ENGINEERING
TEXAS A&M UNIVERSITY

Week 7

Lists and Loops

for loops

The *<list>* determining the values of i can be defined in several ways:

```
for i in <list>:
    #do something here
```

`i` will have the ***value*** of each element of the given list.

for loop examples

What are the values of i for each of these? What is the value of sum?

```
for i in [1, 4, 3]:
    sum += i
```

```
for i in range(4):
    sum += i
```

```
for i in range(1, 13, 3):
    sum += i
```

for loop examples

What are the values of i for each of these? What is the value of sum?

```
for i in [1, 4, 3]:
    sum += i
```

```
i → [1, 4, 3]
sum → 8
```

```
for i in range(4):
    sum += i
```

```
i → [0, 1, 2, 3]
sum → 6
```

```
for i in range(1, 13, 3):
    sum += i
```

```
i → [1, 4, 7, 10]
sum → 22
```

for loop examples

What are the values of i for each of these? What is the value of sum?

```
shoe_sizes = [8, 10, 12, 10]
```

```
for i in range(len(shoe_sizes)):
    sum += i
    sum_2 += shoe_sizes[i]
```

```
for i in shoe_sizes:
    sum += i
```

for loop examples

What are the values of i for each of these? What is the value of sum?

```
shoe_sizes = [8, 10, 12, 10]
```

```
for i in range(len(shoe_sizes)):
    sum += i
    sum_2 += shoe_sizes[i]
```

```
i → [0, 1, 2, 3]
sum → 6
sum_2 → 40
```

```
for i in shoe_sizes:
    sum += i
```

```
i → [8, 10, 12, 10]
sum → 40
```

for loop examples

The iterator can have a more meaningful name:

```
grades_list = [90, 84, 92, 100]
```

```
for i in range(len(grades_list)):
    sum += i
    sum_2 += grades_list[i]
```

```
i → [0, 1, 2, 3]
sum → 6
sum_2 → 366
```

```
for grade in grades_list:
    sum += grade
```

```
i → [90, 84, 92, 100]
sum → 366
```

Example: List Looping (i as index)

ENGINEERING
TEXAS A&M UNIVERSITY

What's a potential problem here:

Find an average grade:

```
grades_list = [87, 93, 75, 100, 82, 91, 85]
sum = 0
for i in range(7):
    sum += grades_list[i]
average = sum/7
print(average)
```

Example: List Looping (i as index)

ENGINEERING
TEXAS A&M UNIVERSITY

Find an average grade:

```
grades_list = [87, 93, 75, 100, 82, 91, 85]
sum = 0
for i in range(7):
    sum += grades_list[i]
average = sum/7
print(average)
```

We'll loop through all 7 grades

Example: List Looping (i as index)

ENGINEERING
TEXAS A&M UNIVERSITY

Find an average grade:

```
grades_list = [87, 93, 75, 100, 82, 91, 85]
sum = 0
for i in range(7):
    sum += grades_list[i]
average = sum/7
print(average)
```

In each iteration, we'll increase the sum

And after all iterations, compute the average

Example: List Looping (i as index)

ENGINEERING
TEXAS A&M UNIVERSITY

Find an average grade:

```
grades_list = [87, 93, 75, 100, 82, 91, 85]
sum = 0
for i in range(7):
    sum += grades_list[i]
average = sum/7
print(average)
```

In each iteration, we'll increase the sum

And after all iterations, compute the average

But what happens if I add a new grade to the list? How do I fix it?

Example: List Looping (i as index)

ENGINEERING
TEXAS A&M UNIVERSITY

Loop using range(len(list))

```
grades_list = [87, 93, 75, 100, 82, 91, 85]
sum = 0
for i in range(len(grades_list)):
    sum += grades_list[i]
average = sum/len(grades_list)
print(average)
```

Example: List Looping (i as index)

ENGINEERING
TEXAS A&M UNIVERSITY

Loop using range(len(list))

```
grades_list = [87, 93, 75, 100, 82, 91, 85]
sum = 0
for i in range(len(grades_list)):
    sum += grades_list[i]
average = sum/len(grades_list)
print(average)
```

len(grades_list) == 7

range(len(grades_list)) == [0, 1, 2, 3, 4, 5, 6]

Example: List Looping (i as index)

ENGINEERING
TEXAS A&M UNIVERSITY

What's an easier way of finding the average (without built-in functions)?:

```
grades_list = [87, 93, 75, 100, 82, 91, 85]
sum = 0
for _____ in __________:
    sum +=
average =
print(average)
```

Example: List Looping (i as index)

ENGINEERING
TEXAS A&M UNIVERSITY

What's an easier way of finding the average (without built-in functions)?:

```
grades_list = [87, 93, 75, 100, 82, 91, 85]
sum = 0
for grade in grades_list:
    sum += grade
average = sum/len(grades_list)
print(average)
```

enumerate!

You can get both the index AND value at the *same time* with enumerate:

```
grades_list = [90, 84, 92, 100]
for i, grade in enumerate(grades_list)
    sum += grade
    sum_2 += grades_list[i]
```

```
i → [0, 1, 2, 3]
grade → [90, 84, 92, 100]
sum → 366
sum_2 → 366
```

Changing a list in a loop

Changing i in the loop doesn't change the value in the list:

```
grades = [87, 93, 75, 100, 82, 91, 85]
for i in grades:
    print(i)
    i = 0
    print(i)
print(grades)
```

Console

Changing a list in a loop

Changing `i` in the loop doesn't change the value in the list:

```
grades = [87, 93, 75, 100, 82, 91, 85]
for grade in grades:
    print(grade)
    grade = 0
    print(grade)
print(grades)
```

Console

```
87
0
93
0
75
0
(...)
[87, 93, 75, 100, 82, 91, 85]
```

As with any good rule, there are confusions:

I've told you that changing the `i` in the loop won't change the values in the list. However, you *can* change the list within the loop...

You need to access the elements directly:

```
grades = [87, 93, 75, 100, 82, 91, 85]
for i, grade in enumerate(grades):
    print(grade, end='  ')
    grades[i] = 0
print()
print(grades)
```

```
87  93  75  100  82  91  85
[0, 0, 0, 0, 0, 0, 0]
```

As with any good rule, there are confusions:

I've told you that changing the `i` in the loop won't change the values in the list. However, you *can* change the list within the loop...

You need to access the elements directly:

```
grades = [87, 93, 75, 100, 82, 91, 85]
for grade in grades:
    print(grade, end='  ')
    grades[4] = 0
```

```
87  93  75  100  0  91  85
```

Formatting Numbers in Strings

You've probably noticed many times you would like an easier way to format a number in a string to look nicer:

- a certain number of decimal places
- a specific width of character spaces
- to be an integer, float or exponential

Formatting Numbers in Strings

- Python allows string formatting using ‘conversion specifiers’

```
my_string = 'The time is %2d:%2d %s.' % (time_h, time_m, ampm)
```

%s = string
%d = integer
%f = float
%e = exponential (Ex 1.7e3)

numeric types (%d, %f) also allow ‘flags’:

- %8d → value of minimum field width
- %08d → leading zeros in field width
- %.2f → float precision
- %-s → left-justified

Formatting Numbers in Strings

```
courseID = 'ENGR'
courseNum = 102
courseGrade = 94.3
my_string = 'My grade in %8s%4d is %8.2f.' % (courseID, courseNum, courseGrade)
print(my_string)
```

Formatting Numbers in Strings

```
courseID = 'ENGR'
courseNum = 102
courseGrade = 94.3
my_string = 'My grade in %8s%4d is %8.2f.' % (courseID, courseNum, courseGrade)
print(my_string)
```

```
My grade in     ENGR 102 is    94.30.
```

Working with Lists and Loops

Warm-ups 1 *(all together now)*:

Count the number of strings where the string length is 2 or more, and the first and last character are the same (when given a list of strings).

Working with Lists and Loops

Write a Python program to count the number of strings where the string length is 2 or more, and the first and last character are the same from a given list of strings.

```
wordsList = ['arroyo', 'elephantine', 'toy', 'shines']
counter = 0
for word in wordsList:
    if len(word) >= 2 and word[0] == word[-1]:
        counter += 1
print(counter)
```

Working with Lists and Loops

Warm-ups 2 *(all together now)*:

Print out all elements of a list with a value less than a user-input number.

Working with Lists and Loops

Print out all elements of a list with a value less than a user-input number.

```
list_b = [7, 8, 120, 25, 44, 20, 27]
user_num = float(input('Provide a number: ')
new_list_b = []

for current_item in list_b:
    if current_item < user_num:
        print(current_item)
        new_list_b.append(current_item)
print(new_list_b)
```

Additional Practice Problem

Write a Python program to...

Check whether a character is between ‘a’ and ‘z’.

e.g., m → True

e.g., @ → False

Working with Lists and Loops

Write a Python program to check whether a character is between ‘a’ and ‘z’.

```
my_char = input('Character: ')
if my_char >= 'a' and my_char <= 'z':
   print("True")
else:
   print("False")
```

Additional Practice Problem

Write a Python program to...

Check whether a list contains a specific sublist.

e.g., list_a= [1, 2, 2, 5, 3, 2], sub_list=[2, 3] → False

e.g., list_a= [1, 2, 2, 5, 3, 2], sub_list=[5, 3] → True

Write a Python program to check whether a list contains a sublist.

```
main_list = [2, 4, 3, 5, 7]
sub_list_test = [4, 3]
sub_set_bool = False

# Testing if empty subset, or same as list, or larger than list
if sub_list_test == [] or sub_list_test == main_list:
   sub_set_bool = True
elif len(sub_list_test) > len(main_list):
   sub_set_bool = False

# Otherwise check each element
else:
   for i in range(len(main_list)):
       if main_list[i] == sub_list_test[0]:
           n = 1
           while (n < len(sub_list_test)) and (main_list[i + n] == sub_list_test[n]):
               n += 1

           if n == len(sub_list_test):
               sub_set_bool = True

print(sub_set_bool)
```

Current Assignments

ENGINEERING
TEXAS A&M UNIVERSITY

Complete the ISEN module on the eCommunity page

- Due 10/13, by end-of-day

Lab Assignment 7	Due 10/13 by end-of-day
Lab Assignment 7b	Due 10/13 by end-of-day

Exam next week (Oct 16 for MW sections / Oct 17 for TR sections)

Preactivity: Complete zyBook ch 3.11 and 10.9–10.11 prior to class next week