Lecture 12

Lists (& Sequences)
# Announcements for Today

## Reading
- Read 10.0-10.2, 10.4-10.6
- Read all of Chapter 8 for Tue

## Prelim, Oct 13<sup>th</sup> 7:30-9:30
- Material up to October 4th
- Study guide next week

## Conflict with Prelim time?
- Submit to Prelim 1 Conflict assignment on CMS
- Must be in by next Tuesday!

## Assignments
- A2 is almost finished
  - **Tomorrow** in Gates 216
  - Graded out of 50 points
  - **Mean**: 45.8, **Median**: 48
  - **A**: 46 (72%), **B**: 38 (20%)
- A3 due next week
  - Due on Thurs, Oct. 6
  - Will grade over break
# Sequences: Lists of Values

## String
- \( s = 'abc\ d' \)
  - Put characters in quotes
    - Use `\` for quote character
  - Access characters with 
    - \( s[0] \) is `'a'`
    - \( s[5] \) causes an error
    - \( s[0:2] \) is `'ab'` (excludes \( c \))
    - \( s[2:] \) is `'c\ d'`

## List
- \( x = [5, 6, 5, 9, 15, 23] \)
  - Put values inside \([\ ]\)
    - Separate by commas
  - Access values with \([\ ]\)
    - \( x[0] \) is 5
    - \( x[6] \) causes an error
    - \( x[0:2] \) is \([5, 6]\) (excludes 2\(^{nd}\) 5)
    - \( x[3:] \) is \([9, 15, 23]\)
Sequences: Lists of Values

**String**

- s = 'abc d'
- Put characters in quotes
  - Use \\ for quote character
- Access characters with []
  - s[0] is 'a'
  - s[5] causes an error
  - s[0:2] is 'ab' (excludes c)
  - s[2:] is 'c d'

**List**

- x = [5, 6, 5, 9, 15, 23]
- Put values inside [ ]
  - Use commas to separate
- Access values with []
  - x[0] is 5
  - x[6] causes an error
  - x[0:2] is [5, 6] (excludes 2\textsuperscript{nd} 5)
  - x[3:] is [9, 15, 23]
Lists Have Methods Similar to String

\[ x = [5, 6, 5, 9, 15, 23] \]

- **index(value)**
  - Return position of the value
  - **ERROR** if value is not there
  - \( x.\text{index}(9) \) evaluates to 3

- **count(value)**
  - Returns number of times value appears in list
  - \( x.\text{count}(5) \) evaluates to 2

But you get length of a list with a regular function, not method:
\[ \text{len}(x) \]
Representing Lists

Wrong

Box is “too small” to hold the list

Correct

Variable holds id

Put list in a “folder”

\[ x = [5, 7, 4, -2] \]
Lists vs. Class Objects

List

• Attributes are indexed
  ▪ Example: \texttt{x[2]}

\[\begin{array}{|c|}
\hline
x & id2 \\
\hline
id2 & \text{list} \\
\hline
0 & 5 \\
1 & 7 \\
2 & 4 \\
3 & -2 \\
\hline
\end{array}\]

RGB

• Attributes are named
  ▪ Example: \texttt{c.red}

\[\begin{array}{|c|}
\hline
id3 & \text{RGB} \\
\hline
red & 128 \\
green & 64 \\
blue & 255 \\
\hline
\end{array}\]
When Do We Need to Draw a Folder?

• When the value *contains* other values
  ▪ This is essentially what we mean by ‘object’
• When the value is *mutable*

<table>
<thead>
<tr>
<th>Type</th>
<th>Container?</th>
<th>Mutable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>float</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>str</td>
<td>Yes*</td>
<td>No</td>
</tr>
<tr>
<td>Point3</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>RGB</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>list</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Lists are Mutable

• List assignment:
  \( <\text{var}>[<\text{index}>] = <\text{value}> \)
  ▪ Reassign at index
  ▪ Affects folder contents
  ▪ Variable is unchanged

• Strings cannot do this
  ▪ \( s = 'Hello World!' \)
  ▪ \( s[0] = 'J' \) ERROR
  ▪ String are immutable

\[ x = [5, 7, 4, -2] \]

\[ \begin{array}{cccc}
  0 & 1 & 2 & 3 \\
  5 & 7 & 4 & -2 \\
\end{array} \]

\( x[1] = 8 \)
Lists are Mutable

- **List assignment:**
  \[\text{var}[\text{index}] = \text{value}\]
  - Reassign at index
  - Affects folder contents
  - Variable is unchanged

- Strings cannot do this
  - \(s = 'Hello World!'\)
  - \(s[0] = 'J'\) **ERROR**
  - String are **immutable**

- \(x = [5, 7, 4, -2]\)

- \(x[1] = 8\)
List Methods Can Alter the List

- **append(value)**
  - A **procedure method**, not a fruitful method
  - Adds a new value to the end of list
  - \texttt{x.append(-1)} \textbf{changes} the list to \([5, 6, 5, 9, -1]\)

- **insert(index, value)**
  - Put the value into list at index; shift rest of list right
  - \texttt{x.insert(2,-1)} \textbf{changes} the list to \([5, 6, -1, 5, 9,]\)

- **sort()**
  - What do you think this does?
Lists and Functions: Swap

```python
def swap(b, h, k):
    """Procedure swaps b[h] and b[k] in b
    Precondition: b is a mutable list, h and k are valid positions in the list"""
    temp = b[h]
    b[h] = b[k]
    b[k] = temp

swap(x, 3, 4)
```

Swaps b[h] and b[k], because parameter b contains name of list.
**Lists and Functions: Swap**

```python
def swap(b, h, k):
    """Procedure swaps b[h] and b[k] in b
    Precondition: b is a mutable list, h and k are valid positions in the list"""
    temp = b[h]
    b[h] = b[k]
    b[k] = temp
```

*swap(x, 3, 4)*

Swaps b[h] and b[k], because parameter b contains name of list.
Lists and Functions: Swap

```python
def swap(b, h, k):
    """Procedure swaps b[h] and b[k] in b
    Precondition: b is a mutable list, h and k are valid positions in the list"""
    temp = b[h]
    b[h] = b[k]
    b[k] = temp
```

Swaps b[h] and b[k], because parameter b contains name of list.

```
1  temp = b[h]
2  b[h] = b[k]
3  b[k] = temp
```

```
 swap(x, 3, 4)
```

```
   swap  3
      |   |
  b   | id4 | h  3
 temp |  6   | k  4
```

```
   id4
0  5
1  4
2  7
3  X 5
4  5
```

9/29/16  Lists & Sequences 14
Lists and Functions: Swap

```python
def swap(b, h, k):
    """Procedure swaps b[h] and b[k] in b
    Precondition: b is a mutable list, h and k are valid positions in the list"""
    temp = b[h]
    b[h] = b[k]
    b[k] = temp

swap(x, 3, 4)
```

Swaps b[h] and b[k], because parameter b contains name of list.
List Slices Make Copies

\[
x = [5, 6, 5, 9]
\]

\[
y = x[1:3]
\]

\[
x \text{ id5}
\]

\[
y \text{ id6}
\]

\[
\begin{array}{c|c|c|c}
0 & 5 & \text{id5} & \text{list} \\
1 & 6 & & \\
2 & 5 & & \\
3 & 9 & & \\
\end{array}
\]

\[
\begin{array}{c|c|c|c}
0 & 6 & \text{id6} & \text{list} \\
1 & 5 & & \\
\end{array}
\]

\[
\text{copy = new folder}
\]
Exercise Time

• Execute the following:
  >>> x = [5, 6, 5, 9, 10]
  >>> x[3] = -1
  >>> x.insert(1,2)

• What is x[4]?

A: 10
B: 9
C: -1
D: ERROR
E: I don’t know
Exercise Time

• Execute the following:
  >>> x = [5, 6, 5, 9, 10]
  >>> x[3] = -1
  >>> x.insert(1,2)

• What is x[4]?

  -1

• Execute the following:
  >>> x = [5, 6, 5, 9, 10]
  >>> y = x[1:]
  >>> y[0] = 7

• What is x[1]?

  A: 7
  B: 5
  C: 6
  D: ERROR
  E: I don’t know
Exercise Time

• Execute the following:
  >>> x = [5, 6, 5, 9, 10]
  >>> x[3] = -1
  >>> x.insert(1,2)

• What is x[4]?

• Execute the following:
  >>> x = [5, 6, 5, 9, 10]
  >>> y = x[1:]
  >>> y[0] = 7

• What is x[1]?

-1

6
Lists and Expressions

• List brackets [] can contain expressions
• This is a list **expression**
  ▪ Python must evaluate it
  ▪ Evaluates each expression
  ▪ Puts the value in the list
• Example:
  >>> a = [1+2, 3+4, 5+6]
  >>> a
  [3, 7, 11]

• Execute the following:
  >>> a = 5
  >>> b = 7
  >>> x = [a, b, a+b]
• What is x[2]?

A: 'a+b'
B: 12
C: 57
D: ERROR
E: I don’t know
Lists and Expressions

- List brackets [] can contain expressions
- This is a list **expression**
  - Python must evaluate it
  - Evaluates each expression
  - Puts the value in the list
- Example:
  ```python
  >>> a = [1+2,3+4,5+6]
  >>> a
  [3, 7, 11]
  ```

- Execute the following:
  ```python
  >>> a = 5
  >>> b = 7
  >>> x = [a, b, a+b]
  ```
- What is x[2]?

```python
>>> x[2]
12
```
Lists of Objects

- List positions are variables
  - Can store base types
  - But cannot store folders
  - Can store folder identifiers
- Folders linking to folders
  - Top folder for the list
  - Other folders for contents

Example:
```python
>>> r = colormodel.RED
>>> b = colormodel.BLUE
>>> g = colormodel.GREEN
>>> x = [r, b, g]
```

9/29/16

Lists & Sequences
Lists of Objects

• List positions are variables
  ▪ Can store base types
  ▪ But cannot store folders
  ▪ Can store folder identifiers

• Folders linking to folders
  ▪ Top folder for the list
  ▪ Other folders for contents

• Example:
  >>> r = colormodel.RED
  >>> b = colormodel.BLUE
  >>> g = colormodel.GREEN
  >>> x = [r,b,g]