Tough Questions

- The string: "Stringinginginging"
  - How long is the string?
  - How many 'n' characters are there in the string?
  - What is the position of the first 'i' character in the string?

- The list: [23, -56, 2.2, 5, 17, -3]
  - How long is the list?
  - What is the biggest number in the list?
  - What is the smallest number in the list?
  - What is the sum of the numbers in the list?

Abstraction: Sequence

- Strings and lists are both *sequences*
  - A set of things in single-file order
  - It makes sense to talk about the length of a sequence, and the position or *index* of each element
- Python lists are identified by square brackets
  - favorite_numbers = [1, 2, 3, 4]
  - Can hold different data types, but that's usually a bad idea
- Python sequence operators and functions work on lists and strings the same
  - "one" + "two" => "onetwo"
  - [1,2] + [3,4] => [1,2,3,4]
  - len("This string") => 11
  - len([4, 5, 'bob', 17.6]) => 4

Abstraction: Sequence

- In a sequence, each element has an index
  - We count indeces starting with 0
  - scores = [98, 92, 45, 93, 91]
  - The index of the value 98 is 0
  - The index of the value 45 is 2
  - The index of the value 91 is 4
- Each position in the sequence is a variable
  - Stores data, just like any other variable
  - Has a name, just like any other variable
  - That name uses the *subscript operator* []
    - scores[0] evaluates to 98
    - scores[3] evaluates to 93
    - scores[2] = 17 assigns the value 17 into the third slot of the list
    - print(scores[4] + scores[2]) prints 108

Abstraction: Sequence

- Lists and strings are both sequences
  - However, strings are *immutable*
  - You cannot assign new values to the elements of a string

```
name = “Mothuselah”
name[1] = ‘e’
Error!
```

Abstraction: Sequence

- More sequence functions
  - min, max, sum
- *Method* syntax is a different way to call a function
  - len(name) vs. name.len()
  - Tells a certain variable *object* to do something
- Some built in sequence methods
  - my_string = “hop on pop”
  - my_string.index(‘n’) => returns 5, the index of the first ‘n’ char
  - some_list = [15, 4, 15, 6, 15]
  - some_list.count(15) => returns 3, the number of 15s in the list

Abstraction: Sequence

- Computers are really good at repetitive tasks!
  - Simple construct to do something to every element in a sequence
  - For each loop!

```
for item in [1,2,3,4]:
    print(item)
1
2
3
4
```

  - *item* is a variable which is assigned each value in the sequence
  - *for* and *in* are keywords (reserved for the language)
  - The indented lines after the : will be executed once for each item in the sequence