Announcements

• Homework 3 Due Wednesday 10/1 @ 11:59pm

• Optional Hog Contest Due Wednesday 10/1 @ 11:59pm

• Project 2 Due Thursday 10/9 @ 11:59pm

  ▪ Project party Monday 10/6, 6pm–8pm in location TBD

• Special event on Tuesday 10/14 @ 7pm in Wheeler: Fireside chat with Founder & CEO of DropBox Drew Houston, hosted by John

• You can submit questions, and I'll ask them: http://goo.gl/HtkXFf
Dictionaries

{"Dem": 0}
Limitations on Dictionaries

Dictionaries are unordered collections of key-value pairs.

Dictionary keys do have two restrictions:

- A key of a dictionary cannot be a list or a dictionary (or any mutable type).

- Two keys cannot be equal; There can be at most one value for a given key.

This first restriction is tied to Python's underlying implementation of dictionaries.

The second restriction is part of the dictionary abstraction.

If you want to associate multiple values with a key, store them all in a sequence value.
Linked Lists
Linked List Data Abstraction

**Constructor:**

```python
def link(first, rest):
    """Construct a linked list from its first element and the rest."""
```

**Selectors:**

```python
def first(s):
    """Return the first element of a linked list s."""

def rest(s):
    """Return the rest of the elements of a linked list s."""
```

**Behavior condition(s):**

If a linked list $s$ is constructed from a first element $a$ and a linked list $b$, then

- $\text{first}(s)$ returns $a$, which is an element of the sequence
- $\text{rest}(s)$ returns $b$, which is a linked list
Implementing Recursive Lists with Pairs

We can implement linked lists as pairs. We'll use two-element lists to represent pairs.

A linked list is a pair

![Diagram of linked list representation as pairs]

The 0-indexed element of the pair is the first element of the linked list

The 1-indexed element of the pair is the rest of the linked list

"empty" represents the empty list

This data structure has many names:
- Linked list (C, Java)
- List (Lisp)
- Forward list (C++)

(Demo)
Sequence Abstraction Implementation
Implementing the Sequence Abstraction

```python
def len_link(s):
    """Return the length of linked list s."""
    length = 0
    while s != empty:
        s, length = rest(s), length + 1
    return length

def getitem_link(s, i):
    """Return the element at index i of linked list s."""
    while i > 0:
        s, i = rest(s), i - 1
    return first(s)
```

**Length.** A sequence has a finite length.

**Element selection.** A sequence has an element corresponding to any non-negative integer index less than its length, starting at 0 for the first element.

(Demo)
Recursive implementations

(Demo)
Linked List Processing

extend
reverse
apply_to_all_link
join_link
partitions
print_partitions

(Demo)
Rooted Trees
Rooted Trees Have a Value at the Root of Every Tree

Previously, trees *either* had branches *or* they were a leaf value; Rooted trees have both

A rooted tree has a root value and a sequence of branches, which are rooted trees

A rooted tree with zero branches is called a leaf

The root values of sub–trees within a rooted tree are often called node values or nodes
Implementing the Rooted Tree Abstraction

```python
def rooted(value, branches):
    for branch in branches:
        assert is_rooted(branch)
    return [value] + list(branches)

def root(tree):
    return tree[0]

def branches(tree):
    return tree[1:]

def is_rooted(tree):
    if type(tree) != list or len(tree) < 1:
        return False
    for branch in branches(tree):
        if not is_rooted(branch):
            return False
    return True

>>> rooted(3, [rooted(1, []), ... rooted(2, [rooted(1, []), ... rooted(1, [])])])
[3, [1], [2, [1], [1]]]  

(Demo)
```

A rooted tree has a root value and a sequence of branches, which are each rooted trees.
Encoding Strings

(Bonus Material)
Representing Strings: the ASCII Standard

American Standard Code for Information Interchange

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NUL</td>
<td>SOH</td>
<td>STX</td>
<td>ETX</td>
<td>EOT</td>
<td>ENQ</td>
<td>ACK</td>
<td>BEL</td>
<td>BS</td>
<td>HT</td>
<td>LF</td>
<td>VT</td>
<td>FF</td>
<td>CR</td>
<td>SO</td>
<td>SI</td>
</tr>
<tr>
<td>1</td>
<td>DLE</td>
<td>DC1</td>
<td>DC2</td>
<td>DC3</td>
<td>DC4</td>
<td>NAK</td>
<td>SYN</td>
<td>ETB</td>
<td>CAN</td>
<td>EM</td>
<td>SUB</td>
<td>ESC</td>
<td>FS</td>
<td>GS</td>
<td>RS</td>
<td>US</td>
</tr>
<tr>
<td>2</td>
<td>!</td>
<td>&quot;</td>
<td>#</td>
<td>$</td>
<td>%</td>
<td>&amp;</td>
<td>(</td>
<td>)</td>
<td>*</td>
<td>+</td>
<td>,</td>
<td>-</td>
<td>.</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>:</td>
<td>;</td>
<td>&lt;</td>
<td>=</td>
<td>&gt;</td>
<td>?</td>
</tr>
<tr>
<td>4</td>
<td>@</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
<td>O</td>
</tr>
<tr>
<td>5</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td>U</td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td>[</td>
<td>\</td>
<td>]</td>
<td>^</td>
<td>_</td>
</tr>
<tr>
<td>6</td>
<td>\</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>m</td>
<td>n</td>
<td>o</td>
</tr>
<tr>
<td>7</td>
<td>p</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
<td>v</td>
<td>w</td>
<td>x</td>
<td>y</td>
<td>z</td>
<td>{</td>
<td></td>
<td></td>
<td>}</td>
<td>~</td>
</tr>
</tbody>
</table>

8 rows: 3 bits

16 columns: 4 bits

- Layout was chosen to support sorting by character code
- Rows indexed 2–5 are a useful 6-bit (64 element) subset
- Control characters were designed for transmission

(Demo)
Representing Strings: the Unicode Standard

- 109,000 characters
- 93 scripts (organized)
- Enumeration of character properties, such as case
- Supports bidirectional display order
- A canonical name for every character

U+0058 LATIN CAPITAL LETTER X

U+263a WHITE SMILING FACE

U+2639 WHITE FROWNING FACE
Representing Strings: UTF-8 Encoding

UTF (UCS (Universal Character Set) Transformation Format)

Unicode: Correspondence between characters and integers

UTF-8: Correspondence between those integers and bytes

A byte is 8 bits and can encode any integer 0–255.

<table>
<thead>
<tr>
<th>bytes</th>
<th>integers</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000000</td>
<td>0</td>
</tr>
<tr>
<td>00000001</td>
<td>1</td>
</tr>
<tr>
<td>00000010</td>
<td>2</td>
</tr>
<tr>
<td>00000011</td>
<td>3</td>
</tr>
</tbody>
</table>

Variable-length encoding: integers vary in the number of bytes required to encode them.

In Python: `string` length is measured in characters, `bytes` length in bytes.

(Demo)