CS3:
Introduction to Symbolic Programming

Lecture 11:
Tree Recursion, beginning lists, and Midterm 2

Spring 2007
Nate Titterton
nate@berkeley.edu
# Schedule

<table>
<thead>
<tr>
<th></th>
<th>Date</th>
<th>Lecture:</th>
<th>Lab:</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>April 2-6</td>
<td>Midterm review, tree recursion</td>
<td>Lists, tree-recursion</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Miniproject #3 due Tuesday</em></td>
</tr>
<tr>
<td>12</td>
<td>April 9-13</td>
<td>(5-7 pm): <em>Midterm #2</em> -- 145 Dwinelle</td>
<td>Advanced list processing</td>
</tr>
<tr>
<td>13</td>
<td>April 16-20</td>
<td>CS3 Projects, Lists</td>
<td>Begin work on CS3 Big Project</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Non-functional programming</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reading: <em>Simply Scheme</em>, chapter 20</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>April 23-27</td>
<td>Advanced lists, project review</td>
<td>Work on projects</td>
</tr>
<tr>
<td>15</td>
<td>Apr 30-May 4</td>
<td>CS at Berkeley (guest lecture)</td>
<td>Finish projects (due end of week)</td>
</tr>
<tr>
<td>16</td>
<td>May 7</td>
<td>Exam review</td>
<td><em>no more labs!</em></td>
</tr>
</tbody>
</table>
Any questions about the miniproject?
Midterm 2
Announcements

• Midterm 2 is coming…
  - Next week, 80 minutes (5:10-6:30).
  - Room 145 Dwinelle
  - Open book, open notes, etc.
  - Check for practice exams and solution on the course portal and in the reader.

• Midterm 2 review session
  - Sunday, Apr. 8, 4-6pm in 430 Soda.
What does midterm #2 cover?

- Advanced recursion (accumulating, multiple arguments, etc.).
- Tree-recursion (from this week)
- All of higher order functions
- Those "big" homeworks (bowling, compress, and occurs-in)
- Elections and number-name miniproject
- Reading and programs:
  - Change making, Roman numerals
  - Difference between dates #3 (HOF),
  - tic-tac-toe
- SS chapters 14, 15, 7, 8, 9, 10
- Everything before the first Midterm (although, this won't be the focus of a question)
Tree recursion

(coming this week)
What will happen?

• What will `countem` return for n=1, 2, …?

```
(define (countem n)
  (if (= n 0)
      '()
      (se (countem (- n 1))
          n
          (countem (- n 1))))
```


Tree recursion

A recursive technique in which more than one recursive call is made within a recursive case.
## Pascal's triangle

<table>
<thead>
<tr>
<th>rows (R)</th>
<th>columns (C)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Pascal’s Triangle

- How many ways can you choose C things from R choices?
- Coefficients of the \((x+y)^R\): look in row R
- etc.
(define (pascal C R)
  (cond
    ((= C 0) 1) ; base case
    ((= C R) 1) ; base case
    (else ; tree recurse
      (+ (pascal C (- R 1))
         (pascal (- C 1) (- R 1)))
    )))
Problems
• Write a procedure to generate the possible binary numbers given \( n \) bits.

(binary 1) \( \rightarrow \) (0 1)
(binary 2) \( \rightarrow \) (00 01 10 11)
(binary 3) \( \rightarrow \) (000 001 010 011 100 101 110 111)
## Schedule

<table>
<thead>
<tr>
<th>Date</th>
<th>Activity Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apr 2-6</td>
<td>Lecture: Midterm review, tree recursion</td>
</tr>
<tr>
<td></td>
<td>Lab: Lists, tree-recursion</td>
</tr>
<tr>
<td></td>
<td><strong>Miniproject #3 due Tuesday</strong></td>
</tr>
<tr>
<td>Apr 9-13</td>
<td>Lecture (5-7 pm): <em>Midterm #2</em> -- 145 Dwinelle</td>
</tr>
<tr>
<td></td>
<td>Lab: Advanced list processing</td>
</tr>
<tr>
<td>Apr 16-20</td>
<td>Lecture: CS3 Projects, Lists</td>
</tr>
<tr>
<td></td>
<td>Lab: Begin work on CS3 Big Project</td>
</tr>
<tr>
<td></td>
<td>Non-functional programming</td>
</tr>
<tr>
<td></td>
<td><strong>Reading: Simply Scheme, chapter 20</strong></td>
</tr>
<tr>
<td>Apr 23-27</td>
<td>Lecture: Advanced lists, project review</td>
</tr>
<tr>
<td></td>
<td>Lab: Work on projects</td>
</tr>
<tr>
<td>Apr 30-May 4</td>
<td>Lecture: CS at Berkeley (guest lecture)</td>
</tr>
<tr>
<td></td>
<td>Lab: Finish projects (due end of week)</td>
</tr>
<tr>
<td>May 7</td>
<td>Lecture: Exam review</td>
</tr>
<tr>
<td></td>
<td><em>no more labs!</em></td>
</tr>
</tbody>
</table>
Any questions about the miniproject?
Midterm 2

Click to add text
Announcements

• Midterm 2 is coming…
  - Next week, 80 minutes (5:10-6:30).
  - Room 145 Dwinelle
  - Open book, open notes, etc.
  - Check for practice exams and solution on the course portal and in the reader.

• Midterm 2 review session
  - Sunday, Apr. 8, 4-6pm in 430 Soda.
What does midterm #2 cover?

• Advanced recursion (accumulating, multiple arguments, etc.).
• Tree-recursion (from this week)
• All of higher order functions
• Those "big" homeworks (bowling, compress, and occurs-in)
• Elections and number-name miniproject
• Reading and programs:
  - Change making, Roman numerals
  - Difference between dates #3 (HOF),
  - tic-tac-toe
• SS chapters 14, 15, 7, 8, 9, 10
• Everything before the first Midterm (although, this won't be the focus of a question)
Tree recursion

(coming this week)
What will happen?

• What will \texttt{countem} return for \texttt{n=1, 2, ...}?

\begin{verbatim}
(define (countem n)
  (if (= n 0)
      '()
      (se (countem (- n 1))
          n
          (countem (- n 1))))
\end{verbatim}

\begin{verbatim}
STk> (countem 1)
(1)
STk> (countem 2)
(1 2 1)
STk> (countem 3)
(1 2 1 3 1 2 1)
STk> (countem 4)
(1 2 1 3 1 2 1 4 1 2 1 3 1 2 1)
\end{verbatim}
Tree recursion

A recursive technique in which more than one recursive call is made within a recursive case.
# Pascal's Triangle

<table>
<thead>
<tr>
<th>rows (R)</th>
<th>columns (C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1 1</td>
</tr>
<tr>
<td>2</td>
<td>1 2 1</td>
</tr>
<tr>
<td>3</td>
<td>1 3 3 1</td>
</tr>
<tr>
<td>4</td>
<td>1 4 6 4 1</td>
</tr>
<tr>
<td>5</td>
<td>1 5 10 10 5 1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

- How many ways can you choose C things from R choices?
- Coefficients of the \((x+y)^R\): look in row R
- etc.
(define (pascal C R)
  (cond
    ((= C 0) 1) ;base case
    ((= C R) 1) ;base case
    (else       ;tree recurse
      (+ (pascal C (- R 1))
          (pascal (- C 1) (- R 1))
       )))
  )
> (pascal 2 5)

(pascal 2 5)

( +
  (pascal 2 4)
  (+
    (pascal 2 3)
    (pascal 1 3)
  )
)

( +
  (pascal 1 4)
  (+
    (pascal 1 3)
  )
  (pascal 0 3)
)

⇒ 1
• Write binary, a procedure to generate the possible binary numbers given n bits.

(binary 1) → (0 1)
(binary 2) → (00 01 10 11)
(binary 3) → (000 001 010 011 100 101 110 111)

(define (binary n)
  (if (= n 1)
      (list 0 1)
      (se (prepend-every 0 (binary (- n 1)))
          (prepend-every 1 (binary (- n 1))))))

(define (prepend-every what sent)
  (if (empty? sent) '()
      (se (word what (first sent))
          (prepend-every what (bf sent))))