CS 603 Exam 1 **Spring 2008** **Name ______________**

1. Show that this grammar is ambiguous by drawing two parse trees that generate the same string.

   S → X | Y
   X → aXb | aa
   Y → aaYb | b

2. Draw a diagram that illustrates the internal representation of this S-expression:
   (a (b c) d ( ) (e ((f g) h (i) j)))

3. Write an unambiguous context-free grammar that generates S-expressions that contain only the symbol x. For example, (x (x x) x ( ) (x ((x x) x (x) x))). Your production rules may use concatenation and |, but not other extended BNF operations.

4. Suppose we give Impcore a new primitive function && which implements binary short-circuit AND. So the expression (&& x y) in Impcore should be equivalent to x&&y in C or C++ or Java. Define the && function by completing these two Impcore-style natural operational semantics rules.

$\phi(f) = \text{PRIMITIVE}(\&\&)$ ______________________________ $\langle \text{APPLY}(f,e_1,e_2), \xi, \phi, \rho \rangle \Downarrow$	$\phi(f) = \text{PRIMITIVE}(\&\&)$ ______________________________ $\langle \text{APPLY}(f,e_1,e_2), \xi, \phi, \rho \rangle \Downarrow$

5. Suppose we give μScheme a new primitive function || which implements binary short-circuit OR. So the expression (|| x y) in μScheme should be equivalent to x||y in C or C++ or Java. Define the || function by completing these two μScheme-style natural operational semantics rules.

$\langle e, \rho, \sigma \rangle \Downarrow \langle \text{PRIMITIVE}(\mid\mid), \sigma_1 \rangle$ ______________________________ $\langle \text{APPLY}(e,e_1,e_2), \rho, \sigma \rangle \Downarrow$	$\langle e, \rho, \sigma \rangle \Downarrow \langle \text{PRIMITIVE}(\mid\mid), \sigma_1 \rangle$ ______________________________ $\langle \text{APPLY}(e,e_1,e_2), \rho, \sigma \rangle \Downarrow$

6. Write a μScheme function (diagonal M) where M is a square matrix stored as a list of row lists. It should return a list of the main diagonal elements of M. Example: (diagonal '((a b c d) (e f g h) (i j k l) (m n o p))) returns (a f k p).

$$\begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix}$$

7. Write a μScheme function (scan op id L) where op is a binary function, id is the identity value, and L is a list. It should return a list of the values obtained by folding op across each possible prefix of L. Example: (scan + 0 '(2 3 5 7 11)) returns (0 2 2+3 2+3+5 2+3+5+7 2+3+5+7+11) = (0 2 5 10 17 28). [You may assume that op is an associative operation.]

8. The Impcore function below shows an inefficient way to compute Fibonacci numbers. Note that (fib 0) = (fib 1) = 1, (fib 2) = 2, (fib 3) = 3, (fib 4) = 5, (fib 5) = 8, etc. Write an equivalent function so that (fib n) runs in O(n) time. Hint: use a helper function that "remembers" the previous two values.

```
(define fib (n) (if (<= n 1) 1
        (+ (fib (- n 2)) (fib (- n 1)))))
```

9. Complete the μScheme function Stack so that the print statements in the client code below will produce the given output.

```
(define Stack ( )
  (let ((L '( )))
    (lambda (m)
```

```
(val A (Stack))
(val B (Stack))
(val k 0)
(while (< k 10) (begin
        ((A 'push) k)
        ((B 'push) (+ k 1))
        (set k (+ k 2))
))
(while (not (A 'isEmpty)) (begin
        (print (A 'top))
        (A 'pop)
))
(while (not (B 'isEmpty)) (begin
        (print (B 'top))
        (B 'pop))
)
```

```
8
6
4
2
0

9
7
5
3
1
```

10. Using either C or C++ or Java, write a definition for a Stack abstract data type represented as a linked list of ints. Provide these operations with the same functionality as in the preceding problem: isEmpty, top, pop, push.