Chapter 3

Strings and Lists

Strings

- In Python (as in most object-oriented languages) a string is an object of a string class, but it is set up to act as a sequence
- It holds a collection of individual items, in this case letters
- Placed sequentially in memory
- Items are numbered starting with 0

Creating strings

- Already familiar with the following:

  name = input(“Enter your name: ”)

- You may also store literals

  salutation = “To whom it may concern”

Strings in memory

- Each item is stored in an individual memory space sequentially
- May be accessed individually using the index
- The first index (first character in the string) is 0
- All other indices are the number of steps away from the first one

Memory Example

className = “CSE 1284”

Values stored	Index
‘C’	**0**
‘S’	1
‘E’	2
‘ ’	3
‘1’	4
‘2’	5
‘8’	67
‘4’	

Accessing a single character

- Use the variable name (really an object name, but we'll cover that when we get to classes) and the index.
- Example:

```
className = "CSE 1284"
firstCharacter = className[0]
```

Number of characters in a string

- len() function
- Example:

```
numberCharacters = len(className)
```

Lists

- A list is also an object of a class in Python. Specifically, it is a container class.
- Like a string is contains a collection of individual items (usually called elements).
- Placed sequentially in memory
- Items are numbered starting with 0
- Stored similarly in memory to strings

Creating Lists

- Using [] creates a list
- Examples:

```
emptyList = []
listOfFive = [1, 2, 3, 4, 5]
```

Accessing Individual Elements of the List

- Use the index just like with strings:

  listOfFive[3]

- Unlike strings, you may change elements of a list with assignment statements

  listOfFive[3] = 3

- listOfFive now contains [1, 2, 3, 3, 5]

Adding elements

- append() adds elements to the end of the list
- Example:

```
listOfFive.append(28)
```

Now listOfFive has 6 elements with 28 being the last element in the list.

Removing elements

- pop(i) removes the value at index I
- remove(value) removes the first instance of value in the list
- Example:

listOfFive.pop(0) removes the first element

listOfFive.remove(3) removes the first 3 in the list

Common functions

- len(list)
- sum(list)
- min(list)
- max(list)
- list1 + list2
- list.index(value)
- list.count(value)

Membership operators (from chapter 4)

- in / not in
- Works with lists and strings
- value in list – returns True or False
- Example:

```
if 0 in listOfFive:
    listOfFive.remove(0)
```

Identity Operators (chapter 4)

- is / is not
- Checks to see if 2 variables refer to the same object.
- Checks memory space rather than value
- Use == to check value
- Example:

```
if x is y:
    print("The same object")
```

- Good practice is to use == whenever possible instead of is / is not