Lists

Note: `myList = [5, "a", "list", 5, 6]`

Modify Lists

- `myList.append("new") -> [5, "a", "list", 5, 6, "new"]`
- `myList += ["new"] -> [5, "a", "list", 5, 6, "new"]`
- `myList.extend([1,2]) -> [5, "a", "list", 5, 6, 1, 2]`
- `myList.remove(5) -> ["a", "list", 5, 6] # only removes the first 5`
- `myList.pop(2) -> [5, "a", 5, 6] # removes and returns the value at index 2`
- `del myList[2] -> [5, "a", 5, 6] # does not return value`
- `del myList -> # removes entire variable`
- `myList.clear -> [] # empties list but keeps the variable`

Looping Through a List

```
for x in myList:
    print(x)

for i in range(len(myList)):
    print(myList[i])

i = 0
while i < len(myList):
    print(myList[i])
    i += 1

[print(x) for x in myList]
```

Looping Issues

```
i = 0
while (i < 5):
    print(myList[i])

for x in myList:
    myList += [x + 1]
    print(x)
```

Sorting Lists

- `[5, "a", "list", 5, 6].sort()` -> ERROR
  - Can't compare `5` and `"a"`
- `["boat", "car", "airplane"].sort() -> ["airplane", "boat", "car"]`
- `[80, 30, 45].sort() -> [30, 45, 80]`

- `["boat", "car", "airplane"].sort(reverse = True) -> ["car", "boat", "airplane"]`

Copying Lists

- Shallow Copy
  - One object
  - One place in memory
  - Two references / pointers to that object
  - `list_0 = list_1`
- Deep Copy
  - Two objects
  - Two places in memory
  - One reference to each object
  - `list_0 = list_1.copy()`
  - `list_0 list(list_1)`

Built-in List Methods

Method	Description
`append()`	Adds an element at the end of the list
`clear()`	Removes all the elements from the list
`copy()`	Returns a copy of the list
`count()`	Returns the number of elements with the specified value
`extend()`	Add the elements of a list (or any iterable), to the end of the list
`index()`	Returns the index of the first element with the specified value
`insert()`	Adds an element at the specified position
`pop()`	Removes the element at the specified position
`remove()`	Removes the item with the specified value
`reverse()`	Reverses the order of the list
`sort()`	Sorts the list

List Activity

- In a python script and using the command line to run
  - Create a list named `my_list` that holds `["a", 10, "b", 20]`
  - Print the `"b"` using `my_list[2]`
  - Sum the `10` and `20` by selecting the correct positions from the list, similar to above
  - Perform the same addition with `"a"` and `"b"` from the list
  - Now, try with `"a"` and `10`
    - What happens?
    - What is one way to have `"a" + 10` not give an error?

- Iterate over the entire list and print each item in the list:

```
for i in my_list:
    print(i)
```