Lecture 7

Lists (& Sequences)
# Announcements For This Lecture

## Readings
- Chapter 10 (lists)
- Fri will cover for-loops

## Assignment 1
- **Due Thursday**
  - Due *before* midnight
  - Submit something…
  - Can resubmit to Sep. 30
- Grades posted Sat/Sun
- Complete the Survey
  - Must answer individually

---

9/19/16  Lists & Sequences  2
Sequences: Lists of Values

**String**

- \( s = 'abc d' \)
- Put characters in quotes
  - Use \( \backslash \) 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'

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td></td>
</tr>
</tbody>
</table>

**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]

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>5</td>
<td>9</td>
<td>15</td>
<td>23</td>
</tr>
</tbody>
</table>
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 2nd 5)
    - $x[3:]$ is [9, 15, 23]
Lists Have Methods Similar to Strings

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

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

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

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

Wrong

x = [5, 6, 7, -2]

Correct

x = id1

id1

0 5
1 7
2 4
3 -2

x = [5, 7, 4, -2]
Representing Lists

Wrong

Correct

Box is “too small” to hold the list

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

Wrong

\[ x = [5, 6, 7, -2] \]

Box is “too small” to hold the list

Correct

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

Put list in a “folder”
Representing Lists

Wrong

$x = [5, 6, 7, -2]$

Box is “too small” to hold the list

Correct

$x = \text{id1}$

Put list in a “folder”

Unique tab identifier

$x = [5, 7, 4, -2]$
Representing Lists

Wrong

\[ x = [5, 6, 7, -2] \]

Box is “too small” to hold the list

Correct

\[ x = \text{id1} \]

Variable holds id

Put list in a “folder”

\[ \text{id1} \]

Unique tab identifier

\[ \begin{array}{c}
0 \\
1 \\
2 \\
3 \\
\end{array} \]

\[ \begin{array}{c}
5 \\
7 \\
4 \\
-2 \\
\end{array} \]

\[ x = [5, 7, 4, -2] \]
Modifying List Contents

- **List assignment:**
  - `<var>[<index>] = <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`
Modifying List Contents

- **List assignment**: 
  ```
  <var>[<index>] = <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`
Exercise: List Assignment

• Assignment copies id into y
  >>> x = [5, 7, 4, -2]
  >>> y = x

• Execute the assignments:
  >>> x[2] = 8
  >>> y[2] = 3

• What is value of x[2]?

  A: 8
  B: 3
  C: id1
  D: I don’t know
Exercise: List Assignment

• Assignment copies id into y
  >>> x = [5, 7, 4, -2]
  >>> y = x

• Execute the assignments:
  >>> x[2] = 8
  >>> y[2] = 3

• What is value of x[2]?

A: 8
B: 3  CORRECT
C: id1
D: I don’t know
Exercise: List Assignment

• Assignment copies id into y
  >>> x = [5, 7, 4, -2]
  >>> y = x

• Execute the assignments:
  >>> x[2] = 8
  >>> y[2] = 3

• What is value of x[2]?
  
  A: 8  
  B: 3  CORRECT  
  C: id1  
  D: I don’t know

A: 8  
B: 3  CORRECT
C: id1  
D: I don’t know
List Methods Can Alter the List

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

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

- **sort()**
  - What do you think this does?

```
x = [5, 6, 5, 9]
```
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.
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)
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)
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

g = [1, 2, 3, 4, 5, 6, 7]
s[3][0] = 5

swap(g, 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.

Frame is erased, but folder is not
List Slices Make Copies

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

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

copy = new folder
Exercise Time

- Execute the following:
  ```python
  >>> 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:
  ```python
  >>> x = [5, 6, 5, 9, 10]
  >>> x[3] = -1
  >>> x.insert(1,2)
  ```
- What is x[4]?

- Execute the following:
  ```python
  >>> x = [5, 6, 5, 9, 10]
  >>> y = x[1:]
  >>> y[0] = 7
  ```
- What is x[1]?

```
-1
```

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

- Execute the following:
  ```python
  >>> x = [5, 6, 5, 9, 10]
  >>> x[3] = -1
  >>> x.insert(1, 2)
  ```
- What is x[4]?

- Execute the following:
  ```python
  >>> 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]?

<table>
<thead>
<tr>
<th>A:</th>
<th>'a+b'</th>
</tr>
</thead>
<tbody>
<tr>
<td>B:</td>
<td>12</td>
</tr>
<tr>
<td>C:</td>
<td>57</td>
</tr>
<tr>
<td>D:</td>
<td>ERROR</td>
</tr>
<tr>
<td>E:</td>
<td>I don’t know</td>
</tr>
</tbody>
</table>
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]?

12
Nested Lists

- Lists can hold any objects
- Lists are objects
- Therefore lists can hold other lists!

\[ a = [2, 1] \]
\[ b = [3, 1] \]
\[ c = [1, 4, b] \]
\[ x = [1, a, c, 5] \]
Two Dimensional Lists

Table of Data

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5</td>
<td>4</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>8</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>0</td>
</tr>
</tbody>
</table>

Each row, col has a value

Images

Each row, col has an color value

Store them as lists of lists (row-major order)

d = 
[[5,4,7,3],[4,8,9,7],[5,1,2,3],[4,1,2,9],[6,7,8,0]]
Overview of Two-Dimensional Lists

- Access value at row 3, col 2:
  \[ d[3][2] \]
- Assign value at row 3, col 2:
  \[ d[3][2] = 8 \]
- An odd symmetry
  - Number of rows of \( d \): \( \text{len}(d) \)
  - Number of cols in row \( r \) of \( d \): \( \text{len}(d[r]) \)
How Multidimensional Lists are Stored

- \[ b = \left[ \left[ 9, 6, 4 \right], \left[ 5, 7, 7 \right] \right] \]

- \( b \) holds name of a one-dimensional list
  - Has \( \text{len}(b) \) elements
  - Its elements are (the names of) 1D lists

- \( b[i] \) holds the name of a one-dimensional list (of ints)
  - Has \( \text{len}(b[i]) \) elements
Ragged Lists: Rows w/ Different Length

- \( b = [[17, 13, 19], [28, 95]] \)

- Will see applications of this later
Slices and Multidimensional Lists

- Only “top-level” list is copied.
- Contents of the list are not altered
- \( b = \langle [9, 6], [4, 5], [7, 7] \rangle \)

\[
x = b[:2]
\]
Slices and Multidimensional Lists

- Create a nested list
  ```python
  >>> b = [[9,6],[4,5],[7,7]]
  ```
- Get a slice
  ```python
  >>> x = b[:2]
  ```
- Append to a row of x
  ```python
  >>> x[1].append(10)
  ```
- x now has nested list
  ```python
  [[9, 6], [4, 5, 10]]
  ```

What are the contents of the list (with name) in b?

A: [[9,6],[4,5],[7,7]]
B: [[9,6],[4,5,10]]
C: [[9,6],[4,5,10],[7,7]]
D: [[9,6],[4,10],[7,7]]
E: I don’t know