**Sequences: Lists of Values**

<table>
<thead>
<tr>
<th>String</th>
<th>List</th>
</tr>
</thead>
<tbody>
<tr>
<td>• s = 'abc d'</td>
<td>x = [5, 6, 9, 15, 23]</td>
</tr>
<tr>
<td>0 1 2 3 4</td>
<td>0 1 2 3 4 5</td>
</tr>
<tr>
<td>• Put characters in quotes</td>
<td>• Put values inside []</td>
</tr>
<tr>
<td>• Use \ for quote character</td>
<td>• Separate by commas</td>
</tr>
<tr>
<td>• Access characters with []</td>
<td>• Access values with []</td>
</tr>
<tr>
<td>§ s[0] is 'a'</td>
<td>§ x[0] is 5</td>
</tr>
<tr>
<td>§ s[0:2] 'ab' (excludes c)</td>
<td>§ x[0:6] '5, 6, 9, 15, 23'</td>
</tr>
<tr>
<td>§ s[2:] 'c d'</td>
<td></td>
</tr>
</tbody>
</table>

**String 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

---

**Representing Lists**

<table>
<thead>
<tr>
<th>Wrong</th>
<th>Correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>x = 5, 6, 7, -2</td>
<td>x = [5, 6, 7, -2]</td>
</tr>
</tbody>
</table>

- Box is “too small” to hold the list
- Put list in a “folder”

**Lists vs. Class Objects**

- Attributes are indexed
  * Example: x[6]
- Attributes are named
  * Example: c.red

**Lists are Mutable**

- List assignment:
  * <var>[<index>] = <value>
  * Reassign at index
  * Affects folder contents
  * Variable is unchanged
  * Strings cannot do this
  * s = 'Hello World!'
  * s[0] = 'd' ERROR
  * String are immutable

<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>
List Methods Can Alter the List

- `append(value)`
  - 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)`
  - Puts 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()`
  - Sorts the list in place

List Slices Make Copies

- `x = [5, 6, 5, 9]`
- `y = x[1:3]`

List and Functions: Swap

- `def swap(b, h, k):
  # Swaps b[h] and b[k] in b
  # Precond: b is a mutable list,
  # h, k are valid positions
  temp = b[h]
  b[h] = b[k]
  b[k] = temp

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]`?
  - A: `a+b`
  - B: 12
  - C: 57
  - D: ERROR
  - E: I don’t know

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 = introcs.RGB(255, 0, 0)
  >>> g = introcs.RGB(0, 255, 0)
  >>> b = introcs.RGB(0, 0, 255)
  >>> x = [r, g, b]
  ```

- 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

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

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