for OOPL given in the files **5-3.scm** and **5-4-4.scm**. We explore introducing *static fields* into class definitions. Static fields associate some state with a class; all the instances of a class share this state. Static fields can appear in a method body. For example, one might write the following program and obtain as the result, the list (01):

```
class c1
  static next_serial_number
  field my_serial_number
  method get_serial_number () my_serial_number
  method initialize ()
    begin
      set my_serial_number = next_serial_number;
      set next_serial_number = add1(next_serial_number);
    end
let o1 = new c1()
    o2 = new c1()
in list (send o1 get_serial_number(),
        send o2 get_serial_number())
```

Explain *all* the modifications to the interpreter code to incorporate the *static fields* shown in the grammar rule for class definition below, *informally* and *clearly* first, and formally in code later. Initialize each static field to 0, by default.

```
(class-decl
  ("class" identifier
   "extends" identifier
   (arbno "static" identifier)
   (arbno "field" identifier)
   (arbno method-decl)
  )
  a-class-decl)
```

Specifically, locate all the changes to the original interpreter code on the accompanying interpreter code handout and/or locate changes/insertions using file name, page number, and line numbers.

## 2 Attribute Grammars (8 pts)

Consider the following grammar for the base-3 numerals  $\mathcal{T}$ .

$$T ::= 0 \mid 1 \mid 2 \mid T T$$

Write an attribute grammar to determine the numeric value of these numerals. For instance,  $Value(201) = 19$ ,  $Value(1010) = 30$ , etc.

## 3 Axiomatic Semantics (5 + 3 pts)

Determine the following weakest preconditions. (Assume all variables are of *integer* type.)

$wp(\{ \text{if odd}(j) \text{ then } k := k*i; \quad j := j-1 \text{ else } i := i*i; \quad j := j \text{ div } 2 \},$   
 $\quad [ k*(i^j) = n ] ) = ?$

$wp(\{ \text{while } i > 0 \text{ do } i := i - 5; \}, \quad i = 0) = ?$

## 4 Algebraic Specification (8 pts)

Give an algebraic specification of the ADT **Int\_Bag** that supports the following operations: **empty**, **insert**, **isEmpty**, **count**, **union** and **intersection**. (Recall that a bag is a homogeneous collection of values where duplication is significant, but the order of values is not.) Informally,

- **empty**: Yields the empty bag.
- **insert**: Takes an integer and a bag as input, and yields the bag resulting from introducing one occurrence of the integer into the bag.
- **isEmpty**: Checks if the bag is empty.
- **count**: Takes an integer and a bag as input, and yields the number of occurrences of the integer in the bag.
- **union**: Takes two bags as input, and yields the bag containing integers that belong to either bags. (That is,  $\text{union}([1,1,2], [2,3]) = [1,1,2,2,3]$ , etc.)
- **intersection**: Takes two bags as input, and yields a bag containing integers that belong to both the bags. (That is,  $\text{intersection}([1,1,2], [1,1,1,2,3]) = [1,1,2]$ ,  $\text{intersection}([1,2,2,2,3],[1,1,2,4]) = [1,2]$ ,  $\text{intersection}([1],[2]) = []$ , etc)