YOUR NAME PLEASE:

Computer Science 201a
Practice Midterm
October 13, 2000

Open book and open notes. Show ALL work you want graded on the test itself, including the backs of pages as necessary.

For problems that do not ask you to justify the answer, an answer alone is sufficient. However, if the answer is wrong and no derivation or supporting reasoning is given, there will be no partial credit.

GOOD LUCK!

<table>
<thead>
<tr>
<th>problem</th>
<th>points</th>
<th>actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>total</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>
1. Define a procedure (bit? x) that returns 
#t if its argument is 0 or 1, 
#f otherwise.

Examples:

(bit? 0) => #t, 
(bit? 1) => #t, 
(bit? 2) => #f, 
(bit? '(a b)) => #f
2. Define a procedure (cb 1s) that takes a list 1s and returns the number of top-level items in 1s that are 0’s or 1’s. Your procedure should do flat recursion and generate a recursive process. You may use the predicate bit? from problem 1.

Example: (cb '(0 0 1 ((0 a) c 1) 1 0)) => 5
3. Define a procedure \( \text{cb-it ls count} \)
that is an iterative version of \( \text{cb} \) from problem 2.
That is, \((\text{cb-it ls 0})\) should return the same value as \((\text{cb ls})\).
4. Define a procedure (cb-deep ls)
that returns the number of 0’s or 1’s
at any level of the list ls.

Example: (cb-deep '(0 0 1 ((0 a) c 1) 1 0)) => 7
5. Recall the definition of map:

(define map
  (lambda (proc ls)
    (cond
      ((null? ls) '())
      (else (cons (proc (car ls)) (map proc (cdr ls)))))))

(a) What is the value of
(map (lambda (x) (* x x)) '(3 5 7))?

(b) What is the value of
(map (lambda (x) (cons x '())) '(a (b c) (d) e))?
6. Consider the following Scheme expression:

(let ((x 5) (y 2))
  (let ((x (* y y)) (y (+ x x)))
    (list x y)))

(a) Rewrite the above expression using lambdas and no lets.

(b) What is the value of the let expression given above?

(c) Draw a picture of the environments (global and local) when the (list x y) expression is evaluated as part of the evaluation of the let expression above. (Don’t forget pointers from one environment to another.)
7. Consider the following RAM program, P.

1. lda 1
2. sub 2
3. jnz 7
4. sti 1
5. sta 1
6. jmp 2
7. hlt

Suppose we set up memory with 0 in all words except

1: 6
2: 3
3: 1

Simulate the program P started at instruction 1 until it halts,
and show the value of PC and AC
before the execution of each instruction.
Also, describe the final contents of memory.
8. Construct a Turing machine $M$ using the alphabet $b$, $0$, and $1$, where $b$ is the blank symbol, such that if $M$ is started with its head on the rightmost symbol of a binary number representing the positive integer $x$, then it eventually halts with its head on the rightmost symbol of a binary number representing the integer $(x+1)$.

Example: (initial tape, final tape, head position shown by ^)

```
b b b 1 0 1 1 b b b
------------^-----

b b b 1 1 0 0 b b b
------------^-----
```
9. Consider the finite state machine, F, described as follows:

states: q0, q1, q2
symbols: a, b
initial state: q0
accepting states: q1, q2
transition function:

(q0,a) -> q1  (q0,b) -> q0
(q1,a) -> q2  (q1,b) -> q1
(q2,a) -> q0  (q2,b) -> q1

(a) Draw a (circles and arrows) diagram of F, indicating states, transitions, start state, and accepting states.

(b) Does F accept or reject each of the following strings? Show the sequence of states F passes through for each string

Example: (0) aab  accept  q0->q1->q2->q1

(1) a

(2) abaa

(3) bbb

(4) aaabba

(5) ababab
10. (Dreaded decidabilities)
Are the following problems decidable or undecidable?
Give a brief justification of your answer.

(a)
Input: a string \( w \) and a regular expression \( E \).
Output: 1 if \( E \) matches \( w \), 0 if \( E \) doesn’t match \( w \).

(b)
Input: a Turing machine \( T \) and a tape \( t \).
Output: 1 if \( T \) doesn’t halt on \( T \), 0 if \( T \) does halt on \( t \).

(c)
Input: a Scheme procedure \( \text{proc} \).
Output: 1 if \( (\text{proc } 0) \) and \( (\text{proc } 1) \) both halt, 0 otherwise.