Today

- Lists
- Lists vs sentences
- List
- Append
- Cons
- Trial by error coding = bad

Ice cream flavors

- I want to list ice cream flavors
- (chocolate mint chip mocha almond fudge)
- (chocolate (mint chip) (mocha almond fudge))
- This is a list! Here is how it was created:
  (list 'chocolate '(mint chip) '(mocha almond fudge))

Lists vs sentences

- Sentences can be made of
  - words/numbers
- Lists can have be made of
  - words/numbers
  - sentences
  - #f or #t
  - procedures
  - other lists!!!

Sentence

- Examples
  - (se 'cat 'dog) → '(cat dog)
  - (se '(cat) '(dog)) → '(cat dog)
  - (se 'cat '(dog)) → '(cat dog)
  - (se '(cat) 'dog) → '(cat dog)
  - (se 'cat '()) → '(cat)
  - (se '() 'dog) → '(dog)

Accessors

<table>
<thead>
<tr>
<th>sentence &amp; word stuff</th>
<th>list stuff</th>
</tr>
</thead>
<tbody>
<tr>
<td>first</td>
<td>car</td>
</tr>
<tr>
<td>butfirst</td>
<td>cdr</td>
</tr>
<tr>
<td>last</td>
<td>☒</td>
</tr>
<tr>
<td>Butlast</td>
<td>☒</td>
</tr>
</tbody>
</table>
other procedures

<table>
<thead>
<tr>
<th>sentence &amp; word stuff</th>
<th>list stuff</th>
</tr>
</thead>
<tbody>
<tr>
<td>empty?</td>
<td>null?</td>
</tr>
<tr>
<td>sentence?</td>
<td>list?</td>
</tr>
<tr>
<td>item</td>
<td>list-ref</td>
</tr>
<tr>
<td>sentence</td>
<td>list</td>
</tr>
</tbody>
</table>

list

- Takes any number of arguments and puts them in a list

\[
\text{list} \quad \text{null} \quad \text{cons} \quad \text{append}
\]

cons

- Takes two arguments
- Makes the first arg the car of the new list
- Makes the second arg the cdr of the new list
- The second argument MUST be a list

append

- Takes two lists and turns them into one
- Both arguments MUST be lists

Examples:

- \( \text{list } \text{cat } \text{dog} \) results in \'(\text{cat dog})\)
- \( \text{list } \text{(cat) } \text{(dog)} \) results in \'(\text{(cat) (dog)})\)
- \( \text{list } \text{cat } \text{(dog)} \) results in \'(\text{(cat) dog})\)
- \( \text{list } \text{cat } \text{dog} \) results in \'(\text{cat dog})\)
- \( \text{list } \text{cat } \text{()} \) results in \'(\text{cat ()})\)
- \( \text{list } \text{() } \text{dog} \) results in \'(\text{() dog})\)
- \( \text{list } \text{cat } \text{() } \text{dog} \) results in \'(\text{(cat ()) dog})\)
- \( \text{cons } \text{cat } \text{dog} \) results in \'(\text{cat dog})\)
- \( \text{cons } \text{(cat) } \text{(dog)} \) results in \'(\text{(cat) (dog)})\)
- \( \text{cons } \text{cat } \text{()} \) results in \'(\text{cat ()})\)
- \( \text{cons } \text{() } \text{dog} \) results in \'(\text{() dog})\)
- \( \text{cons } \text{cat } \text{() } \text{dog} \) results in \'(\text{(cat ()) dog})\)
- \( \text{append } \text{cat } \text{dog} \) results in \'(\text{cat dog})\)
- \( \text{append } \text{(cat) } \text{(dog)} \) results in \'(\text{(cat) (dog)})\)
- \( \text{append } \text{cat } \text{dog} \) results in \'(\text{cat dog})\)
- \( \text{append } \text{() } \text{dog} \) results in \'(\text{() dog})\)
- \( \text{append } \text{cat } \text{() } \text{dog} \) results in \'(\text{(cat () dog})\)
Examples:
1. (append '(cat) '(dog)) → '(cat dog)
2. (append '(cat) '()) → '(cat)
3. (append '() '(dog)) → '(dog)
4. (append '(cat) '(())) → '(cat ())
5. (append '(()()) '(dog)) → '(() () dog)

Trial by error coding

(define (pluralize L)
  (if (null? L)
      '()
      (cons (plural (car L))
            (pluralize (cdr L)))))