

---

**Midterm (30 pts)**

## 1 Recursive Definition (10 pts)

Define a binary predicate `similarForm?` that takes two expressions specified by the following grammar and returns true iff the expressions are structurally similar.

```
<expression> ::= <symbol>
                | <number>
                | ()
                | (<expression> . <expression>)
```

Two expressions are *structurally similar* if both are symbols, or numbers, or empty lists, or contain, recursively, structurally similar pairs.

```
(similarForm? 'x 'x) returns #t
(similarForm? '() '(x)) returns #f
(similarForm? '(256 x ()) '(512 y ())) returns #t
(similarForm? '(((a) b (2)) '((x)) y (3))) returns #t
(similarForm? '(((a) b (2)) '(x y (3))) returns #f
```

## 2 Recursive Definition (6 pts)

Consider the following transcript of a session with the Scheme interpreter.

```
(define (my_if my_test my_then my_else)
  (if my_test my_then my_else)
)

(my_if (zero? 2) 'yes 'no)
;no

(define (fact n)
  (my_if (zero? n) 1 (* n (fact (- n 1)) )
  )
)

(fact 2)
;--InFiNiTe LoOp--
```

Explain why the interpreter goes into an infinite loop for the seemingly innocuous call `(fact 2)`.

### 3 ADT Specification (4 + 10 pts)

A multiset is a homogeneous collection of values. Similarly to sets, the order of elements is not significant, but in contrast with sets, the multiplicity of an element is significant. You are required to specify the ADT **Char\_MS** that supports the following operations: **create**, **insert**, **isEmpty?**, **howMany**, **either**, and **common**. Informally,

- **create**: Yields the empty multiset.
- **insert**: Takes a char and a multiset as input, and yields the multiset resulting from introducing the char into the multiset.
- **isEmpty?**: Checks to see if a multiset is empty.
- **howMany**: Takes a char and a multiset as input, and yields the number of occurrences of the char in the multiset.
- **either**: Takes two multisets as input, and yields the multiset containing chars in either. (That is, **either**(['a', 'b', 'a'], ['b', 'c']) = ['a', 'b', 'a', 'b', 'c'], etc)
- **common**: Takes two multisets as input, and yields a multiset containing chars that are common to both. (That is, **common**(['a', 'b'], ['a', 'a', 'b', 'c']) = ['a', 'b'], **common**(['a', 'b', 'b', 'b', 'c', 'c'], ['a', 'a', 'c', 'c', 'd']) = ['a', 'c', 'c'], etc)

1. Specify the signatures of the aforementioned operations on ADT **Char\_MS**.
2. Give an algebraic specification of the ADT **Char\_MS**. (*Hint*: Use **howMany** to define **common**.)