INFO5002: Intro to Python for Info Sys

Advanced Data Types

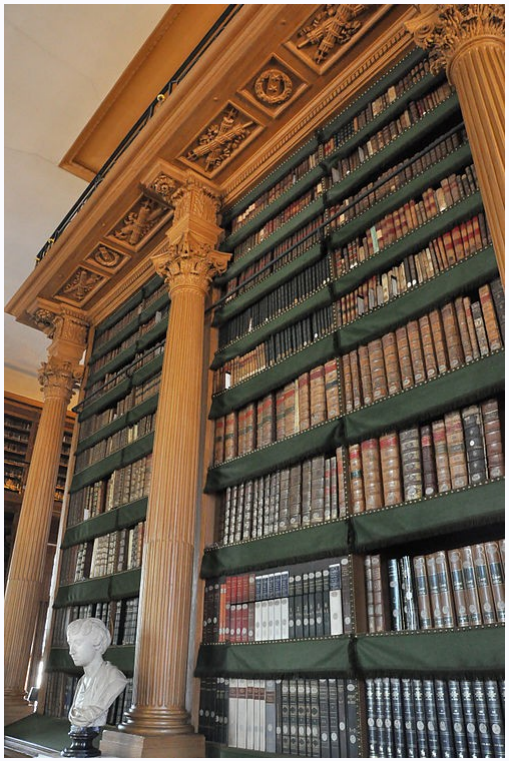
PCC 33-70 and 91-112

Northeastern University

Slides created by: Zachary Doucet

Lists

- To hold a collection of data you can use lists.
  - Todo list
  - Bookshelf
  - Roster



Source: Wikimedia

Working with lists

- We can create a list with brackets.

```
x = []
y = [1, 2, 3]
```

- Lists hold items in a specific order.
- Use brackets to access items on a list.

```
z = [2, 3, 4]
z[1]
```

Watch out: IndexError

- Can also use brackets to assign at a location.

```
q = ["a", "b", "c"]
q[2] = "f"
```

```
x = [1, 8, 4, 2]
_len = len(x)
```

- We can get the number of elements in a list with len.

Modifying lists

- We can add to the end of a list with append.

```
x = [1, 2]
x.append(3)
```

- We can add to a specific index with insert.

```
y = ["a", "c"]
y.insert(1, "b")
```

- We can delete at a specific index with del or pop.

```
x = [1, 2, 3]
del x[0]
```

or

```
x = [1, 2, 3]
x.pop(0)
```

- We can remove at the end with pop.

```
x = [1, 2, 3]
x.pop()
```

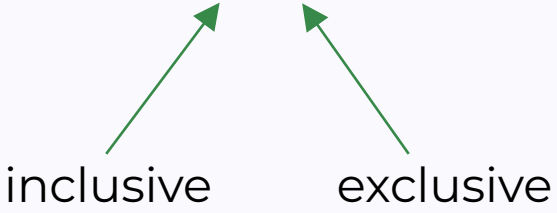
Modifying lists (continued)

- We can remove a specific value *once* with remove.

```
x = [1, 8, 4, 2]
x.remove(8)
```

- We can get elements between two indices with **slice**.

```
x = [1, 2, 3, 4, 5]
y = x[1:2]
```



Tuples

- Immutable ordered collection to store multiple data together.
- Create using parentheses.

```
salad = ("spinach", "tomato", "vinegar")
```

- Get an element with index operator.

```
first_ingredient = salad[0]
```

- Get number of elements with len function.

Dictionaries

- Collection of ordered mutable key-value pairs.
- You cannot have duplicate keys.
- Define with braces and colons.

```
x = {"model": "Kia Rio", "year": 2003, "mpg": 25.32}
```

- Get element with index operator. `x["model"]`
- Modify value of specific key with index op. `x["year"] = 2012`

Dictionaries (continued)

- Get number of elements with len function.

Sets

- Unordered unindexed collection of unique values.
- Define with braces. `fruits = {"mango", "apple", "pear"}`
- Add with add. `fruits.add("banana")`
- You **cannot** access a specific element (unindexed) therefore must use a loop!
- Remove element with remove. `fruits.remove("mango")`

Sequence iteration with for

- We can iterate over a sequence using the for loop.

```
x = [1, 2, 3, 4, 5]
sum = 0
for i in x:
    sum += i
```

```
y = (1, 10, 15)
product = 1
for val in y:
    product *= y
```

```
z = "turnip”
reverse = ""
for c in z:
    reverse = c + reverse
```

```
d = {"x": 1, "y": 2}
resultant = 1
for k in d:
    resultant /= d[k]
```

```
x = {"apricot", "apple", "banana"}
num_a = 0
for fruit in x:
   for character in fruit:
      if character == "a":
         num_a += 1
```

Remember trying to print 5 times?

- We initially did so with a while loop but we can do so also with a for loop.

```
print("hello")
print("hello")
print("hello")
print("hello")
print("hello")
```

```
x = 0
while x < 5:
    print("hello")
    x += 1
```

special library function

```
for i in range(0,5):
    print("hello")
```

The range function

- We can create a sequence of numbers starting from *a* to excluding *b* with a step of *c* with range.

```
range(start, stop, step)
```

What do I see when I execute?

```
for i in range(2, 10, 2):
    print(i)
```

The range function shorthands

```
range(one_argument)
```
==
```
range(0, one_argument, 1)
```

```
range(arg1, arg2)
```
==
```
range(arg1, arg2, 1)
```

Let’s practice

- Create the following function:
  I. sum which takes a list and returns the sum of all the values in the list.
  II. collect_stats which takes in a list of numbers and returns a tuple of the average and the median. Assume list sorted.
  III. remove_odd which takes a list and returns a new list without the odd numbers.
  IV. percent_passed which takes a list of exam grades and returns the percent of students that passed, assuming a pass >= 60.

And some more

- Create the following function:

  I. even_sum which takes a number n and returns the sum of all even numbers between 0 and n. You must use for loop.

  II. average_position which takes in an array of tuples and returns the average (x, y).

  III. remove_duplicate which takes in a list and returns a list with the duplicates removed.