**Applying and Processing Lists and Strings**

Starting with Practice Of The Day 1

http://www.cs.virginia.edu/~up3f/cs1110/practice-of-the-day/

My Answer:

```
# Write a function called most_common_names that
# takes a list of names and returns 2 things:
# (1) the name that appears the most often, and
# (2) the number of occurrence
#
# If there is a tie, return the first most common name and its occurrence
#
# remember: the order of the return is important
#
# hint: Lists have a function called .count(x) that returns
#       the number of times x appears in the list

def most_common_names(names_list):
  max_count = 0
  max_name = ""
  for  name in names_list:
      times = names_list.count(name)
      if times > max_count:
          max_count = times
          max_name = name
  return max_name, max_count


print(most_common_names(["Tom", "Mary", "Jeff", "Tom", "Jay", "Ann", "Paul",
"Ann", "Ann", "Tom", "Mary", "Tom", "Jay"]))
print(most_common_names(["Jim", "Mary", "Jeff", "Mary", "Jay", "Ann", "Paul",
"Ann"]))
```

Answer we put together as a class:

```
def most_common_names(names_list):
   max_count = 0
   max_name = ""
   # our running variables to keep count
   for i in range(len(names_list)):
       # using this, we access the item as "names_list[i]"
       # if we say instead "for i in names_list", we access the item simply as "i"
       if max_count < names_list.count(names_list[i]):
           # if the maximum occurrences is less than the amount of times "name"
occurs
           max_count = names_list.count(names_list[i])
           # then, replace the max with how many times this name occurs
           max_name = names_list[i]
           # and, change the name that is occurring most often to this new name
   return max_name, max_count
          # returns a tuple “multiple return”, must be in the correct order
```

```
print(most_common_names(["Tom", "Mary", "Jeff", "Tom", "Jay", "Ann", "Paul",
"Ann", "Ann", "Tom", "Mary", "Tom", "Jay"]))
print(most_common_names(["Jim", "Mary", "Jeff", "Mary", "Jay", "Ann", "Paul",
"Ann"]))
```

***Watch these in Visualize Python to clarify how computers treat a list

Strings as Sets, and What We Can Do With Them:
**len("string")** returns the number of characters in a string (its "size")
**"String1" + "string2"** concatenates two strings into "String1string2"
**"String" * integer** repeats a string (eg. "hello" * 3 = "hellohellohello")
**int(string)** casts the string as an int IF possible
**float(string)** casts the string as a float IF possible
**str(value)** converts the value into a string
**in** and **not in** check whether a character or set of characters appears in a string
**isdigit()** and **isspace()** recognize the type of specific characters within a string
**islower()** and **isupper()** check case, **lower()** and **upper()** convert to all one case
**strip(character)** and variations remove characters (see slides)
**join()** combines strings (see slides for variations)
**split(char)** divides the string into two separate strings whenever it sees the character
    If no character is given, it will split at spaces
**count(substring)** tells us how many times the substring appears
**endswith(substring)** checks if it ends with the substring
**find(substring)** tells us its index
**index(substring)** also tells us its index
**replace(old, new)** replaces a substring

| String | Both | List |
|---|---|---|
| Immutable (cannot be changed after assignment) | Is Indexed<br>Can be sliced using [:]<br>Complex data type | Mutable (can be changed after assignment) |

Two strings are equal to one another IFF:
    Same length
    Same sequence
    Same case

In-Class Example of String Manipulation:
(from http://www.cs.virginia.edu/~up3f/cs1110/examples/datatype/)

```
# Convert the sentence to a string in which
# the words are separated by spaces
# and only the first word starts an uppercase letter.
```

```python
# For example, the string "StudyAndDoMorePractice"
# would be converted to "Study and do more practice"

def convert_sentence(sentence):
   result = ""
   word = ""
   for letter in sentence:
       if letter.isupper():
           if len(word) > 0:
               if len(result) > 0:
                   word = word.lower()
               result += word + " "
               word = ""
       word += letter
   result += word.lower()
   return result
# watch in  Visualize Python


# main
in_sentence = "StudyAndDoMorePractice"
expected_sentence = "Study and do more practice"
test1 = convert_sentence(in_sentence)
if test1 == expected_sentence:
   print("You correctly converted " + in_sentence + " to \"" + test1 + "\"")
else:
   print("convert_sentence(" + in_sentence + ") should be \"" + expected_sentence + "\" +
"
           " but you got \"" + test1 + "\". Please check your code.")

in_sentence = "PythonIsFun"
expected_sentence = "Python is fun"
print(convert_sentence(in_sentence))

print(convert_sentence("ProblemSolvingSkillsNeedPractice"))
```