OMIS 30: Intro to Programming (with Python)

Week 3, Class 2

Introduction to Programming
Instructor: Denis Vrdoljak

Learn programming with Python

Note Taker Needed!

If you take good notes, and would be willing to share your notes in exchange for a $60 giftcard or a letter of commendation, please sign up here:

https://www.scu.edu/disabilities/

Office Hours

Instructor	Days available
Yuan Wang (our TA)	M 2:30-3:30p, W 9-10a
Mike Davis (other section's instructor)	Tu 9:30-10:20a, Th 12-1p
Denis Vrdoljak	Tu 3:40-4:40p, W 5-6p

Course Topics

- ~~Computer setup~~
- ~~Shell - ls, mv (and rename), cp, pwd,cd, '..',mkdir, rm, touch, echo~~
- ~~cat, pipe, output redirect (> and >>), 'python --version' (introduce args)~~
- ~~Python Basics - print, input, math.~~
- Pseudo-code, algorithm design, comments
- iterables (~~lists~~, sets, dicts, ~~strings~~)
- Loops, Nested Loops, Recursion
- Flow Control (~~If, else, elif~~, try, except)
- Functions
- ~~Strings, upper(), lower()~~
- ~~indexing and slicing iterables~~
- lists, extending, ~~appending~~
- ~~mutability~~
- Jupyter Notebooks

Goals from last week

- Understand what dynamic typing means and how it works
- Understand Primitives and Iterables in Python
- Be able to index and slice strings and lists, and work with dictionaries
- Know some common methods for strings, lists and dictionaries
- Know how to print to standard out
- Know how to take in a user input
- Control a Program's flow with IF and WHILE loops

HW Review

By the end of this week you should:

- Understand and be able to use While and For loops
- Know how to write a standalone Python script, and how to run it from the Command Line
- Have made progress on Project 1!

WHILE loop review

- What is the expected output?

```
1  i = 1
2  while i < 11:
3      if i % 2 == 0:
4          print(i, "is even")
5      else:
6          print(i, "is not even")
7      i += 1
```

WHILE loop review

- What is the expected output?

```
1 is not even
2 is even
3 is not even
4 is even
5 is not even
6 is even
7 is not even
8 is even
9 is not even
10 is even
```

```
1  i = 1
2  while i < 11:
3      if i % 2 == 0:
4          print(i, "is even")
5      else:
6          print(i, "is not even")
7      i += 1
```

WHILE Loop Activity

WHILE loop review

- What is the expected output?

```
1  animals = ['Cat', 'Dog', 'Elephant', 'Giraffe']
2  i = 0
3
4  while i < len(animals):
5      print(animals[i] + " found in the barn.")
6      i += 1
```

WHILE loop review

- What is the expected output?

```
1  animals = ['Cat', 'Dog', 'Elephant', 'Giraffe']
2  i = 0
3
4  while i < len(animals):
5      print(animals[i] + " found in the barn.")
6      i += 1
```

```
Cat found in the barn.
Dog found in the barn.
Elephant found in the barn.
Giraffe found in the barn.
```

FOR Loops

- For loops iterate through an iterable.
- This can be a list, a string...or a set, a tuple, a dictionary, etc
- For loops stop once the last element in the iterable is used
- break and continue work with for loops, just like with while loops

FOR Loops

- For loops iterate through an iterable.
- This can be a list, a string...or a set, a tuple, a dictionary, etc
- For loops stop once the last element in the iterable is used
- break and continue work with for loops, just like with while loops

```
barn = ['cat','dog','elephant','giraffe','pig']
for animal in barn:
    print(animal + " found in the barn")
```

FOR Loops

- For loops iterate through an iterable.
- This can be a list, a string...or a set, a tuple, a dictionary, etc
- For loops stop once the last element in the iterable is used
- break and continue work with for loops, just like with while loops

```
barn = ['cat','dog','elephant','giraffe','pig']
for animal in barn:
    print(animal + " found in the barn")
```

output:

```
cat found in the barn
dog found in the barn
elephant found in the barn
giraffe found in the barn
pig found in the barn

In [27]:
```

FOR Loops

- For loops iterate through an iterable.
- This can be a list, a string...or a set, a tuple, a dictionary, etc
- For loops stop once the last element in the iterable is used
- break and continue work with for loops, just like with while loops

```
for letter in "animal":
    print(letter)
```

FOR Loops

- For loops iterate through an iterable.
- This can be a list, a string...or a set, a tuple, a dictionary, etc
- For loops stop once the last element in the iterable is used
- break and continue work with for loops, just like with while loops

```
for letter in "animal":
    print(letter)
```

output:

```
a
n
i
m
a
l

In [29]:
```

For Loops vs While Loops

For loops iterate through an iterable (like a list)

While loops repeat until their boolean equality becomes false

```
# Simple WHILE loop
barn = ['cat','dog','elephant','giraffe','pig']
i=0
# i is a counter that will help us iterate through the list
# when i gets to the end of the list, the loop will stop
while i < len(barn):
    animal = barn[i]
    print(animal + " found in the barn")
    i += 1
    #increment counter
```

For Loops vs While Loops

For loops iterate through an iterable (like a list)

While loops repeat until their boolean equality becomes false

```
for animal in barn:
    print(animal + " found in the barn")
```

For Loops vs While Loops

```
# Simple WHILE loop
barn = ['cat','dog','elephant','giraffe','pig']
i=0
# i is a counter that will help us iterate through the list
# when i gets to the end of the list, the loop will stop
while i < len(barn):
    animal = barn[i]
    print(animal + " found in the barn")
    i += 1
    #increment counter
```

```
for animal in barn:
    print(animal + " found in the barn")
```

Introducing range()

- range() creates a list of numbers.

- Basically, it counts for you. (Remember, Python starts counting at 0!)

So, the list:

Counttofive = [0,1,2,3,4]

can be generated as:

range(5)

Introducing range()

- range() creates a list of numbers. Basically, it counts for you.

```
Counttofive =[0,1,2,3,4]

for i in Counttofive:
    print(i)
```

```
0
1
2
3
4
```

```
for i in range(5):
    print (i)
```

```
0
1
2
3
4
```

range(): Counting from 1

```
Counttofive =[1,2,3,4,5]

for i in Counttofive:
    print (i)
```

```
1
2
3
4
5
```

```
for i in range(1,6):
    print (i)
```

```
1
2
3
4
5
```

range() in a Nutshell

```
Counttofive =[0,1,2,3,4]

for i in Counttofive:
    print(i)
```

```
0
1
2
3
4
```

```
Counttofive =[1,2,3,4,5]

for i in Counttofive:
    print (i)
```

```
1
2
3
4
5
```

```
for i in range(5):
    print (i)
```

```
0
1
2
3
4
```

```
for i in range(1,6):
    print (i)
```

```
1
2
3
4
5
```

FOR loops with range()

- another way to loop is to use range to slice by index

```
# for loops with range
barn = ['cat','dog','elephant','giraffe','pig']
i=0

# anther way to loop is to use range to slice by indexe

for idx in range(len(barn)):
    print(barn[idx] + " found in the barn")

print("\n\n")
```

```
cat found in the barn
dog found in the barn
elephant found in the barn
giraffe found in the barn
pig found in the barn
```

Print Statements - Escape Characters

- Printing special characters in Python like quotes “ or ‘ or a backslash \
  - Need to be ‘escaped’ with a preceding backslash \
  - If you want to print:
    - print(“Have a “nice” day!”) -> Python thinks the quotes are the end/beginning of new strings
    - Can do:
    - print(“Have a \”nice\” day!”) -> Have a “nice” day!
- Can also add special characters to a print statement:
  - \n = newline, \t = tab (there are more)
  - print(“Have a \n nice day!” -> Have a
    nice day!

Continue Line

- In PEP8, the suggested line length for coding in Python is <80
  - Backslash (“\”) can be used anywhere to carry the code to the next line
  - Example to concatenate strings:

```
>> print("Use a line break " + \
    "so that the extra lines do " + \
    "not interrupt the flow of the code")
```

- **Output:** Use a line break so that the extra lines do not interrupt the functionality of the code

  **beware of handling spaces in strings!*

Homework 3.2

Email this homework to: denisvrdoljak@berkeley.edu

- Complete the problems published on GitHub:
  - https://github.com/denisvrdoljak/OMIS30_Fall2018/blob/master/wk3/wk3hw2.md
- Submit as a Jupyter notebook vi aCamino

Appendix

- Reserved keywords in Python:
  - https://pentangle.net/python/handbook/node52.html
  - DO NOT USE THESE as VARIABLE NAMES!
- Dos/Windows vs Unix/Linux:
  - https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/4/html/Step_by_Step_Guide/ap-doslinux.html
  - You should be familiar with the basic commands for navigating aroudn the file structure and modifying/creating files/folders. You should be aware of the harder to remember ones (like grep and vi) so you know what to Google when you need them!