

## Midterm Test

July 6, 2000

---

**Duration:** 50 minutes

**Aids allowed:** None

**Weight:** 15% of your course grade

This exam contains a total of 7 pages (including this one). Write your answers clearly in the spaces provided. Use the back pages for your rough work.

**Surname:** \_\_\_\_\_

**First name:** \_\_\_\_\_

**Student #:** \_\_\_\_\_

**Tutor (circle one):**

Diana Inkpen  
L0101 (A-Le)

George Giakkoupis  
L0101 (Li-Z)

Anastasia Bezerianos  
L5101 (A-Li)

Sean Thompson  
L5101 (Lo-Z)

# 0: \_\_\_\_\_/ 2

# 1: \_\_\_\_\_/ 6

# 2: \_\_\_\_\_/ 6

# 3: \_\_\_\_\_/ 6

# 4: \_\_\_\_\_/ 6

# 5: \_\_\_\_\_/ 6

# 6: \_\_\_\_\_/12

# 7: \_\_\_\_\_/10

# 8: \_\_\_\_\_/10

TOTAL: \_\_\_\_\_/64

Good Luck!

**Question 0.** [2 MARKS]

Write your name (or your initials if your name is long) and student number legibly at the top of every page of this test.

**Question 1.** Parameter Passing [6 MARKS]

Considering static scoping and three of the four possible parameter-passing methods (call by value, call by reference, and call by name), what will be the value of **x**, and **y** at the end of the following program?

```
int x, y;
procedure P(int a, int b, int c) {
  int x;
  x = 3;
  b = b+x+6;
  c = c+a;
}

x = 5;
y = 10;
P(y,x,x);
```

	<b>x</b>	<b>y</b>
Call by value		
Call by reference		
Call by name		

**Question 2.** List Representation [6 MARKS]

Draw a tree showing Scheme's internal representation for `lst` after the following expression is evaluated:

```
(define lst '((1 2) () (3 . 4)))
```

**Question 3.** Unification [6 MARKS]

What is the result of unifying the following expression? Write the answer in the appropriate box, or "cannot" if the two expressions cannot be unified.

$[a, b, X]$	$[X, Y, [a]]$	
$[X, b   X]$	$[Y, b, c, d]$	
$p(a, X)$	$p(Y, [c, d])$	

**Question 4.** `sum-all` [6 MARKS]

Write a Scheme function (`sum-all L`) that finds the sum of all the numbers in a list that may contain nested sublists of numbers. For example,

```
1 ]=> (sum-all '())
```

```
;Value: 0
```

```
1 ]=> (sum-all 7)
```

```
;Value: 7
```

```
1 ]=> (sum-all '((5 6) 9 (7 (7 8))))
```

```
;Value: 42
```

```
1 ]=> (sum-all '(5 6 9 7 7 8))
```

```
;Value: 42
```

```
(define (sum-all L)
```

**Question 5.** Scheme Interpretation [6 MARKS]**Part (a)** [3 MARKS]

Given the following Scheme function `f`:

```
(define f
  (lambda (item ls)
    (cond
      ((null? ls) '())
      ((equal? (car ls) item) (cdr ls))
      (else (cons (car ls) (f item (cdr ls)))))
    )
  )
)
```

What is the result of the following calls:

1. `(f 'a '())` \_\_\_\_\_
2. `(f '4 '(2 3 4 3 9 4))` \_\_\_\_\_
3. `(f 'ian '(joe jack ian jack ian ian))` \_\_\_\_\_

**Part (b)** [3 MARKS]

Given the following Scheme function `g`:

```
(define (g L)
  (cond
    ((number? L) '())
    ((number? (car L)) (cdr L))
    (else (cons (g (car L)) (cdr L))))
  )
)
```

What is the result of the following calls:

1. `(g '((9 8 9 (7) 4)))` \_\_\_\_\_
2. `(g '((9 8 (5 6)) 9 (7) 4))` \_\_\_\_\_
3. `(g '(((9 5) 9) 7))` \_\_\_\_\_

Note that `number?` is a built-in predicate that returns true if its argument is a number and false otherwise.

**Question 6.** Mystery Function [12 MARKS]

Consider the following mystery function:

```
(define (foo M L)
  (cond ((null? L) '())
        ((number? L) (M L))
        (else (cons (foo M (car L)) (foo M (cdr L))))))
)
```

(a) [4 MARKS] For each of the following expressions, indicate if the expression is legal and if so, what it is going to return. Note that `zero?` is a built-in predicate that tests if its argument is 0.

1. `(foo zero? '(1 2 3 0))`
2. `(foo (lambda (x y) (+ x y)) '(1 2 3 4))`
3. `(foo (lambda (x) (+ 1 x)) '(1 (2 (3 4) 5) 6))`

(b) [2 MARKS] What does `foo` do, in general?

(c) [6 MARKS] Rewrite `foo` using `map`.

**Question 7.** Prolog [10 MARKS]

Consider the following Prolog facts and rules:

```
male(john).
male(philip).
female(suzanne).
female(janette).
parent(suzanne, john).
parent(suzanne, janette).
parent(philip, john).
parent(philip, janette).
father(X,Y) :- male(X), parent(X,Y).
sibling(X,Y) :- parent(Z,X), parent(Z,Y).
```

- (a) [2 MARKS] Write a Prolog rule `uncle(X,Y)` which holds if `X` is an uncle of `Y`.
- (b) [2 MARKS] Express the `sibling` rule in logic.
- (c) [2 MARKS] What will be Prolog's first answer to the query `father(X,Y)`?
- (d) [2 MARKS] What will be Prolog's first answer to the query `sibling(john,X)`?
- (e) [2 MARKS] What will be Prolog's first answer to the query `sibling(X,john)`?

**Question 8.** Short Questions [10 MARKS]

Answer the following short questions.

(a) [2 MARKS] List three language evaluation criteria that affect readability.

(b) [2 MARKS] What is pseudo-compilation?

(c) [2 MARKS] What characterizes functional programming?

(d) [2 MARKS] What is lexical scoping?

(e) [2 MARKS] What is a Horn clause?

Total Marks = 64