Lecture 12

Lists (& Sequences)
Announcements for Today

Reading

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

Assignments

• A2 still being graded
  ▪ Missed some in grading

• Remember the survey
  ▪ Last day for A2
  ▪ Each partner must fill out

• Prelim, Oct 15th 7:30-9:30
  ▪ Material up to October 6th
  ▪ Study guide next week

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

• A3 due next week
  ▪ Due on Friday, Oct. 9
  ▪ Turn in before you leave
# Sequences: Lists of Values

## String

- \(s = 'abc d'\)

<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>

- 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 e)
  - \(s[2:]\) is 'c d'

## List

- \(x = [5, 6, 5, 9, 15, 23]\)

<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>

- 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 2nd 5)
  - \(x[3:]\) is [9, 15, 23]
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'

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]
Lists Have Methods Similar to String

- **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

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

But you get length of a list with a regular function, not method:

`len(x)`
Representing Lists

Wrong

Correct

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

Box is “too small” to hold the list

Put list in a “folder”

Unique tab identifier

Variable holds id

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

List

- Attributes are indexed
  - Example: \( x[2] \)

```
+----+------+
|    | id2   |
+----+------+
| 0  |  5    |
| 1  |  7    |
| 2  |  4    |
| 3  | -2    |
+----+------+
```

RGB

- Attributes are named
  - Example: \( c.red \)

```
+----+------+
|    | id3   |
+----+------+
| red| 128   |
| green| 64   |
| blue| 255   |
+----+------+
```
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>Point</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**

\[
\begin{array}{c}
\text{id1} \\
0 & 5 \\
1 & 7 \\
2 & 4 \\
3 & -2 \\
\end{array}
\]

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

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

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

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

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

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

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

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

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

x = [5, 6, 5, 9]

- **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?

See Python API for more
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
```

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

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

```
 0 5
 1 4
 2 7
 3 6
 4 5
```

```
 1
```

```
 x
```

```
 b
 id4
 3

 h

 k
 id4
 4
```

```
 swap
```

```
 1
```

```
 3
```

```
 4
```

```
 10/2/15
```

```
 lists & sequences
```

```
 10/2/15
```
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)
**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
```

**Example:**

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

**Explanation:**
- The `swap` function swaps elements at positions `h` and `k` in the list `b`.
- The function definition includes a precondition that `b` is a mutable list, and `h` and `k` are valid positions in the list.
- The swap operation is demonstrated with the variables `b`, `h`, `k`, `temp`, and `x`.

**Diagram:**
- The diagram shows a sample list before and after the swap operation.
- The list contains elements: 5, 4, 7, 5, 5.
- The swap operation is illustrated with the values at positions 3 and 4 being exchanged.

**Notes:**
- The swap operation is effective because the parameter `b` contains the name of the list.
- The variables `h`, `k`, and `temp` are used to perform the swap.

**Date:**
- 10/2/15

**Course:**
- Lists & Sequences
**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

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

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

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]?
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:
  >>> 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
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]
  ```
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]
  ```