CompSci 101
Collections and Strings
Reminders

• KISS

• “Good programmers are simply good designers.”
  • -Dr. Washington

• Design first and always!
Collection Data Type

• Collection of books, toys, shoes
  • Direct access to each item
• Comprised of smaller pieces
  • Strings and lists
• Strings
  • Smaller strings of size one char
  • Empty string - "" or ""
• Operations on strings
  • + → concatenation
  • * → repetition

Code example:

```python
if __name__ == '__main__':
    result1 = "Hey there!"
    result2 = "How are you?"

    # concatenate two strings
    result = result1 + result2
    print(result)
```

```python
if __name__ == '__main__':
    result1 = "Hey there!"
    result2 = "How are you?"

    # repeat a string
    result = result1 * 3
    print(result)
```
Indexing a String

\( \text{string\_name}[\text{index}] \)

- \text{string\_name}-your variable name
- \text{index}-character element directly accessing
  - Leftmost 0 to \text{string\_length}-1

- What about \text{string\_name}[\text{-1}]?
- **Whitespaces in a string count**
Slicing Strings

• Slicing bread, tomatoes, etc.
• Substring (smaller part) of the larger string

\texttt{string\_name}[n:m]

\textit{n}-index of the first character in the substring

\textit{m}-index of the character that immediately follows the last character in the substring
Comparing Strings

• Compares strings to determine the relationship between them
  • ==, >, <, >=, <=, !=

• `string1 == string2`

**need to output this or store the result**
in and not in operators

• Is string1 a substring of string2?

```
string1 in string2
```

string can be a variable or a string literal (e.g., “This is literally an example of a string literal.”)
CompSci 101
Lists
List

- Groceries, errands, names, etc.
- Collection of data values
  - Sequential
  - Directly access each element
  - Elements don’t have to be the same type

```python
list_name = [item1, item2, ...item6]
```

**only top-level items in list**
List access and length

• Similar to strings
  `list_name[index]`
• `list_name`-your variable name
• `index`-character element directly accessing
  • leftmost 0 to `list_length-1`

• What about `list_name[-1]`?
Slicing Lists

- Sublist (smaller part) of the larger list

\[
\text{list\_name}[n:m]
\]

- \(n\)-index of the first character in the sublist

- \(m\)-index of the character that immediately follows the last character in the sublist

```python
if __name__ == '__main__':
    ages = [12, 44, 10, 21]
    names = ['Kim', 'Janay', 'TJ', 'Nia']
    combo = ['Tim', 13, 'Ashanti', [40, 'Pink']]

    # slice lists
    print(ages[1:3])
    print(names[:3])
    print(combo[1:1])
```
in and not in operators

• Is list1 a member of list2?

list1 in list2
list1 not in list2