APS106

UNIVERSITY OF TORONTO

Lists 2.0: looping through lists.

Week 4 | Lecture 3 (4.3.2)

if nothing else, write #cleancode

Today's Content

- **Lecture 4.3.1**
  - Lists: indexing and slicing
- **Lecture 4.3.2**
  - **Lists: nested lists and looping**
- **Next Week**
  - Design Problems!
  - Midterm Review

Recap! Adding to a list…

Method	Description	Example

list.append(object)	Append object to end of list	`>>> colours = ['blue', 'yellow']`
		`>>> colours.append('brown')`
		`>>> colours`
		`['blue', 'yellow', 'brown']`

list.extend(list)	Append the items in the list parameter to the list	`>>> colours = ['blue', 'yellow']`
		`>>> colours.extend(['pink','green'])`
		`>>> colours`
		`['blue', 'yellow', 'brown', 'pink', 'green']`

list.insert(int, object)	Insert object at the given index, moving items to make room	`>>> grades = [95, 65, 75, 85]`
		`>>> grades.insert(3, 80)`
		`>>> grades`
		`[95, 65, 75, 80, 85]`

Recap! Removing from a list…

Method	Description	Example

list.remove(object)	Remove the first occurrence of the object; error if not there	`>>> colours = ['blue', 'yellow']`
		`>>> colours.remove('blue')`
		`>>> colours`
		`['yellow']`

list.pop([index])	Remove the item at the end of the list; optional index to remove from anywhere	`>>> colours = ['blue', 'yellow', 'pink']`
		`>>> colours.pop()`
		`'pink'`
		`>>> colours`
		`['blue', 'yellow']`
		`>>> colours.pop(0)`
		`'blue'`
		`>>> colours`
		`['yellow']`

Recap! The fun stuff…

Method	Description	Example

list.reverse()	Reverse the list	`>>> colours = ['blue', 'yellow', 'pink']`
		`>>> colours.reverse()`
		`>>> colours`
		`['pink', 'yellow', 'blue']`

list.sort()	Sort the list from smallest to largest (also sorts list of strings alphabetically)	`>>> grades = [95, 65, 75, 85]`
		`>>> grades.sort()`
		`>>> grades`
		`[65, 75, 85, 95]`

list.count(object)	Return the number of times object occurs in list	`>>> letters = ['a', 'a', 'b', 'c']`
		`>>> letters.count('a')`
		`2`

list.index(object)	Return the index of the first occurrence of object; error if not there	`>>> letters = ['a', 'a', 'b', 'c']`
		`>>> letters.index('a')`
		`0`

List and String Similarities

- Lists share many similarities with strings
  - Indexing (the [ ] operator)
  - Slicing ([start : end] and [start : end : step])
  - Membership (the in operator)
  - Length (built-in function len)
  - Concatenate (the + operator combining lists with other lists)
  - Repeat (the * operator between lists and an integer)
  - Comparison operators (>, < , ==, !=, etc.)

List and String Differences

- Lists can contain a mixture of any Python objects
  - Strings only hold characters
- Lists are mutable (i.e. their elements can be changed)
  - Strings are immutable
- Lists are designated with [ ], with elements separated by commas
  - Strings are designated with " " or ' '

Motivating Example: The Speeder

- We have a list of numbers that represent velocity of a car taken at regular intervals

```
speed_list = [70, 97, 101, 120, 116, 110, 98, 99, 100, 102]
```

- Assuming the speed limit is 100 km/h, we want to examine many times the car is speeding. How do we achieve this?

Motivating Example: The Speeder

- Using what we've learned so far, we would need to write ten if statements to check if velocity is greater than 100 km/h

```
if speed_list[0] > 100:
    print("speeding")

if speed_list[1] > 100:
    print("speeding")

if speed_list[2] > 100:
    print("speeding")

. . .

if speed_list[9] > 100:
    print("speeding")
```

Repeating code -> think loops!

for loops

- A `for` loop starts with the keyword `for`.
- Next, we provide the name of one of more variables
- Our variable `character` will be bound to each of the items in the sequence in turn.
- What is the iterable?
- An iterable is an object that can be iterated over.

GENERAL FORM:

```
for item in iterable:
    do something
```

EXAMPLE:

```
name = 'Sebastian'

for character in name:
    print(character)
```

Example: for Loop through List

- Iterate over a list of strings

```
fruits = ['apples', 'oranges', 'pears', 'apricot']

for fruit in fruits:
    print(fruit)
```

Visualize for Loop through a List

```
fruits = ['apples', 'oranges', 'pears', 'apricot']
for fruit in fruits:
    print(fruit)
```

OUTPUT:

Visualize for Loop through a List

```
fruits = ['apples', 'oranges', 'pears', 'apricot']
for fruit in fruits:
    print(fruit)
```

OUTPUT:

Visualize for Loop through a List

```
fruits = ['apples', 'oranges', 'pears', 'apricot']

for fruit in fruits:
    print fruit
```

OUTPUT:

```
apples
```

Visualize for Loop through a List

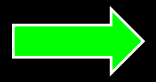
```
fruits = ['apples', 'oranges', 'pears', 'apricot']
for fruit in fruits:
    print(fruit)
```

OUTPUT:

apples

Visualize for Loop through a List

```
fruits = ['apples', 'oranges', 'pears', 'apricot']

for fruit in fruits:
    print fruit
```



OUTPUT:

```
apples
oranges
```

Visualize for Loop through a List

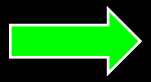
```
fruits = ['apples', 'oranges', 'pears', 'apricot']
for fruit in fruits:
    print(fruit)
```

OUTPUT:

```
apples
oranges
```

Visualize for Loop through a List

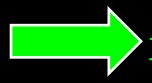
```
fruits = ['apples', 'oranges', 'pears', 'apricot']

for fruit in fruits:
    print fruit
```



```
OUTPUT:
apples
oranges
pears
```

Visualize for Loop through a List

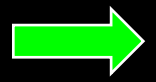
```
fruits = ['apples', 'oranges', 'pears', 'apricot']

for fruit in fruits:
    print(fruit)
```



```
OUTPUT:
apples
oranges
pears
```

Visualize for Loop through a List

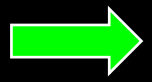
```
fruits = ['apples', 'oranges', 'pears', 'apricot']

for fruit in fruits:
    print fruit
```



```
OUTPUT:
apples
oranges
pears
apricot
```

Visualize for Loop through a List

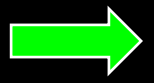
```
fruits = ['apples', 'oranges', 'pears', 'apricot']

for fruit in fruits:
    print(fruit)
```



```
OUTPUT:
apples
oranges
pears
apricot
```

Visualize for Loop through a List

```
fruits = ['apples', 'oranges', 'pears', 'apricot']

for fruit in fruits:
    print(fruit)
```



Next line of code…

```
OUTPUT:
apples
oranges
pears
apricot
```

Let's Code!

- Let's take a look at how this works in Python!
  - Looping through a list
  - BREAKOUT SESSION 1

Open your notebook

Click Link:
1. For Loops Over Lists

Back to Our Speedster

- We have a list of numbers that represent velocity of a car taken at regular intervals

```
speed_list = [70, 97, 101, 120, 116, 110, 98, 99, 100, 102]
```

- Assuming the speed limit is 100 km/h, we want to examine many times the car is speeding. How do we achieve this?

Let's Code!

- Let's take a look at how this works in Python!
  - BREAKOUT SESSION 2

Open your notebook

Click Link:
2. Speedster 1.0

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]

for row in aps106_grades:
    for column in row:
        print(column)
```

OUTPUT:

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

OUTPUT:

```
for row in aps106_grades:
    for column in row:
        print(element)
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

OUTPUT:

```
for row in aps106_grades:
    for column in row:
        print(element)
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

OUTPUT:

```
for row in aps106_grades:
    for column in row
        print(column)
```

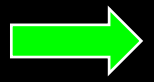
Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

OUTPUT:

Midterm 1

```
for row in aps106_grades:
    for column in row
        print column
```

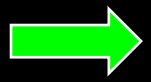


Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

OUTPUT:

Midterm 1

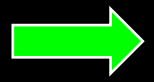


```
for row in aps106_grades:
    for column in row
        print(column)
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

```
for row in aps106_grades:
    for column in row
        print(column
```



OUTPUT:

```
Midterm 1
60
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

```
OUTPUT:
Midterm 1
60
```

```
for row in aps106_grades:
    for column in row
        print(column)
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

OUTPUT:

```
Midterm 1
60
```

```
for row in aps106_grades:
    for column in row
        print(column)
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

OUTPUT:

Midterm 1

60

```
for row in aps106_grades:
    for column in row
        print(column)
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

```
for row in aps106_grades:
    for column in row
        print column
```

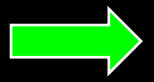
OUTPUT:

```
Midterm 1
60
Midterm 2
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

```
for row in aps106_grades:
    for column in row
        print(column)
```



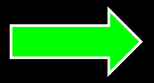
OUTPUT:

```
Midterm 1
60
Midterm 2
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]
```

```
for row in aps106_grades:
    for column in row
        print column
```

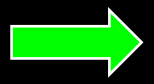


OUTPUT:

```
Midterm 1
60
Midterm 2
90
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]

for row in aps106_grades:
    for column in row:
        print(column)
```



```
OUTPUT:
Midterm 1
60
Midterm 2
90
```

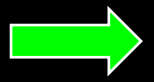
Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]

for row in aps106_grades:
    for column in row
        print(column)
```

```
OUTPUT:
Midterm 1
60
Midterm 2
90
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]

for row in aps106_grades:
    for column in row:
        print(column)
```

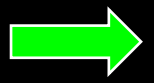


OUTPUT:

```
Midterm 1
60
Midterm 2
90
```

Nested Lists Require Nested Loops!

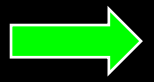
```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]

for row in aps106_grades:
    for column in row
        print column
```



```
OUTPUT:
Midterm 1
60
Midterm 2
90
Exam
```

Nested Lists Require Nested Loops!

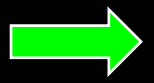
```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]

for row in aps106_grades:
    for column in row
        print(column)
```



```
OUTPUT:
Midterm 1
60
Midterm 2
90
Exam
```

Nested Lists Require Nested Loops!

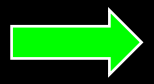
```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]

for row in aps106_grades:
    for column in row
        print column
```



```
OUTPUT:
Midterm 1
60
Midterm 2
90
Exam
100
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]

for row in aps106_grades:
    for column in row
        print(column)
```



```
OUTPUT:
Midterm 1
60
Midterm 2
90
Exam
100
```

Nested Lists Require Nested Loops!

```
aps106_grades = [['Midterm 1', 60],
                 ['Midterm 2', 90],
                 ['Exam', 100]]

for row in aps106_grades:
    for column in row:
        print(column)
```

OUTPUT:

```
Midterm 1
60
Midterm 2
90
Exam
100
```

Let's Code!

- Let's take a look at how this works in Python!
  - Looping through nested lists with nested loops

Open your notebook

Click Link:
3. Looping Over Nested Lists

APS106

UNIVERSITY OF TORONTO

Lists 2.0: looping through lists.

Week 4 | Lecture 3 (4.3.2)

if nothing else, write #cleancode