Programming

Algorithms

- A set of rules or steps used to solve a problem
  - https://en.wikipedia.org/wiki/Algorithm

Data Structures

- A particular way of organizing data in a computer
  - https://en.wikipedia.org/wiki/Data_structure

What is not a “Collection”

Most of our variables have one value in them - when we put a new value in the variable, the old value is overwritten.

-
  ```
  $ python
  x = 2
  x = 4
  print(x)
  4
  ```

A List is a kind of collection

Collections allow us to put many values in a single “variable”

A collection is nice because we can carry all many values in one convenient package.

- friends = [‘Joseph’, ‘Glenn’, ‘Sally’]
  carryon = [‘socks’, ‘shirt’, ‘perfume’]

List Constants

List constants are surrounded by square brackets and the elements in the list are separated by commas.

- print([1, 24, 76])
  [1, 24, 76]

A List element can be any Python object - even another List.

- print([‘red’, ‘yellow’, ‘blue’])
  [red, yellow, blue]
    - print([‘red’, 24, 98.6])
      [red, 24, 98.6]
        - print([1, [5, 6], 7])
          [1, [5, 6], 7]

A List can be empty.

- print([])
  []

We already use Lists!

```
for i in [5, 4, 3, 2, 1]:
    print(i)
print(‘Blastoff!’)
```

```
5
4
3
2
1
Blastoff!
```

Lists and Definite Loops - Best Pals

```
friends = [‘Joseph’, ‘Glenn’, ‘Sally’]
for friend in friends:
    print(‘Happy New Year: ‘, friend)
print(Done!)
```

```
Happy New Year: Joseph
Happy New Year: Glenn
Happy New Year: Sally
Done!
```

Looking inside Lists

Just like strings, we can get at any single element in a List using an index specified in square brackets.

Joseph	Glenn	Sally
0	1	2

```
friends = [‘Joseph’, ‘Glenn’, ‘Sally’]
print(friends[1])
Glenn
```

Lists are Mutable

Strings are “immutable” - we cannot change the contents of a string - we just make a new string to make any change.

- fruit = ‘Banana’
  fruit[0] = ‘b’
  Traceback
  TypeError: ‘str’ object does not
  support item assignment
  - x = fruit.lower()
    print(x)
    banana

Lists are mutable - we can change an element of a List using the index operator.

- lotto = [2, 14, 26, 41, 63]
  print(lotto)
  [2, 14, 26, 41, 63]
  - lotto[2] = 28
    print(lotto)
    [2, 14, **28**, 41, 63]

How long is a List?

The len() function takes a List as a parameter and returns the number of elements in the List.

-
  ```
  greet = ‘Hello Bob’
  print(len(greet))
  9
  ```

Actually len() tells us the number of elements of any set or sequence (such as a string….)

-
  ```
  x = [1, 2, ‘joe’, 99]
  print(len(x))
  4
  ```

Using the range() function

The range() function returns a List of numbers that range from zero to one less than the parameter.

- print(range(4))
  [0, 1, 2, 3]
    - friends = [‘Joseph’, ‘Glenn’, ‘Sally’]
      print(len(friends))
      3
        - print(range(len(friends)))
          [0, 1, 2]

We can construct an index loop using for and an integer iterator.

A tale of two loops

```
friends = [‘Joseph’, ‘Glenn’, ‘Sally’]

for friend in friends:
   print(‘Happy New Year:’, friend)

 for i in range(len(friends)):
   friend = friends[i]
   print(‘Happy New Year:’, friend)
```

```
Happy New Year: Joseph
Happy New Year: Glenn
Happy New Year: Sally
```

- friends = [‘Joseph’, ‘Glenn’, ‘Sally’]
  print(len(friends))
  3
    - print(range(len(friends)))
      [0, 1, 2]