

Course Objectives

After completing this course, students will be able to:

* Summarize the CTE squad's responsibilities, objectives, and deliverables from each CPT stage
* Analyze threat information
* Develop a Threat Emulation Plan (TEP)
* Generate mitigative and preemptive recommendations for local defenders
* Develop mission reporting
* Conduct participative operations
* Conduct reconnaissance
* Analyze network logs for offensive and defensive measures 

Course Objectives (Continued)

Students will also be able to:

* Analyze network traffic and tunneling protocols for offensive and defensive measures
* Plan non-participative operations using commonly used tools, techniques and procedures (TTPs)

Module 2: Threat Emulation (Objectives)

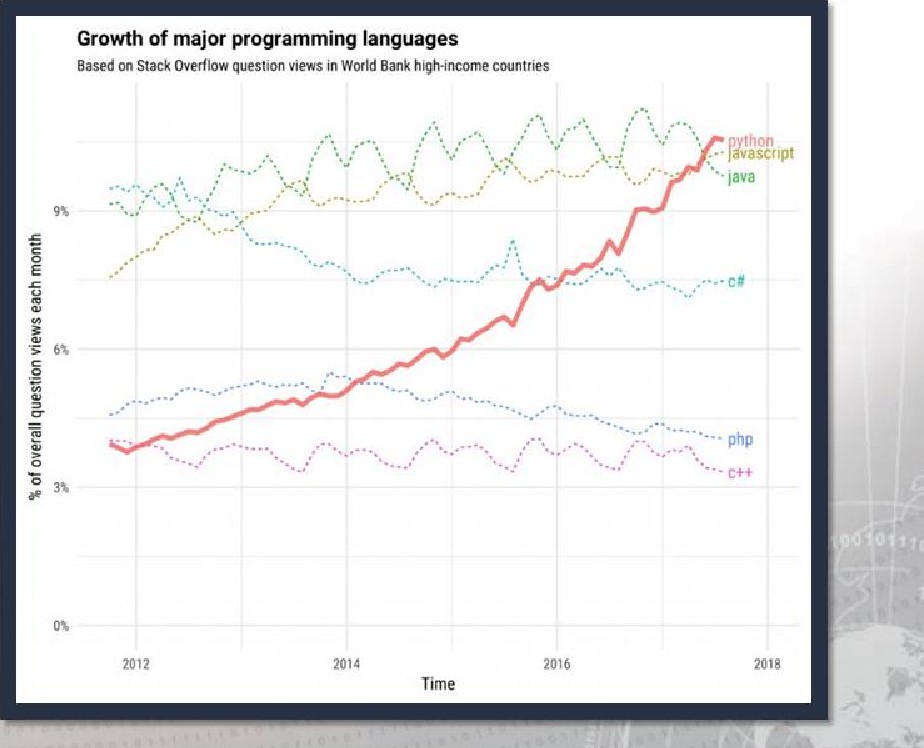
* Conduct reconnaissance
* Generate mission reports from non-participative operations  Plan a non-participative operation using social engineering
* Plan a non-participative operation using Metasploit
* Analyze network logs for offensive and defensive measures
* Analyze network traffic and tunneling protocols for offensive and defensive measures
* Plan a non-participative operation using Python
* Develop fuzzing scripts
* Develop buffer overflow exploits

Module 2 — Lesson 9: Python Refresher (Objectives)

* Manipulate variables, strings, lists, dictionaries, conditionals, loops, and functions in Python
* Create variables, strings, lists, dictionaries, conditionals, loops, and functions in Python

Python History

* 1989 — Python was created
* 1991 - Python 1
* 2000- Python 2
* Most applications still use Python 2
* Python 2 has features that are not forward compatible
* End of life projected for 2020
* 2008- Python 3

Why Python?

* Popularity
* Availability

Why Python? (continued)

Many benefits:

* Open source
* Easy to learn
* Easy to read
* Many modules are available
* Runs on multiple platforms
* Suitable for everyday tasks
* Web development
* Data science
* Machine learning
* Mine Twitter data
* Create games

What is Python?

Python is interpreted

* Similar to PERL and PHP
* No compiling is needed like in C or C++  Python is processed at runtime by an interactive interpreter

Python is interactive

* Type straight into an interactive CLI Python interpreter

Python is Object-Oriented

* Python supports Object-Oriented programming that encapsulates code within an object

What is Python: Compiled vs Interpreted Languages

|  |  |
| --- | --- |
|  |  |
| |  | | --- | |  | | The compiler compiles the code into binary for the machine to use. | |
|  |

Interactive Interpreter or REPL (Read Evaluate Print Loop)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Interactive Interpreter or    REPL (Read Evaluate Print Loop) |  | | #type python3 to launch the interactive interpreter  #the interpreter reads the input which is 40 + 2  #the interpreter evaluates the input  #the interpreter will print the response  #the interpreter will loop back to the beginning and wait for the next thing to read | |  | |  | |  | |  |

# Editors & Integrated Development Environments (IDES)





Windows Linux osx

|  |
| --- |
|  |
| Xcode   * EditXT   PyCharm   * Pydev   TextMate  IDLE |

|  |
| --- |
|  |
| * Notepad++ * Visual Studio PyScripter * PyCharm   Atom  Ema  IDLE |

* Ninja-IDE

PyCharm

. Gedit

* VIM

Atom Nano

* IDLE

# Bonus Activity

• An "always-on" exercise is available to you throughout the Python portion of this course.

eyÉhon RHENR

I

# Variables: Creating

* Variables are the basic building blocks in programming
* Everything is seen as an object
* Variables keep track of state
* State: id, type, and value associated with the object 

assignment operator

variable name  string literal

bank account = "empty"

the state of bank\_account is empty



# Variables: Naming Convention

|  |
| --- |
| import keyword print(keyword.kwlist)  [ ' False ' , 'None' ' True ' 'and' , ' ' assert '  'break' , ' class ' ' continue , ' def' 'del' ,  'elif' , 'else ' , 'except', 'finally',  'from' , 'global' , ' if' , ' import ', '  ' lambda ', 'nonlocal ' , 'not', ' 'pass '  ' return ' 'try' ' 'while', 'with ', 'yield ' ] |

* Pythonic code and Pythonistas
* Do not use keywords as variable names!
* Results in a SyntaxError
* Use an underscore to separate words

|  |
| --- |
| Standards    https://www.python.org/dev/peps/ https://pep8.org/ |
|  |

* Do not start with numbers
* Keep lowercase

|  |
| --- |
| Help Function |
|  |
|  |

# Variables: Naming Convention

|  |
| --- |
| dir(\_builtins\_)  [ ' ArithmeticError' ,  ' AssertionError '  ' AttributeError', ' BaseException' ,  ' BlockingIOError' ,  ' BrokenPipeError' ,  ' BufferError', ' BytesWarning' ,  ' ChildProcessError' ,  ' ConnectionAbortedError ' ,  ' ConnectionError' ,  ' ConnectionRefusedError' ,  ' ConnectionResetError '  ' Deprecationwarning ' , 'EOFError',  ' Ellipsis ' ' EnvironmentError' , |

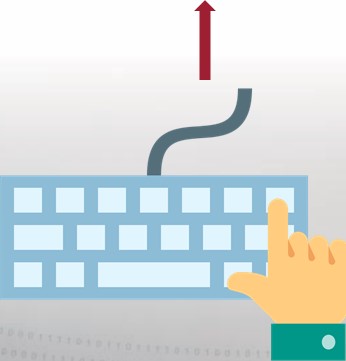
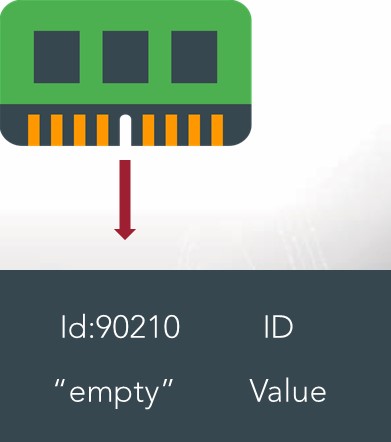
* Python will let you use built-ins as variables, but you should avoid using them.

Commonly Used Built-ln Functions

dict id list max open range str sum type

* Fight the temptation to use these as variable names!

# Variables: value

• value: the data that the object holds

bank account = "empty"

bank account empty"

Type:String Type

# Variables: id

• id: A unique ID for the object; its location in memory

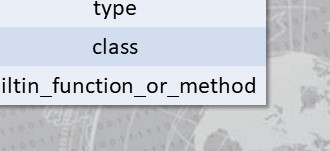
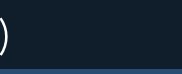
Explain what is occurring with these variables?

Notice anything?

# Variables: type Objects and Types

|  |  |
| --- | --- |
| Objects | Types |
| String str  Integer int  Floating Point float  List list  Dictionary dict  Tuple tuple  function function  type type  subclass of class class  Built-in function builtin function or method | |

|  |
| --- |
|  |
| >>>type(name)  <class ' str ' > |

* type: the class of the object
* Reveal an object's type by passing it through type(): type()

## Writing and Reading

print Nobody likes you') #print will write to standard out

Nobody likes you

> > > print ('We are', 2, 'wild and crazy guys' ) #print statement w/ strings and numbers

We are 2 wild and crazy guys print ('The' , 92, 'Dream Team')

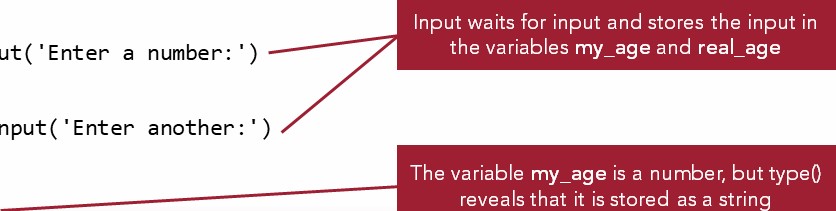
The 92 Dream Team my\_age = input ("What's my age again?" ) \*waits for input and stores answer in variable my \_age

What' s my age again? 23 my\_age

'23'



# Writing and Reading



input(

input(

type(my\_age)

my

\_

age

=

23

real

\_

age

=

42







<class ' str ' > my\_age + real\_age

'

' 2342 int(my\_age) + int (rea 1\_age)

65

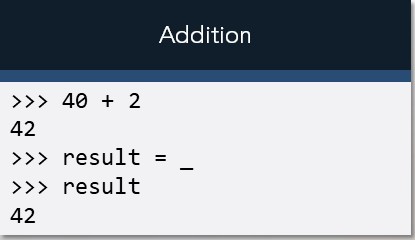
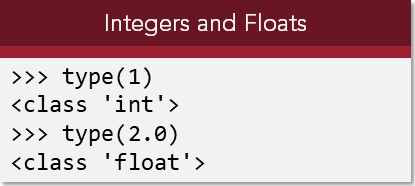
into converts my\_age and real\_age from strings into integers, letting you add them

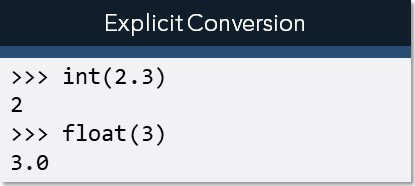


Mutability



|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | Mutable objects can change their value; they can alter their state but their identity stays the same.  Mutable Types:   * Dictionaries * Lists | |  |

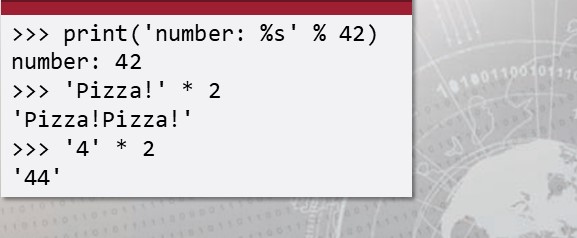
Numbers



Coercion

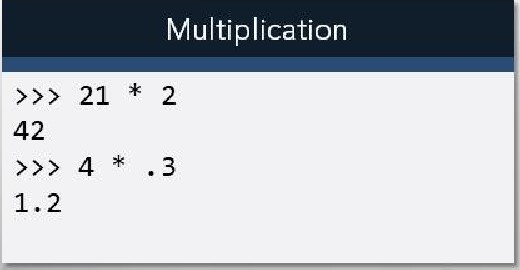
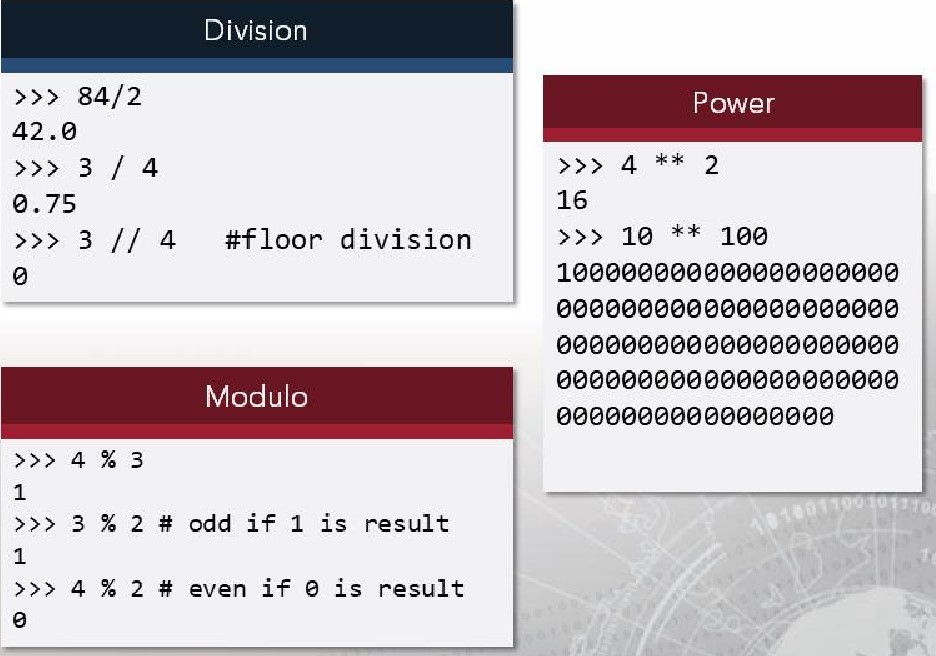
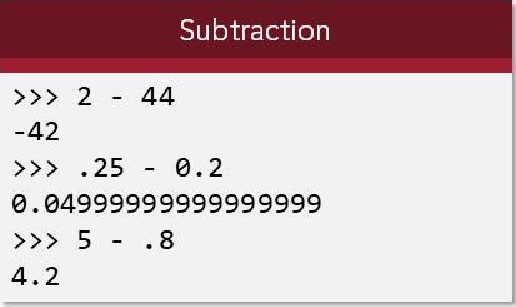
Between

Strings



and

Numbers

Numbers

|  |
| --- |
|  |
| >>> 4 + 2 \* 3  10    18 |

## Numbers

 Python follows the traditional

PEMDAS order of operations.

("Please excuse my dear Aunt Sally")

# PEMDAS



Parenthesis

Addition

Exponent



& Subtraction

## Strings



Double

Quotes

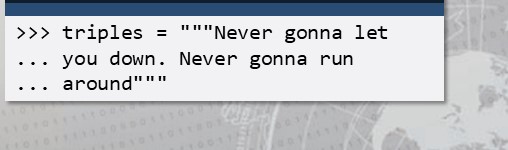
* Strings are immutable objects that hold character data
* Could be a single character, word, paragraph, multiple paragraphs, even zero characters
* Denoted by wrapping with:

(single quotes)

|  |  |
| --- | --- |
|  |  |
| second | give you up" |

(double quotes)

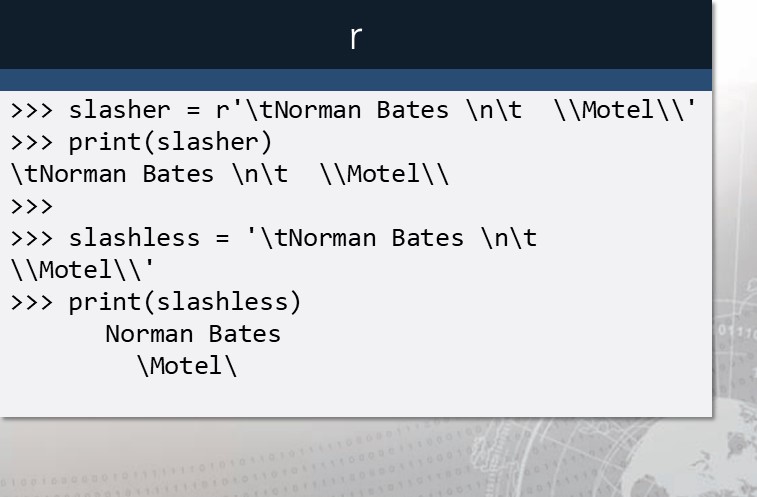
(triple doubles)

Triple Doubles  Usually used as a docstring (triple singles). 

### Strings: Escaping Characters

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Escape Sequence Output | | |  | | | \ "  \u12af  \U12af89bc    \084  \xFF | Backslash  Single quote  Double quote  ASCII Backspace  Tab  Unicode 16 bit  Unicode 32 bit Unicode character  Octal character  Hex character | |  |

### Strings: Avoid Escaping with r



•

Have

r

precede

a

string

to

turn

the

string

into

a

raw

string.

•

This

is

usually

done

with

regular

expressions

and

in

Windows

paths

where

the

backslash

is

the

delimiter.

#### Strings: Formatting Strings

• Use the .format method



Using .format

|  |
| --- |
| ' child: { } ' . format( ' Beyonce  ' child : Beyonce '  ' child: {name}' . format(name=' Kelly' ) ' child: Kelly'    ' child:  { [child]] ' .  child' : 'Michelle' } )  ' child: Michelle'  ' Last : {2}, First:  {0}' . format( ' Beyonce' , ' Kelly' , 'Michelle  ' Last: Michelle, First: Beyonce' |

'Kelly' is the value of the name.

'Michelle' is an indexed item in the 'child'  dictionary.

'Beyonce', is at position 0, the second,

'Kelly', is position 1 , and 'Michelle' is at 2.

### Strings: F-Strings

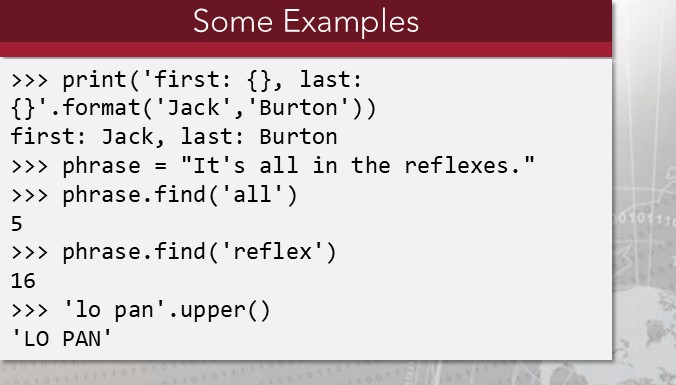


|  |
| --- |
|  |
| name = 'Marshall ' f ' Hi. My name is {name}'  'Hi . My name is Marshall' f ' Hi. My name is {name. lower( )} '  'Hi . My name is marshall ' |

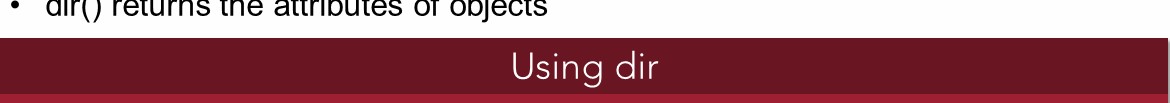
* F-strings introduced in Python 3.6 Using f-strings
* Allow for the inclusion of code in placeholders
* Placeholders can contain function calls, method calls, or any arbitrary code
* Potential for exploitation
* Reliance on input from outside of the original code



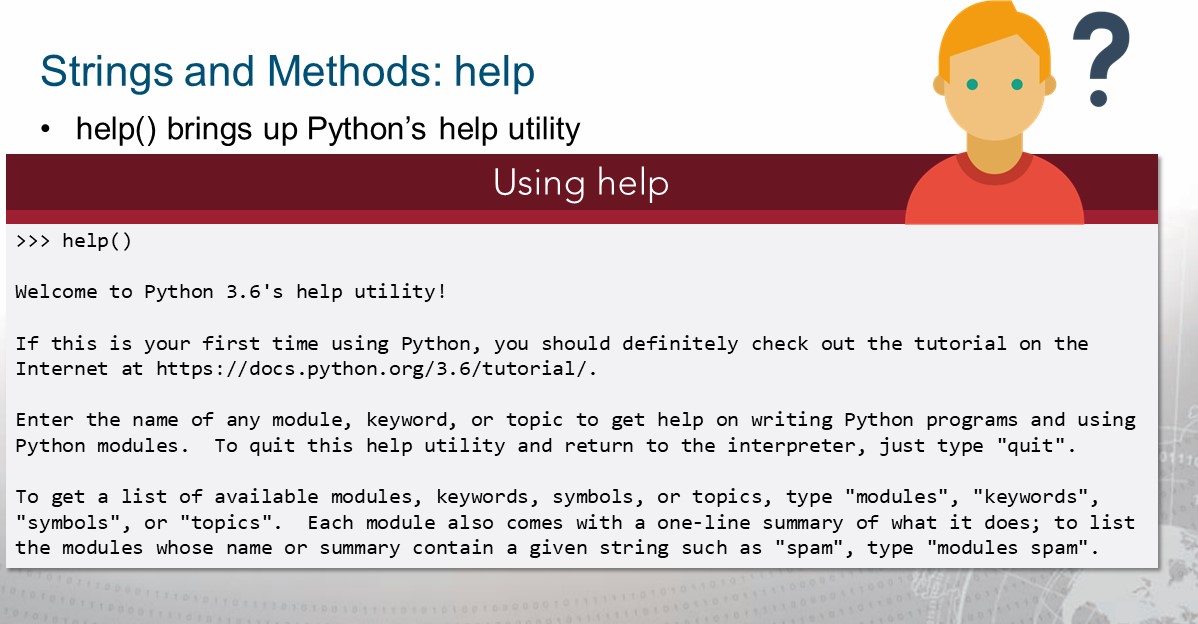
## Strings and Methods

* Because everything is seen as an object, each of the objects have methods.
* Invoking dir on an object will show the methods associated with the object.
* STRINGMETHODS entry in the help section has documentation and examples.

### Strings and Methods: dir

dir() returns the attributes of objects

|  |
| --- |
|  |
| setattr subclasshook  dir(phrase)  ' capitalize' , ' casefold' , ' center ' , ' count' , 'encode', 'endswith' , ' expandtabs ' ,  ' find ' , ' format ' , ' format\_map', ' index' , ' isalnum' , ' isalpha' , 'isdecimal', ' isdigit' ,  ' isidentifier' , ' is lower' , ' isnumeric ' ' isprintable ' ' isspace ' , ' istitle ' , ' isupper' ,  ' join ' 1 just ' , 'lower', ' Istrip ' , ' maketrans ' , ' partition' , ' replace', ' rfind ' ,  ' rindex', ' rjust' , ' rpartition' , ' rsplit' , ' rstrip' , ' split ' , ' split lines ' ,  ' startswith' , ' strip ' , ' swapcase' , 'title', ' translate' , ' upper ' , ' zfill' ] |



### Strings and Methods: help

help() also provides documentation for a method, module, class or function.

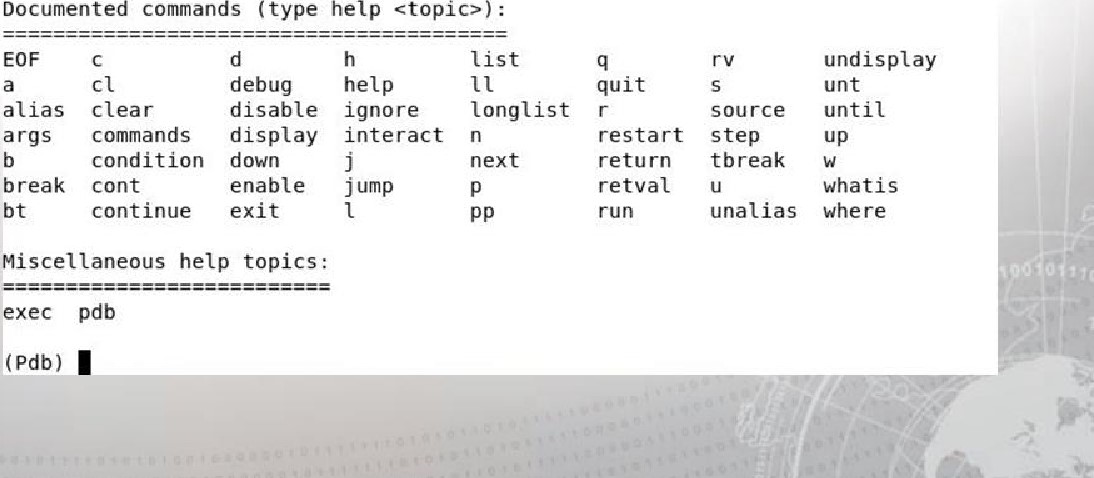
|  |
| --- |
| f ' HI . My name is {name. upper( )} '  'Hi. My name is MARSHALL |



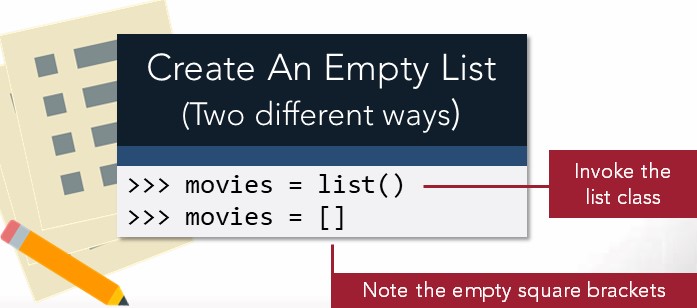
### pdb

* Built-in Python debugger

(Pdb) h



:

Lists

* Mutable
* Holds a list of objects  May hold any type of item, but it's good practice to hold a single type of item



|  |
| --- |
| movies = [ 'Ferris Beuller\'s Day Off', 'Gladiator' , 'Golden Child' ]  > > > print (movies)  [ 'Ferris Beuller's Day Off', 'Gladiator', 'Golden Child' |

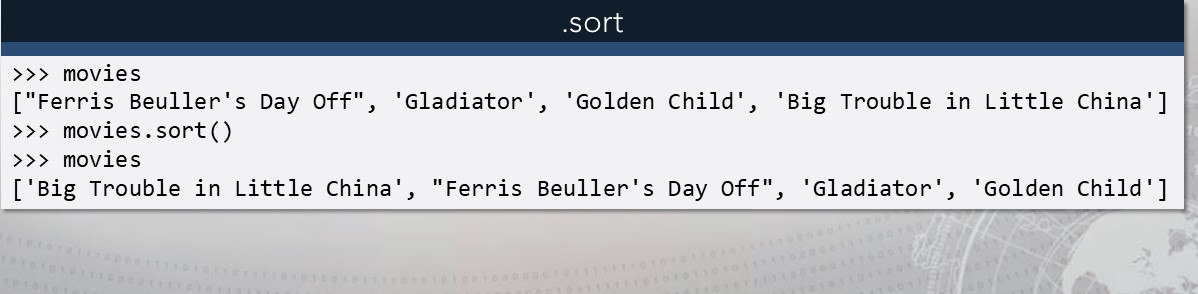
Lists



•

Use

modify



Create A Tuple With 0 Create A Tuple With 1

Items Item In It

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | empty = tuple()  - OR  empty    empty | |  |
| |  | | --- | |  | | one one  (1, ) one (1, ) #pythonic  one  (1, )    one 1,    one  (1, ) | |

### Tuples

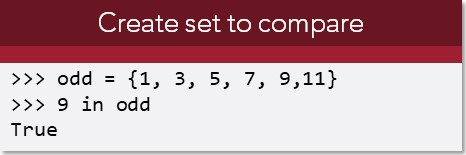
* Immutable
* Cannot use .append or

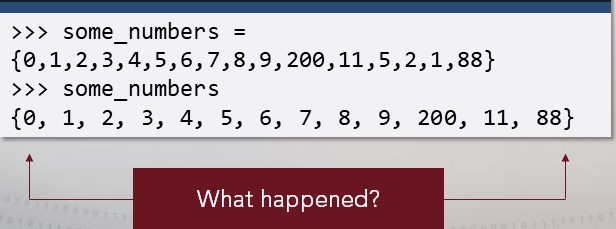
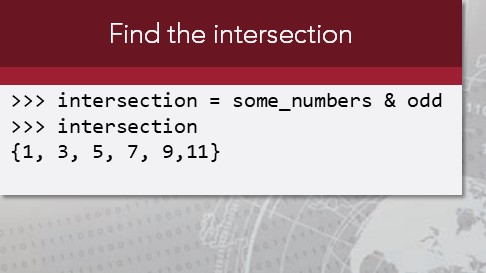
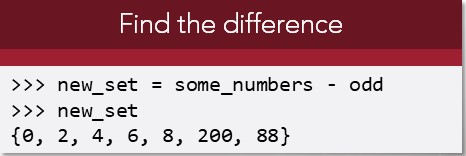
.remove methods

* Elements are enclosed in parentheses

|  |
| --- |
| Create A Tuple With Multiple |
|  |

* Boolean values
* An empty tuple is false
* Tuples with at least one item are true

Sets

* Immutable
* Unordered collection
* Cannot contain duplicates
* Does not care about order
* Good for removing duplicates and checking membership

#### Dictionaries

A Python dictionary contains one or more DICTIONARY

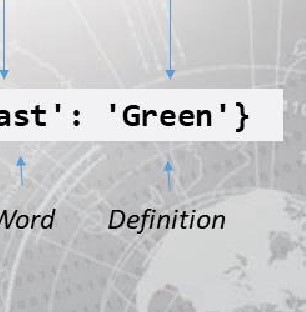
* Link a key to a value values known as keys [i.e. Baltimore], and
* Mutable each key is tied to a value [i.e. Orioles],

Multiple key-value pairs can be stored in a

* Do not copy the variable dictionary.
* Increase reference count to

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| the variable |  |  |  |  |
| Dictionary name | Key | Value | Key | Value |

friend = { 'first ' : 'Rachel',



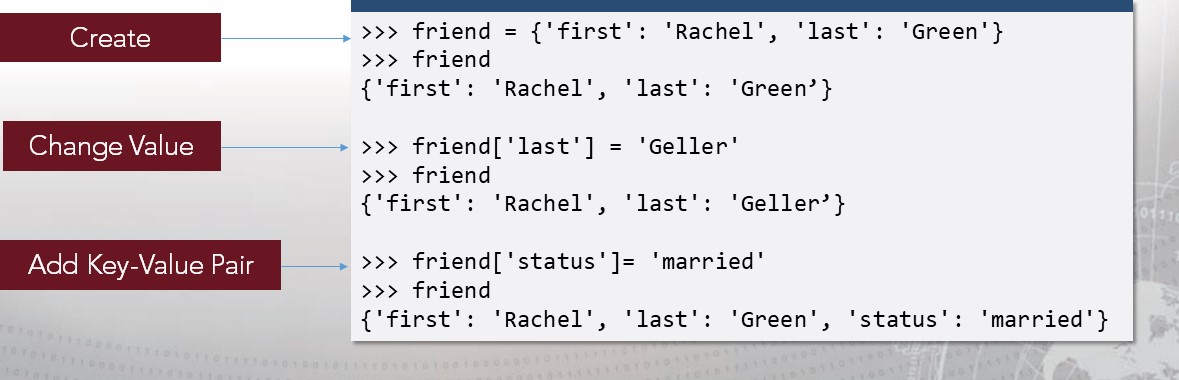
'last

Word

Webster's Dictionary Word Definition

#### Dictionaries

* Manipulating dictionaries
* No duplicate keys; assigning a value replaces the previous one

Change Value

### Exercise: Python Refresher I

Objectives

After completing this exercise, students will be able to:

• Create variables, strings, lists, tuples, sets and dictionaries in Python Manipulate variables, strings, lists, tuples, sets and dictionaries in Python

Duration

This exercise will take approximately 2.5 hours to complete, with 30-45 minutes to review answers.

#### Debrief: Python Refresher I — Variables

General Questions

* How did you feel about this procedure?
* Were there any areas in particular where you had difficulty?
* Do you understand how this relates to the work you will be doing?

#### Debrief: Python Refresher I — Variables

Specific Questions

* How can a variable type be determined in a Python interpreter?
* How can port\_ list be utilized in a script?
* Which of the following are correct naming conventions for variables? 123abc = 7 something with spaces = 42 pass 42 my\_variable = 42 my\_l\_kabob = 42 break = 42

#### Debrief: Python Refresher I — Numbers

General Questions

* How did you feel about this procedure?
* Were there any areas in particular where you had difficulty?
* Do you understand how this relates to the work you will be doing?

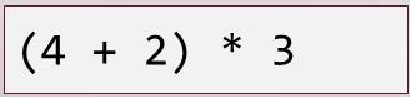
#### Debrief: Python Refresher I — Numbers

Specific Questions

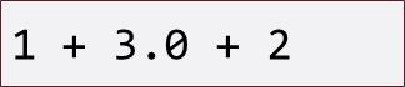
* What is the output of the following code?



* What is the output of the following code?



* What is the output of the following code?



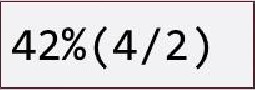
* What is the output of the following code?

|  |
| --- |
| 42  42.0  4/2 |

#### Debrief: Python Refresher I — Numbers

Specific Questions

* What code needs to be entered into the interpreter to figure out the 2 to tenth power?
* What code needs to be entered into the interpreter to turn 57.2 to a type int?
* What code needs to be entered into the interpreter to turn 57.2 to a type str?
* What is the output of the following code?



#### Debrief: Python Refresher I — Numbers

Specific Questions

* You tracked how many hours during a work week you played video games. On Monday you played 1 hour, Tuesday 3.5 hours, Wednesday 1.2 hours, Thursday 3.3 hours, and Friday 6 hours. What is the equation you would enter into the Python interpreter to determine the average number of hours you played video games?
* Dividing an integer by a float will result with what?
  1. An integer
  2. Afloat
  3. A SyntaxError

#### Debrief: Python Refresher I — Numbers

Specific Questions

• Which is the output of the following code?

|  |
| --- |
| width 5 height 10 width \* height |

1. 50
2. widthwidthheightheight

## c . 510

• What is the output of the following code?



### Debrief: Python Refresher I — Numbers

Specific Questions

* What is the output of the • What is the output of the following code? following code?

|  |  |
| --- | --- |
| 42  >>> | 40 + 2  result result |

|  |
| --- |
| 141 \* 2 |

|  |
| --- |
| keaton beetlejuice print ( keaton |

* What is the output of the following code?

### Debrief: Python Refresher I — Strings

General Questions

* How did you feel about this procedure?
* Were there any areas in particular where you had difficulty?
* Do you understand how this relates to the work you will be doing?

### Debrief: Python Refresher I — Strings

Specific Questions

* How can a variable classified as an integer be converted to a string?
* What is the output of the following code?

|  |
| --- |
| 55 . 5  ' string or integer/ print(x + ' ? ' ) |

### Debrief: Python Refresher I — Strings

Specific Questions

• Fill in the blanks to get the output:

## 1 fish 2 fish

|  |
| --- |
| print( |

• How can <string>.format() be used?

### Debrief: Python Refresher I — Strings

Specific Questions

* Why did print(jhopper. find( "are ) return a value of 9?
* On line 62 of the walk-through, the result of my\_docstring being printed out had characters that were not originally entered in the docstring. What is the purpose of those characters? Without making any modifications to the value of the variable, how can my\_docstring be printed without those characters displayed?
* What do Istrip() and rstrip() do?

### Debrief: Python Refresher I — Lists

General Questions

* How did you feel about this procedure?
* Were there any areas in particular where you had difficulty?
* Do you understand how this relates to the work you will be doing?

### Debrief: Python Refresher I — Lists

Specific Questions

* Where is an item placed when appended to a list?
* What occurs when a list is passed through len()?
* What occurs when a list that contains both integers and strings is sorted without any parameters?
* In Part 2 of the walk-through on line 30, explain what happened when port\_ list [e : 15 : 3] was

### Debrief: Python Refresher I — Lists

Specific Questions

* On line 37 through 39 in Part 2, after running port\_list . append (8080) the following output for print( ) does not show a increment in total ports scanned. Why does it still show 15 total ports?
* In Part 3 a variable some nums was set to the value of range(7). On line 17 in Part 3, some\_nums returned numbers 0 through 6. Why was 7 not included?
* In a follow up to question 6, why did some\_nums return a list that started with 0?

### Debrief: Python Refresher I — Lists

Specific Questions

* Why did result of line 19 kick back an error but line 24 did not?
* What occurs when you reference an index that does not exist in a list?
* Write the syntax that would print douglas from the following list.

|  |  |
| --- | --- |
| foo = | [ ' jordan % 'douglas% 'jackson' ] |

* How many items are in the following list:



### Debrief: Python Refresher I — Lists

Specific Questions

• Which lines of code will cause an error? (more than one answer possible) bar = [1307, 'concourse' , [3 ] , 'antoniovas ' , 42]

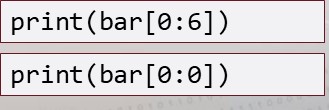
|  |  |  |
| --- | --- | --- |
| print(bar[2] ) |  | print(bar[1], [3]) |

|  |
| --- |
| print(bar[-2]) |

#### print(bar[l])

|  |
| --- |
| print (bar[1:2] ) |

print (bar[l, 3] )

print(bar[1],bar[3])

print(bar[5] )

### Debrief: Python Refresher I — Lists

Specific Questions

* What is the result of the following code?

|  |  |
| --- | --- |
| foo — rocky' , 'skye ' , ' rubble' , foo[l] = foo[3] print (foo[l] ) | marshal 1', 'chase ' ] |

* What is the result of this code and why?

|  |
| --- |
| nums = list (range (7, 11) ) print (len (nums)) |

### Debrief: Python Refresher I — Lists

Specific Questions

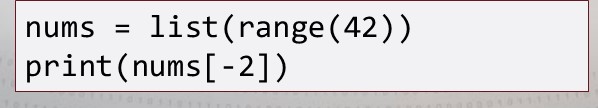
* What is the result of this code and why?

|  |
| --- |
| print(range(20) == range(e, 20) ) |

* What is the result of this code and why?

|  |
| --- |
| nums = list(range(42)) print (nums[4] ) |

* What is the result of this code and why?



### Debrief: Python Refresher I — Lists

Specific Questions

* What is the result of this code and why?

|  |
| --- |
| nums = list(range(42)) print (nums [42] ) |

* What is the result of this code and why?

|  |
| --- |
| nums —— list (range(3, 15, 3) ) print ( nums [ 2] ) |

### Debrief: Python Refresher I — Tuples & Sets (Solutions)

General Questions

* How did you feel about this procedure?
* Were there any areas in particular where you had difficulty?
* Do you understand how this relates to the work you will be doing?

Debrief: Python Refresher I — Tuples & Sets

(Solutions)

Specific Questions

* Which syntax returns a tuple?

|  |  |
| --- | --- |
|  |  |
| |  | | --- | | ' 3 ' | |
|  |
| |  | | --- | | ( (4\*2) , 'j ' , 37) | |



### Debrief: Python Refresher I — Tuples & Sets (Solutions)

Specific Questions

* What is the difference between a list and a tuple?
* What is the result of the following syntax?

|  |
| --- |
| ( 'jiff ' , 'peter pan  ( 'jiff ' , 'peter pan' , ' skippy ' ) pb. append( ' smuckers |

* What is the result of the following syntax?

|  |
| --- |
| china = print (china) |

### Debrief: Python Refresher I — Tuples & Sets (Solutions)

Specific Questions

• What is the value of porcelain after the following syntax?

|  |
| --- |
| china tea =    english\_tea =  porcelain - china tea - english\_tea |

### Debrief: Python Refresher I — Tuples & Sets (Solutions)

Specific Questions

• What is the value of the variable letters?

|  |
| --- |
| phoenician  alphabet - 'c'} letters - phoenician & alphabet |

Debrief: Python Refresher I — Dictionaries

### (Solutions)

General Questions

* How did you feel about this procedure?
* Were there any areas in particular where you had difficulty?
* Do you understand how this relates to the work you will be doing?

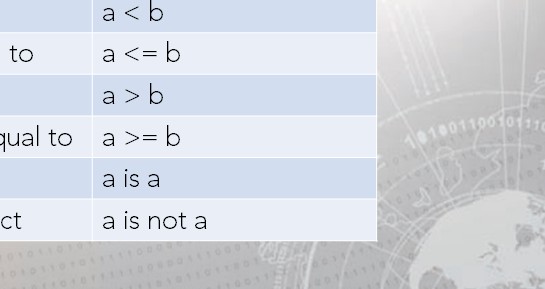
#### Debrief: Python Refresher I — Dictionaries

Specific Questions

* Is a dictionary mutable?
* Can a key in a dictionary hold two different values? What would occur if attempted?
* What can the in statement be used for?
* Can a value in a dictionary be held by different keys'?

##### Conditionals

|  |  |  |
| --- | --- | --- |
| • If this is true then do this | | • if statements |
| • Determine the truthiness of something | | • else statements |
| OperatorMeaning Example | | |

Equalsb

Not equals a != b

Less than

Less than or equal to

Greater than

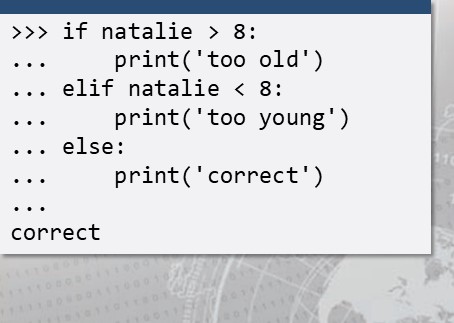
Greater than or equal to is Identical object is not Not identical object

Conditionals



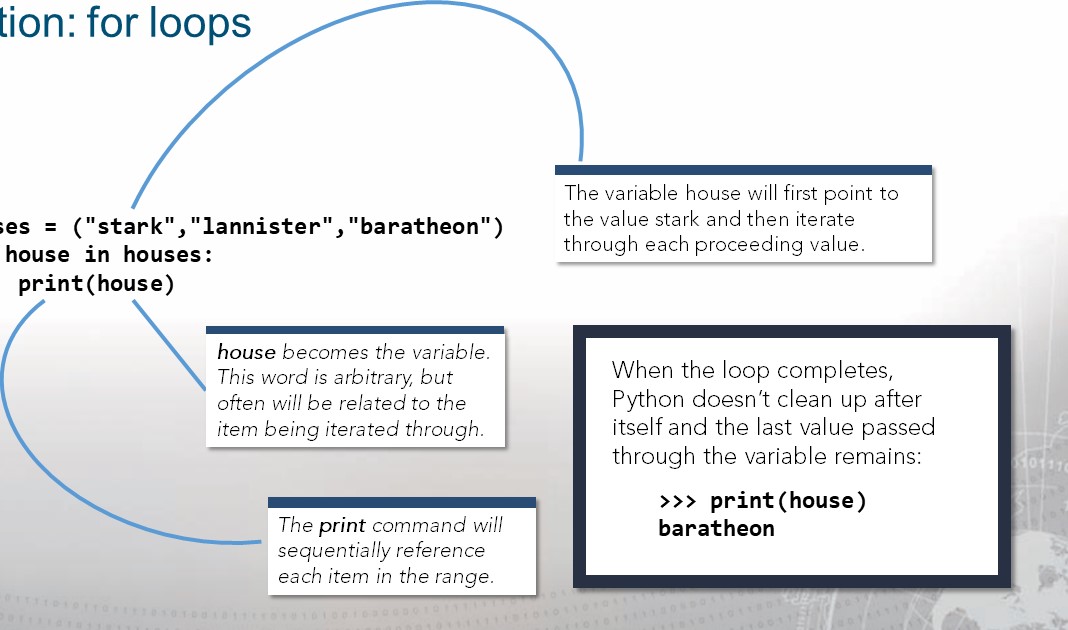
|  |  |  |
| --- | --- | --- |
| natalie = natalie =  True natalie =  True natalie  True    natalie >  False | 8  8 natalie  7  9  if  else:  her age | colon starts    code block  natalie print( ' her age  print( ' not her age |
|
|  |
| white space matters |

|  |  |
| --- | --- |
| •  • | Denoted with a colon Denoted with indentation  Whitespace matters |

Code Blocks

###### Iteration

• Being able to iterate (repeat) through objects



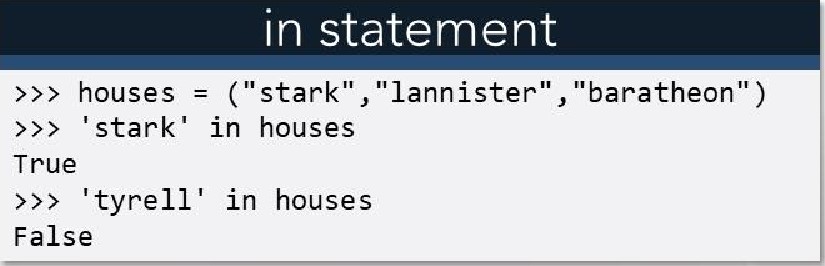
for

Iteration:

houses

###### Iteration: in statement

• The in statement can be used to check membership



###### Iteration: while statement

* loops over block of code while condition holds
* the condition can evaluate to True or False

|  |
| --- |
| >>> while i < 6 : print(i)  1  2  3  4  5  >>> print(i)  6 |

What is happening in this

example?

###### Iteration: break

• To break out of a loop, use the keyword break

|  |
| --- |
| Using break in a for loop |
|  |
| for i in range(9):  if i break print(i)  1  2  3 |

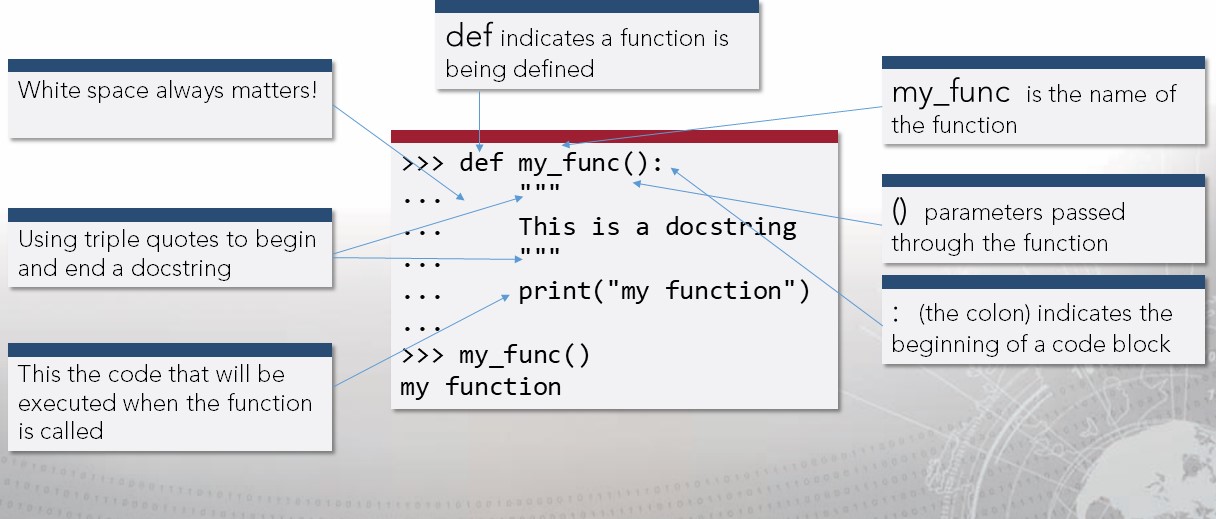
|  |
| --- |
|  |
| i=e while i < 6 : print(i) if i 3:  break |

Using break in a while loop

#### Functions

The key components of a function are:

* the def keyword (def defines that a function is being created)
* a function name (PEP8 says keep it lowercase, use underscores between words, can't be a key word, don't start with a number, don't override built-ins)
* function parameters parentheses (Any input parameters or arguments should be placed within these parentheses)
* a colon C)
* code blocks (white space matters)
* a docstring (optional)  a return statement (implied if not explicit) 



Functions

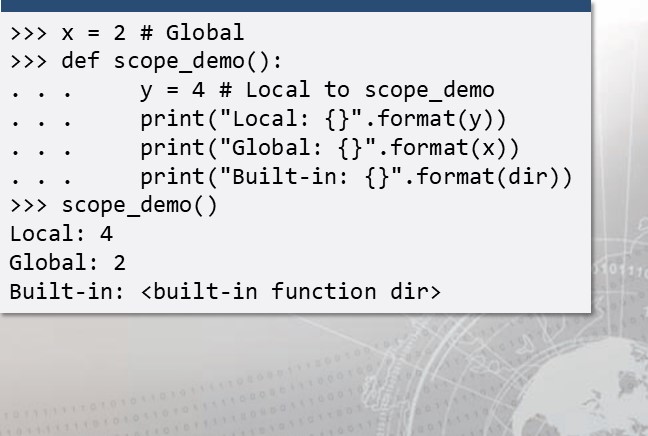
### Functions: Docstrings and help()

• After a docstring is written in a function, help() can be used to see what the author provided for the docstring of the function.

|  |  |  |
| --- | --- | --- |
| |  | | --- | | def my\_func() :  This is a docstring that will describe what my\_func    does, but the docstring is    optional    help(my\_func) | |  |
|  |

#### Functions: Scope

|  |  |
| --- | --- |
| Global Scope | Built-in Scope |
| Variablesthat are defined at the global level. | Variablesthat are predefined in Python. |

Local Scope Variablesthat are defined inside of functions.

##### Starting A Script

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | Which to use? | |  | | * #!/usr/bin/env python3 * #! /bin/python3 * #!/usr/bin/python * #!/usr/bin/pythonX.Y * #!/usr/bin/python3.3 * #!/bin/bash | |  |



### Exercise: Python Refresher I l

Objectives

After completing this exercise, students will be able to:

• Create conditionals, loops, and functions in Python • Manipulate conditionals, loops, and functions in Python

Duration

This exercise will take approximately 2 hours to complete.

#### Debrief: Python Refresher I l — Conditionals & Iterations

General Questions

* How did you feel about this procedure?
* Were there any areas in particular where you had difficulty?
* Do you understand how this relates to the work you will be doing?

Debrief: Python Refresher I l — Conditionals & Iterations

Specific Questions

* How many numbers • What does i equal after does this code print? the following code is ran?

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | i = 42 while print(i) i=i-l | | |  | | --- | | while i < 6 : print(i) | |

Debrief: Python Refresher I l — Conditionals & Iterations

Specific Questions

 What is the result of the • How many numbers following code? does the following code produce?

|  |
| --- |
| while i < 6 :  print(i) |

10 while True:

print(i) i=i-2 if i <=2: break

#### Debrief: Python Refresher I l — Conditionals & Iterations

Specific Questions

• What is the output of the following code?

|  |
| --- |
| concourse concourse [1] = concourse[2] -1 if 1 in concourse :  print (concourse[e]) else:  print (concourse[l] ) |

#### Debrief: Python Refresher I l — Conditionals & Iterations

Specific Questions

• What is the output of the following code?

|  |  |
| --- | --- |
| names - [ ' eric ' , ' karat ' ,  if ' kathy • in names:  print (names[l]) | ' kathy' ] |

#### Debrief: Python Refresher I l — Conditionals & Iterations

Specific Questions

• What is the output of the following code?

|  |
| --- |
| names = [ 'woody ' , ' travis ' , ' matt ' ] names. insert (1, ' jordan ' ) if ' jordan ' names [1] : print (names [2] ) |

I

### Debrief: Python Refresher I l — Functions

General Questions

* How did you feel about this procedure?
* Were there any areas in

particular where you had difficulty?

* Do you understand how this relates to the work you will be doing?

Specific Questions

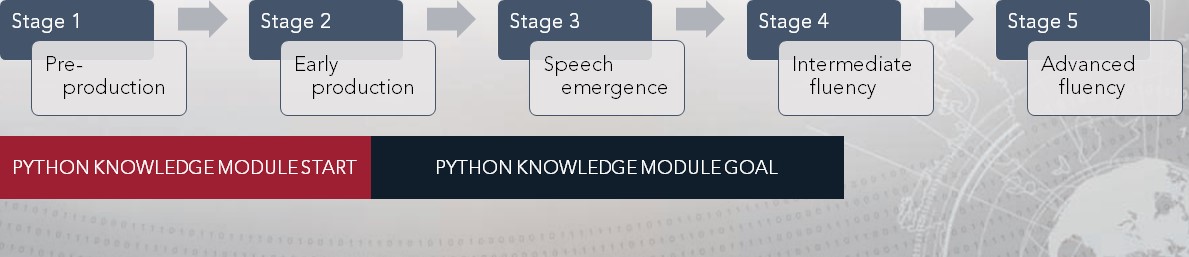
* What are the key elements to creating a function?

#### Debrief

Python is a language... and learning a new language can be intimidating

* Continue to solidify your knowledge
* Engage your classmates and help one another



PYTHON KNOWLEDGE MODULESTART PYTHON KNOWLEDGE MODULE GOAL

Resources

For additional information on and practice with using Python, explore the following suggested resources:

* "Illustrated Guide to Python 3" by Matt Harrison
* Michael Kennedy https://blog.michaelckennedy.net/category/python/
* "Effective Python Penetration Testing" by Rejah Rehim
* "Violent Python" by TJ. O'Connor
* Contributors to https://www.w3schools.com
* "Python Tricks The Book" by Dan Bader  https://python.org

### Lesson Summary

In this lesson we learned how to:

* Manipulate variables, strings, lists, dictionaries, conditionals, loops, and functions in Python
* Create variables, strings, lists, dictionaries, conditionals, loops, and functions in Python



|  |
| --- |
| End of Module 2, Lesson  9 |