INSTRUCTIONS

- You have 10 minutes to complete this quiz.
- The exam is closed book, closed notes, closed computer, closed calculator.
- The final score for this quiz will be assigned based on **effort** rather than correctness.
- Mark your answers **on the exam itself**. We will **not** grade answers written on scratch paper.
- For multiple choice questions,
  - □ means mark all options that apply
  - ○ means mark a single choice

<table>
<thead>
<tr>
<th>Last name</th>
<th>First name</th>
<th>Student ID number</th>
<th>CalCentral email (@berkeley.edu)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Alex Stennet</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Kelly Chen</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Angela Kwon</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Michael Gibbes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ashley Chien</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Michelle Hwang</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Joyce Luong</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Mitas Ray</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Karthik Bharathala</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Rocky Duan</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Kavi Gupta</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Samantha Wong</td>
</tr>
</tbody>
</table>

Teaching Assistant

Name of the person to your left

Name of the person to your right

*All the work on this exam is my own.* (please sign)
1. (3 points) This is a Deep Problem

Stan wants to write \texttt{deep-squares} which takes a deep list of numbers and returns a list with each value squared.

\begin{verbatim}
(define (deep-squares lol)
  (cond ((null? lol) ()
    ((list? (car lol))
      (cons (deep-squares (car lol))
        (deep-squares (cdr lol)))
    (else (cons (square (car lol))
        (deep-squares (cdr lol)))))
)
\end{verbatim}

For which of the following inputs will \texttt{deep-squares} not work as intended?

(a) \texttt{(deep-squares '())} \hspace{1cm} \textcolor{red}{\bullet} \hspace{0.5cm} Works \hspace{0.5cm} \textcolor{green}{\circ} \hspace{0.5cm} Broken
(b) \texttt{(deep-squares '(1 2 3 4))} \hspace{1cm} \textcolor{red}{\bullet} \hspace{0.5cm} Works \hspace{0.5cm} \textcolor{green}{\circ} \hspace{0.5cm} Broken
(c) \texttt{(deep-squares '(1 2 3 4 5))} \hspace{1cm} \textcolor{green}{\circ} \hspace{0.5cm} Works \hspace{0.5cm} \textcolor{red}{\bullet} \hspace{0.5cm} Broken

Which line number contains the bug? \hspace{1cm} \textcolor{green}{\circ} \hspace{0.5cm} 1 \hspace{0.5cm} \textcolor{green}{\circ} \hspace{0.5cm} 2 \hspace{0.5cm} \textcolor{green}{\circ} \hspace{0.5cm} 3 \hspace{0.5cm} \textcolor{red}{\bullet} \hspace{0.5cm} 4 \hspace{0.5cm} \textcolor{green}{\circ} \hspace{0.5cm} 5 \hspace{0.5cm} \textcolor{green}{\circ} \hspace{0.5cm} 6 \hspace{0.5cm} \textcolor{green}{\circ} \hspace{0.5cm} 7

2. (2 points) ... That Factors Into Your Learning

Implement the \texttt{factors} procedure in Scheme, which takes an integer \textit{n} that is greater than 1 and returns a list of all of the factors of \textit{n} from 1 to \textit{n} - 1 in \textit{increasing order}.

\textbf{You may only use the lines provided. You may not need to fill all the lines.}

\textit{Hint:} The built-in \texttt{modulo} procedure returns the remainder when dividing one number by the other.

\begin{verbatim}
scm> (modulo 5 3)
2
scm> (modulo 14 2)
0

(define (factors n)
  (define (factors-helper i n)
    (if (= i n)
      nil

      (if (= (modulo n i) 0)
        (cons i (factors-helper (+ i 1) n))

        (factors-helper (+ i 1) n)
      )
    )
  )
  (factors-helper 1 n)
)

scm> (factors 6)
(1 2 3)
scm> (factors 7)
(1)
scm> (factors 28)
(1 2 4 7 14)
\end{verbatim}