Python Basics

Background Information

Jupyter Notebooks:

- a notebook is a system for combining code and written description
  - run a cell: Shift-Enter
  - run a cell and add new: Alt-Enter
  - keyboard shortcuts: click above and press 'h'
- this text is in a 'markdown cell'
- markdown used for words, description, and paragraphs
  - cheat sheet: https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet (https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet)

Python:

- our programming language for second semester
- a high-level programming language
- popular with data scientists and academic researchers
- style guide: https://www.python.org/dev/peps/pep-0008/ (https://www.python.org/dev/peps/pep-0008/)

High-Level Programming Languages:

- more abstraction than a lower-level language
- code more readable (looks a lot like English)
- code is easier to debug since it is easier to read and understand
- do more with less: your code is often shorter than it would otherwise be
- code will often run more slowly than with a lower-level language

Open Source

- Python and Jupyter are open source projects
  - free for everyone to use (even commercially)
- Open source software:
  - source code shared with all, often free to use/change/distribute
  - many different open source licenses: Creative Commons, MIT, BSD
  - may or may not require attribution to original author
  - many open source projects on GitHub (github.com)

Working with cells in jupyter notebooks

- let's learn how to add cells

- we'll also change between code and markdown

This is a markdown cell

- First Indented Bullet Point
  - Second Indented Bullet Point
    - Thrid indented Bullet Point
      - Fourth Indented Bullet Point

Comments

In [2]:
```
# comment in Python with hashes
# add two numbers:
print(4+7)
```

```
11
```

Whitespace

- in Python, whitespace matters
  - instead of curly braces, use four spaces to mark a block

In [3]:
```
# in this cell, print(n) is part of the block so it is repeated:
n = 0
for i in range(5):
    n = n + 1
    print(n)
```

```
1
2
3
4
5
```

In [23]:
```
# in this cell, print(n) is not part of the block so it only runs once:
n = 0
for i in range(5):
    n = n + 1
print(n)
```

```
5
```

Selection

- also known as decision statements or 'if-else' statements

In [5]:
```
# use 'elif' instead of 'else if'
temp = 35
if  temp < 32:
    print("It is freezing")
elif temp < 55:
    print("It's warming up")
else:
    print("It's pretty nice out")
```

```
It's warming up
```

Iteration

- also known a loops ('for' and 'while')

In [6]:
```
# count to 7:
for i in range(7):
    print(i + 1,"\n")
```

```
1

2

3

4

5

6

7
```

In [31]:
```
# try a while loop
# read code, THEN press Shift-Enter

bottle_count = 10
output = ""

while bottle_count > -0:
    output += str(bottle_count) + ' bottles of root beer on the wall... \n' # st
    output += '   Take one down, make a float... \n'
    bottle_count = bottle_count - 1


print(output)
```

```
10 bottles of root beer on the wall...
   Take one down, make a float...
9 bottles of root beer on the wall...
   Take one down, make a float...
8 bottles of root beer on the wall...
   Take one down, make a float...
7 bottles of root beer on the wall...
   Take one down, make a float...
6 bottles of root beer on the wall...
   Take one down, make a float...
5 bottles of root beer on the wall...
   Take one down, make a float...
4 bottles of root beer on the wall...
   Take one down, make a float...
3 bottles of root beer on the wall...
   Take one down, make a float...
2 bottles of root beer on the wall...
   Take one down, make a float...
1 bottles of root beer on the wall...
   Take one down, make a float...
```

Writing Procedures

- using iteration and selection together

In [32]:
```
# read this code, THEN press Shift-Enter
for i in range(5):    #5 numbers, starting at 0 gives 0,1,2,3,4
    if i % 2 == 1:
        print(i,"is an odd number. \n")
    else:
        print(i,"is an even number. \n")
print("Whitespace takes some time to understand.")
        #change indentation of above line and predict output!
```

```
0 is an even number.

1 is an odd number.

2 is an even number.

3 is an odd number.

4 is an even number.

Whitespace takes some time to understand.
```

In [34]:
```
# read this code, THEN press Shift-Enter
for i in range(5):    #5 numbers, starting at 0 gives 0,1,2,3,4
    if i % 2 == 1:
        print(i,"is an odd number. \n")
    else:
        print(i,"is an even number. \n")
        print("Whitespace takes some time to understand.")
        #change indentation of above line and predict output!
```

```
0 is an even number.

Whitespace takes some time to understand.
1 is an odd number.

2 is an even number.

Whitespace takes some time to understand.
3 is an odd number.

4 is an even number.

Whitespace takes some time to understand.
```

In [35]:
```
# read this code, THEN press Shift-Enter
for i in range(5):    #5 numbers, starting at 0 gives 0,1,2,3,4
    if i % 2 == 1:
        print(i,"is an odd number. \n")
    else:
        print(i,"is an even number. \n")
    print("Whitespace takes some time to understand.")
        #change indentation of above line and predict output!
```

```
0 is an even number.

Whitespace takes some time to understand.
1 is an odd number.

Whitespace takes some time to understand.
2 is an even number.

Whitespace takes some time to understand.
3 is an odd number.

Whitespace takes some time to understand.
4 is an even number.

Whitespace takes some time to understand.
```

In [33]:
```
# using the append method:

```

```
0 is an even number.

Whitespace takes some time to understand.
1 is an odd number.

2 is an even number.

Whitespace takes some time to understand.
3 is an odd number.

4 is an even number.

Whitespace takes some time to understand.
```

Numbers and mathematical operators

In [38]:
```
 # addition
print(8+2)
 # subtraction
print(8-2)
 # multiplication
print(8*2)
```

```
10
6
16
```

In [43]:
```
 # floating-point division (true division)
print(5 / 2)
 # integer division
print(7 // 2)
 # modulus operator
print(7 % 2)
```

```
2.5
3
1
```

In [44]:
```
 # 5 raised to the 2nd power (5^2)
print(5**2)
```

```
25
```

In [313]:
```
# generate 3 random numbers between 10 and 50


# import the random module:
import random

# for loop:
for i in range(5):
    num = random.randint(10,50)
    print(num)
```

```
50
24
13
15
47
```

In [320]:
```
# access documentation
?random.randint
```

Variables and Functions

In [327]:
```
# to create a variable, simply assign a value to it:
grade = 100
print(100)
```

```
Object `print()` not found.
```

In [335]:
```
# use def to define a function:
def print_message(msg):
        output = "Your message is: "
        output = output + msg
        print(output)

# function call
print_message("Hello World!")
```

```
Your message is: Hello World!
```

Strings

In [338]:
```
# single or double quotes for strings, but be consistent
msg1 = "Mr. Nichols is"
msg2 = "the most average teacher"
print(msg1,msg2)
```

```
Mr. Nichols is the most average teacher
```

In [339]:
```
# backslash to encode special characters
new_line = "\n"     # finish this line with your teacher
print('This is a new line ' + new_line + 'of text')
```

```
This is a new line
of text
```

In [340]:
```
# PRELOADED
# r to represent a raw string (example: if you want a backslash to appear and not
not_new_line = r"\n"   # finish this line with your teacher
new_line = "\n"
print('This will not give a ' + not_new_line + ' new line of text')
print ('This will give a' + new_line + 'new line of text')
```

```
This will not give a \n new line of text
This will give a
new line of text
```

In [346]:
```
# triple (double) quotes to create a multi-line string
myMessage = """
The Red Wheelbarrow
William Carlos Williams

 so much depends
upon

a red wheel
barrow

glazed with rain
water

beside the white
chickens."""
print(myMessage)
```

```
/n
The Red Wheelbarrow
William Carlos Williams

 so much depends
upon

a red wheel
barrow

glazed with rain
water

beside the white
chickens.
```

Name: Saipranav venkatakrishnan

Date:2/16/2020

Task 1

- Write a for-loop that prints out all positive perfect squares less than 31^2 that are not divisible by 3

In [25]:
```
num = []
for i in range(31**2):
    if i % 3 != 0 and i % i**(1/2) == 0:
        num.append(i)
print(num)
```

```
[1, 4, 16, 25, 49, 64, 100, 121, 169, 196, 256, 289, 361, 400, 484, 529, 625, 6
76, 784, 841]
```

Task 2

- Part 1: define a function, describe_temp(current_temp), that returns a descriptive string that describes the current temperature
  - Sample outputs:
    - describe_temp(82) might return '82 degrees is beach weather!'
    - describe_temp(71) might return '71 degrees is barbecue weather!'
    - describe_temp(25) might return '25 degrees is Chicago weather.'
- Part 2: Write an expression that passes a random number from 0 to 100 to describe_temp

In [24]:
```
import random
temp = random.randint(0,100)
if  temp >= 82:
    print(str(temp) + " degrees is beach weather")
elif temp >= 71:
    print(str(temp) + " degrees is barbecue weather")
else:
    print(str(temp) + " degrees is Chicago weather")
```

```
40 degrees is Chicago weather
```