Introduction to Python

for applications to biomedical industries

BME 6303 | CRN 19454 | 3 credits

Asynchronous Learning | Lectures & Assignments Available Online

Office Hours, Mondays, 2:30 pm Central

QutubLab.org/python

Instructor: Dr. Amina Ann Qutub

Amina.Qutub@utsa.edu

Additional Assistants:

Byron Long (Byron.Long@utsa.edu)

Erin Pollet (Erin.Pollet@utsa.edu)

Jenny Brethen (Jennifer.Brethen@utsa.edu)

Office Hours, Mondays, 2:30 pm Central or by appointment

What does this course offer?

After completing the course, students will have the foundational background to be able to design their own programs, mine public biomedical data sources, and tackle a range of problems in biology and bioengineering using Python.

Accessing course materials

NOTE: Links to the course site are also available through UTSA's Blackboard. Three coding challenges and one final report are submitted through Blackboard.

Main Course Site:

QutubLab.org/python

Materials (Modules, Videos, Reading):

QutubLab.org/pythonmaterials

Syllabus

QutubLab.org/pythonsyllabus

Detailed syllabus posted UTSA's Blackboard

Introduction to Python for Applications to Biomedical Industries		
Date	**Topic**	**HW & Notes**
Week 1 8/24	Course Overview	
Week 2 8/31	Python Syntax (Data Types, Variables)	
Week 3 9/7	Python Operators	
9/7	***No Class, Labor Day***	
Week 4 9/14	Python Logic Expressions	Coding Challenge I due (9/14)
Week 5 9/21	Python Biomedical Application I *Omics and Biosensor Data Processing*	
Week 6 9/28	Python Functions	

Week 7 10/5	Python Classes, Objects & Inheritance	
Week 8 10/12	Python Modules	Coding Challenge II due (10/12)
Week 9 10/19	Python Biomedical Application II *Biomedical Image and Video Analysis*	
Week 10 10/26	Python File Handling	
Week 11 11/2	Machine Learning & AI in Python	

Week 12 11/9	Python: Intro to Databases	Coding Challenge III due (11/12)
Week 13 11/16	Python Special Topics: Web-scraping	
11/26	***No Class, Thanksgiving Break***	
Week 14 11/30	Python Biomedical Application III *Working with Public Biomedical Data*	
Week 15 12/3	Final Coding Project Presentations	Final Coding Project Presentations Due
12/3	Last Day of Classes	
Week 16 12/11	Final Coding Project Due	Final Coding Project Due

What are included in modules?

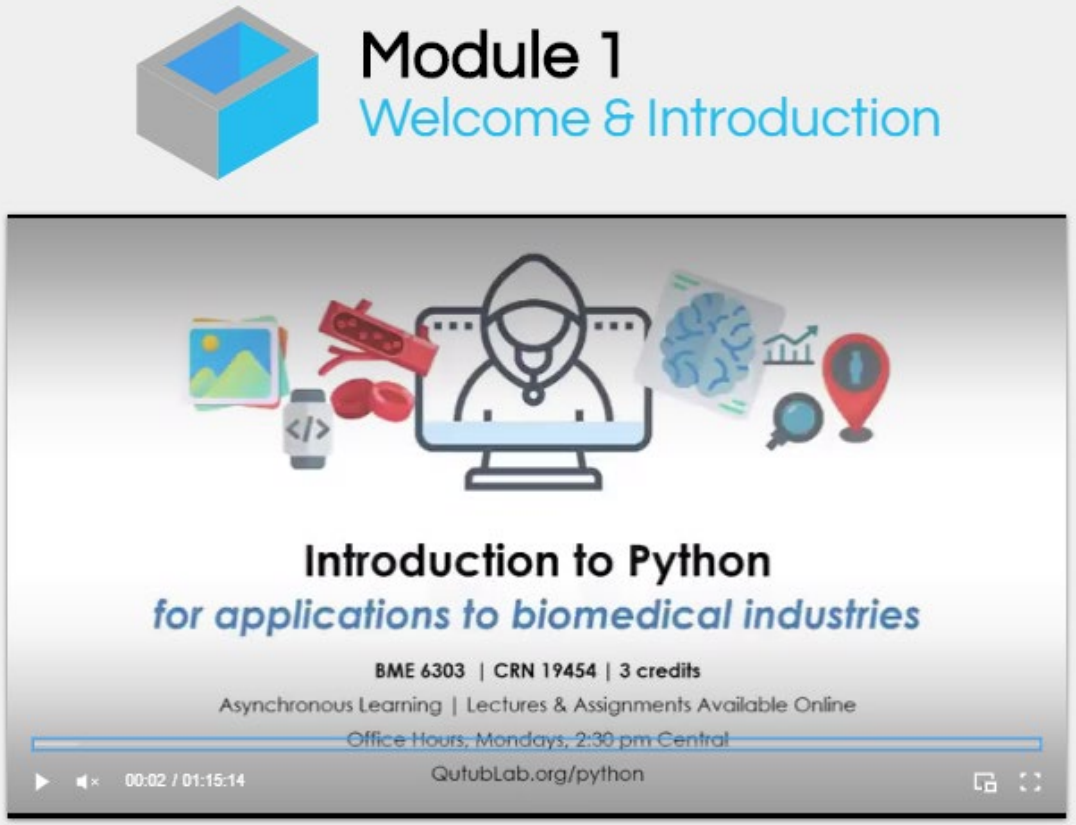
16 Weekly Modules:

QutubLab.org/pythonmaterials

READ: Suggested reading

WATCH: Videos, tutorials

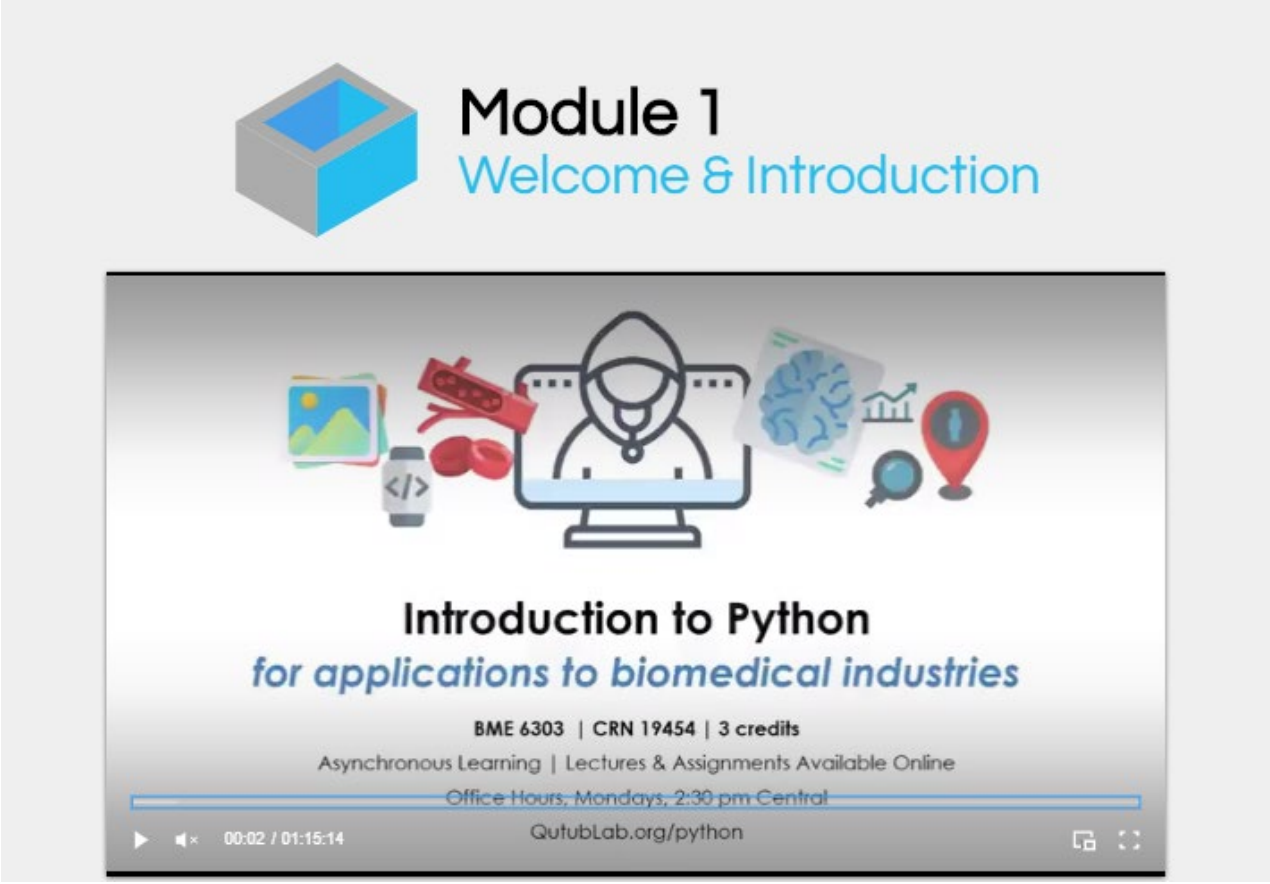
DO: Downloads, programming, coding challenges



Recap for Module 1:
Welcome & Introduction

Read
Beginner's Guide to Python

Watch
Introductory Video for Module 1



Do
1. Download & install Python
2. Download & install PyCharm (or other editor)
3. Bookmark W3Schools Python Intro
4. Sign up for zyBooks and subscribe to the Python book (Code: UTSABME6303Fall2020)
5. Try the first code "3 ways" as presented in the video lecture (@ ~28:20 into the video)

What are the goals of the modules?

Specific Objective I: To gain knowledge of the basic concepts of computer programming by learning the structure, syntax and implementation of the Python language.

Module 2:

Python Syntax (Data Types, Variables)

QutubLab.org/pythonmaterials

Module 2:

Python Syntax (Data Types, Variables)

QutubLab.org/pythonmaterials

Module 2:

Python Syntax (Data Types, Variables)

We'll work in the Command Shell today –

please make sure to open the command shell

QutubLab.org/pythonmaterials

“Computer programming for everyone”

Goals of Python from Guido van Rossum’s DARPA proposal:

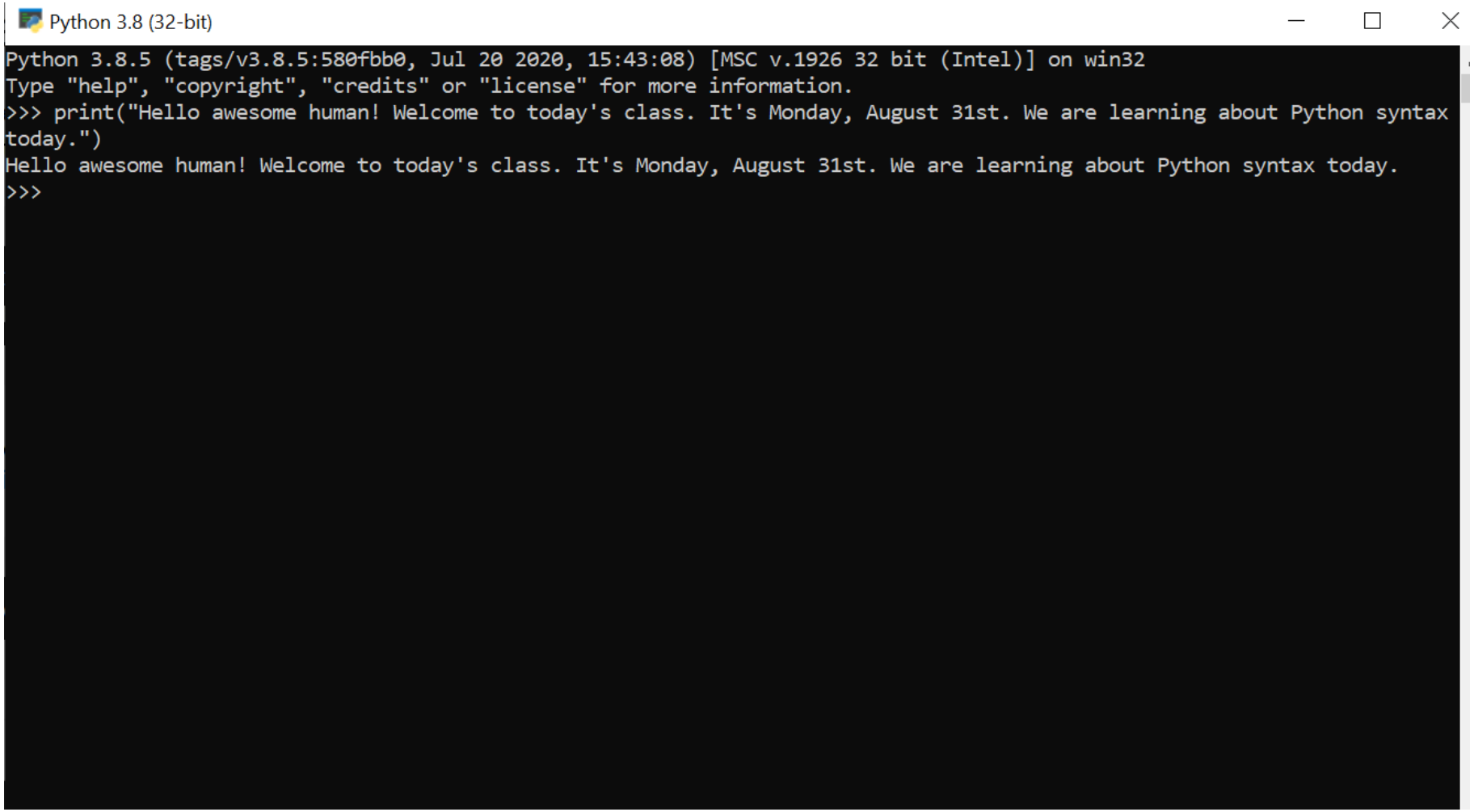
- “**An easy and intuitive language** just as powerful as major competitors
- Open source, so anyone can contribute to its development
- Code that is as **understandable as plain English**
- Suitability for everyday tasks, allowing for short development times”

Python Syntax: Reading References

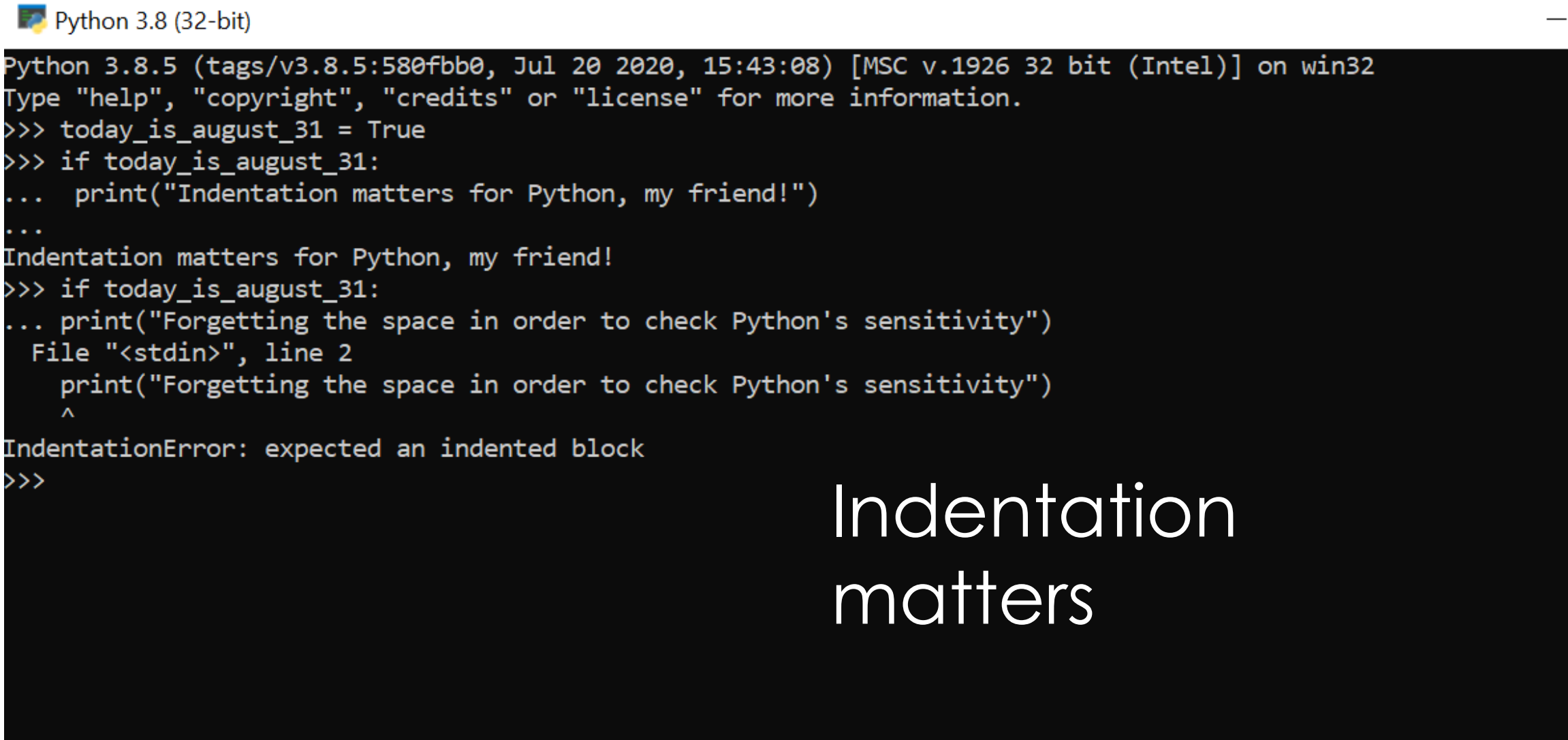
W3Schools Intro to Python Syntax
https://www.w3schools.com/python/python_syntax.asp

Automate the Boring Stuff (Author: Al Sweigart)
https://automatetheboringstuff.com/2e/chapter1/

Python Syntax: print command



Python is like English – nuanced!



Python is like English – nuanced!

```
>>> if today_is_august_31:
...  print("Welcome to our 2nd class of Introduction to Python")
...  print("Let's learn how to use Python syntax. Ready, awesome human?")
...
Welcome to our 2nd class of Introduction to Python
Let's learn how to use Python syntax. Ready, awesome human?
>>> if today_is_august_31:
...  print("Welcome to our 2nd class of Introduction to Python")
... print("Checking - are you really that sensitive to indentation, Python?")
  File "<stdin>", line 3
    print("Checking - are you really that sensitive to indentation, Python?")
    ^
SyntaxError: invalid syntax
>>>
```

Comment like crazy with

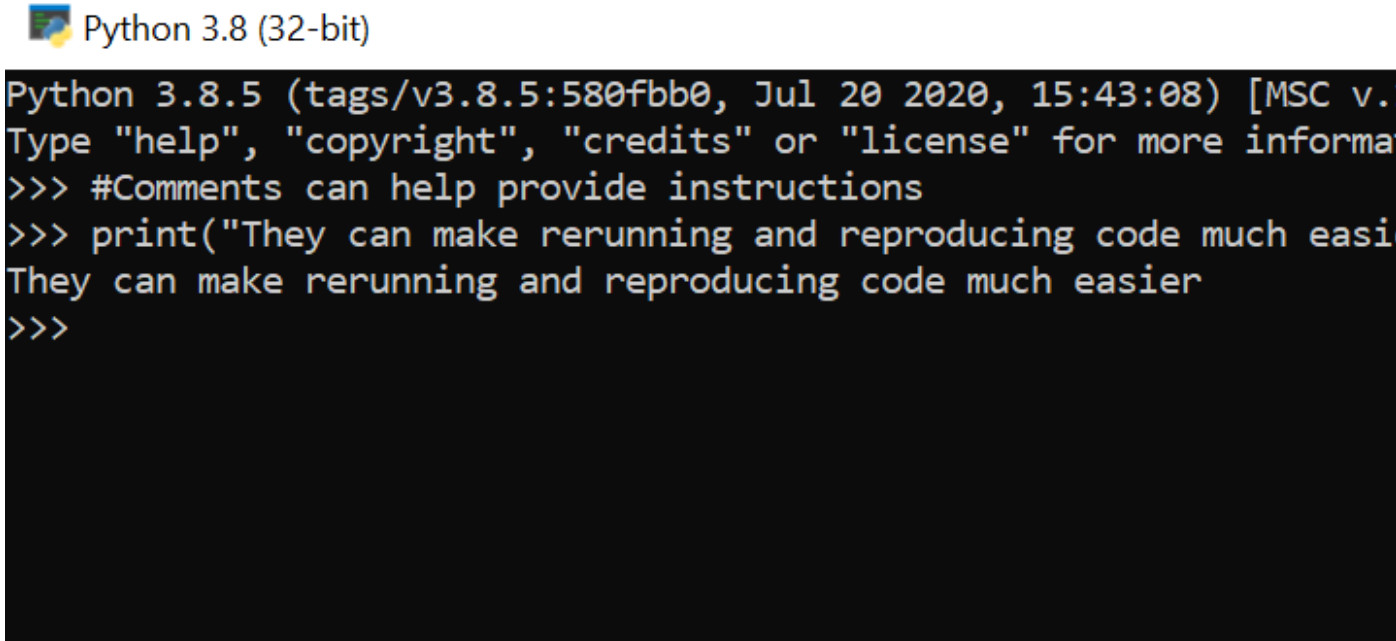
#This is a comment
#It will not be executed
#Comments will be ignored

>>> print("This is interesting") #A comment text can be after a command

"""

Multiple lines can be
commented like strings
in the editor (e.g., PyCharm)

"""



Python makes variables easy

- **Just define it –** assign whatever you want a value, and it becomes a Python variable
- There are no special commands or prefaces to declare a Python variable

```
>>> checkingVariableNames = "Check"
>>> print(checkingVariableNames)
Check
>>> checking_variable_names = "Check 2, team tiger"
>>> print(checkingVariableNames)
Check
>>> print(checking_variable_names)
Check 2, team tiger
>>> are_you_joking2 = "No joke"
>>> print(are_you_joking2)
No joke
>>> are-you_joking2 = "No joke"
  File "<stdin>", line 1
SyntaxError: cannot assign to operator
>>> 2are_you_joking = "No joke"
  File "<stdin>", line 1
    2are_you_joking = "No joke"
     ^
SyntaxError: invalid syntax
>>> variables_can_be_numbers = 5
>>> print(variables_can_be_numbers)
5
>>>
```

Some rules for Python variables

1. Starts with a letter or underscore
2. Case sensitive
3. Cannot start with a number
4. Can only contain underscores or alpha-numeric characters
5. Can end with a number
6. Can be written with camel case (e.g., camelCase)
7. Multiple variables can be assigned on one line

Python variables can be transformers

- **Just define the variable –** no need to declare the type of variable
- Variable type can change
- Example variable types:
  int - integer
  str – string
  float – floating point

```
>>> variable_x = "Awesome human" #variable_x is a str (String)
>>> print(variable_x)
Awesome human
>>> variable_x = 42 #variable_x is now a type int rather than a type str
>>> print(variable_x)
42
>>> variable_x = "Awesome human, variable_x is the answer to the ultimate
 (String)
>>> print(variable_x)
Awesome human, variable_x is the answer to the ultimate question of the u
>>>
```

Python variables can be transformers

```
>>> print(variable_x)
Awesome human, variable_x is the answer to the ultimate question of the universe
>>> print(type(variable_x))
<class 'str'>
>>> variable_x = 42 #variable_x is now a type int rather than a type str
>>> print(type(variable_x))
<class 'int'>
>>> variable_x = 42.314
>>> print(type(variable_x))
<class 'float'>
>>> variable_x, variable_y, variable_z = "Let's", "become", "white hat hackers"
>>> print(variable_z)
white hat hackers
>>> print(type(variable_z))
<class 'str'>
>>> print(variable_y)
become
>>>
```

Concatenating Python variables

```
>>> variable_x, variable_y, variable_z = "Let's", "become", "white hat hackers"
>>> print(variable_z)
white hat hackers
>>> print(type(variable_z))
<class 'str'>
>>> print(variable_y)
become
>>> v = variable_z + variable_y
>>> print(v)
white hat hackersbecome
>>> v = variable_z + " " + variable_y
>>> v = variable_z + variable_y
>>> v = variable_z + " " + variable_y
>>> print(v)
white hat hackers become
>>> v2 = v + "part of a tiger team"
>>> print(v2)
white hat hackers becomepart of a tiger team
>>> v2 = v + " part of a tiger team"
>>> print(v2)
white hat hackers become part of a tiger team
>>>
```

Python variables of all varieties

Text Type:	**str**
Numeric Types:	**int, float, complex**
Sequence Types:	**list, tuple, range**
Mapping Type:	**dict**
Set Types:	**set, frozenset**
Boolean Type:	**bool**
Binary Types:	**bytes, bytearray, memoryview**

Python variables -> casted like actors

```
>>> x = 42 #x is a type int
>>> print(type(x))
<class 'int'>
>>> x_casted = float(x) #x was cast as a type floating number
>>> print(type(x))
<class 'int'>
>>> print(type(x_casted))
<class 'float'>
>>> x_casted = float(x) #x_casted is a type floating number
>>> print(type(x_casted))
<class 'float'>
>>> print(x_casted)
42.0
>>> x = complex(x) #now x is casted as a complex number
>>> print(x)
(42+0j)
>>>
```

```
x = 42          # int
y = 42.314      # float

z = 1j      # complex (a + bj, where a and b are real numbers, and j represents the imaginary unit)
```

Playing with Python variable types

```
>>> tiger_team_C = {"name" : "Saketh", "team": 1}
>>> print(type(tiger_team_C))
<class 'dict'>
>>> tiger_team_D = {"Jason", "Victoria"}
>>> print(type(tiger_team_D))
<class 'set'>
>>> tiger_team_E_includes_Aaron = True
>>> print(type(tiger_team_E))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'tiger_team_E' is not defined
>>> print(type(tiger_team_E_includes_Aaron))
<class 'bool'>
```

Python strings are arrays

```
>>> tiger_team_F = "Many great people including Anuja and Shan"
>>> print(tiger_team_F[0])
M
>>> print(tiger_team_F[1])
a
>>> print(tiger_team_F[6:17])
reat people
>>> print(tiger_team_F[5:17])
great people
>>> print(tiger_team_F[5:16])
great peopl
>>>
```

Python lists, tuples, sets and dictionaries are arrays

Four collection data types

List
- ordered and changeable. Allows duplicate members.

Tuple
- ordered and unchangeable. Allows duplicate members.

Set
- unordered and unindexed. No duplicate members.

Dictionary
- unordered, changeable and indexed. No duplicate members.

Python set example –more to come

```
>>> tiger_team_G = {"Zeus", "Athena"}
>>> print(len(tiger_team_G))
2
>>> print(len(tiger_team_G)) #number of items in set tiger_team_G
2
>>> tiger_team_G.remove("Zeus")
>>> print(len(tiger_team_G)) #number of items in set tiger_team_G
1
>>> print(tiger_team_G)
{'Athena'}
>>>
```

Intro to Python Booleans

```
>>> print("Let's learn about booleans")
Let's learn about booleans
>>> today_is_august_31 = True
>>> today_is_august_31 = True #this variable is a boolean
>>> print(bool("Hello awesome human")) # Oddly, strings can also be evaluated as booleans
True
>>> print(bool(42)) # So can type int
True
>>> bool(123)
True
>>> bool("abcdefgh")
True
>>> bool(["Babble On Dude", "Okay"])
True
>>> bool(False) # some things are actually false when evaluated
False
>>> bool(None) # like the obvious ones
False
>>> bool(0)
False
>>> bool([]) # or not so obvious
False
>>> bool("")
False
>>> print(10>20)
False
>>> print(20>10)
True
>>>
```

Python Syntax: Reading References

W3Schools Intro to Python Syntax
https://www.w3schools.com/python/python_syntax.asp

Automate the Boring Stuff (Author: Al Sweigart)
https://automatetheboringstuff.com/2e/chapter1/

Module 2:
Python Syntax (Data Types, Variables)

Read

W3Schools: Python Syntax Intro

Watch

Introductory Video for Module **2 – posted this week**

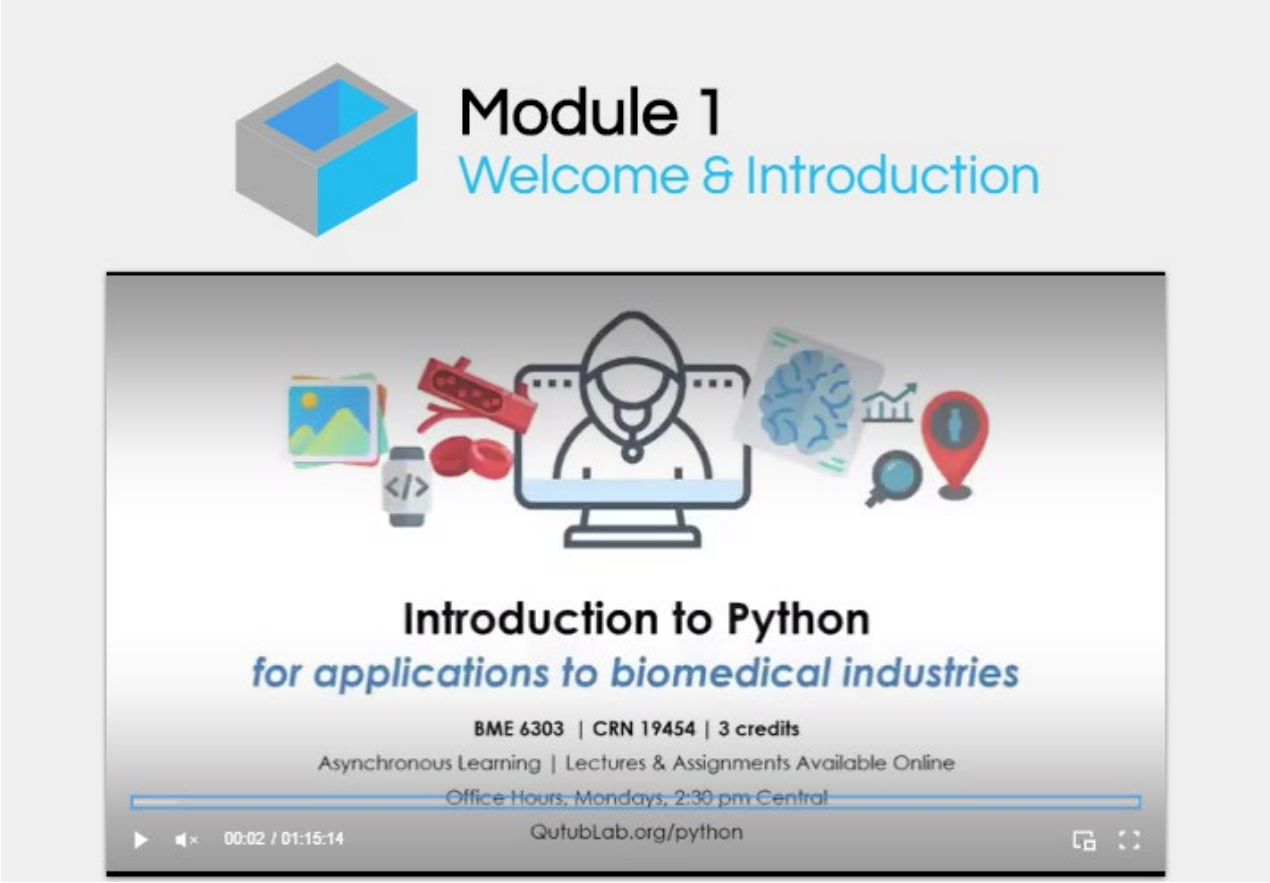
Do

1. Complete Class Survey
2. Make & Upload on Dropbox an 15-30 sec Introduction Video about yourself
3. Complete W3Schools Python Exercises 1-35, **Syntax through Booleans**
4. Optional: Complete zyBooks Chapters 1 (Intro) & 2 (Variables & Expressions)
5. Try the "Joining a tiger team" syntax practice as presented in the video lecture

Recap for Module 1:
Welcome & Introduction

Read
Beginner's Guide to Python

Watch
Introductory Video for Module 1



Do
1. Download & install Python
2. Download & install PyCharm (or other editor)
3. Bookmark W3Schools Python Intro
4. Sign up for zyBooks and subscribe to the Python book (Code: UTSABME6303Fall2020)
5. Try the first code "3 ways" as presented in the video lecture (@ ~28:20 into the video)