

CPTS 355 Midterm, Fall 2004

NAME _____

Directions: Please answer all of the questions. You may use 1 page of notes (front and back). You may **not** use a computer, PDA, calculator or phone during the test.

There are **10** major problems on **8** pages (4 sheets, front and back) totaling 90 points. Make sure that you have a complete exam **now**.

1) **[24 pts] Directions:** State whether each statement is TRUE or FALSE by writing TRUE or FALSE in the space provided. For each false statement explain why it is false.
(2 points for each subproblem)

- a) _____ (+ 2 3 4 5) is a legal Scheme expression.
- b) _____ [1, 2, [3,4]] is a legal ML list value.
- c) _____ Static and dynamic scope rules always give the same results.
- d) _____ Static typing and static scoping are two different names for the same concept.
- e) _____ PostScript uses postfix notation for expressions.
- f) _____ PostScript and Scheme are both type-safe languages.
- g) _____ Scheme has dynamic typing.
- h) _____ ML is both statically typed and statically scoped.
- i) _____ Understanding precedence of arithmetic operators is important for writing correct Scheme programs.
- j) _____ An ML compiler is required to issue an error message whenever a function will not terminate.
- k) _____ (1 2 (3 4)) is a legal Scheme list value.
- l) _____ If a declaration, D, is in the referencing environment of a program location, L, then L is in the scope of D.

Directions: The following are short answer questions. Write your answer in the space provided.

2) [4 points] What is the *type* of the ML function created by the declaration

```
fun f [] = 1
  | f [x] = 1
  | f (x::y::xs) = 1 + (f (y::xs)) + (f xs);
```

3) [8 points] What kind of ML type corresponds most closely to the following kinds of C types? Give an example of each kind of ML type.

a) structs?

b) unions?

4) [10 points] Determine the weakest precondition for the following program fragments, using the assignment rule. Show your work.

a) $z = 3; \quad y = z * 5 + z; \quad \{y == 18\}$

b) $z = 3; \quad y = (x - 5) + z; \quad \{y == 19\}$

5)[10 points] Explain step by step how the postscript interpreter would evaluate the following program. Show the contents of the expression stack and the dictionary stack immediately *after* each `def` is performed and when the program finishes. Assume that program execution begins with the expression stack empty and the dictionary stack containing a single dictionary.

```
/x 3 def
/f { 2 dict begin /y exch def
    /x 4 def
    x y add
    end
  } def
x f
```

6) [12 points] Draw a derivation tree for the sentence

$$3 + 4 * 5 + 2$$

using the grammar given below. Assume that $\langle e \rangle$ is the start symbol.

$$\begin{aligned}\langle e \rangle &\rightarrow \langle t \rangle * \langle e \rangle \mid \langle t \rangle \\ \langle t \rangle &\rightarrow \langle t \rangle + \langle f \rangle \mid \langle f \rangle \\ \langle f \rangle &\rightarrow 3 \mid 4 \mid 5 \mid 2\end{aligned}$$

What are the non-terminals of this grammar? _____

Which operator, + or *, has higher precedence? _____

What is the value of this expression when computed according to your parse tree? _____

In this grammar + is _____ associative.

7) [8 points] In the following program, assume that $n \geq 1$.

```
i = 1;
p = 1;
while (i < n) {
    i = i + 1;
    p = p * i;
}
```

$((i \leq n) \text{ and } (p == i!))$ is an invariant of this loop. Use this fact to show that when the loop terminates $p == n!$

9) **[10 pts]** In project 2 you defined a `zip` function that turned two lists into a list of pairs. For this problem, write a recursive ML function that is the inverse of `zip`: it takes as argument a list of pairs and returns a pair of lists. For example, `unzip [(1, 2), (3, 4), (5, 6)]` should return `([1, 3, 5], [2, 4, 6])`. You may use any built-in ML functions you like except for `unzip` itself.

10) [4 **points**] What programming language feature or mechanism that *you* have encountered or heard about are *you* most curious about? Make a case for including discussion of it in the second half of the course.