Module 7: Introduction to Python

LEARNING THE PYTHON BASICS

```
#!/bin/python3

#Math
print(50+50) #add
print(50-50) #subtract
print(50*50) #multiply
print(50/50) #divide
print(50+50-50*50/50) #pemdas
print(50 ** 50) #exponents
print(50 % 6) #modulo
print(50/6)
print(50//6) #no leftovers
```

```
#Print string
print ("Hello, World!") #double quotes
print('\n') #newline
print ('Hello, World!') #single quotes

print ("""This string runs
multiple lines!""") # triple quote for multiline

print ("This string is " + "awesome!") # concatenation
```

```
#Variables and Methods
quote = "All is fair in love and war."
print(quote.upper()) #method for making quote uppercase
print(quote.lower()) #method for making quote lowercase
print(quote.title()) #title case

print(len(quote)) #gives us all characters (length).

name = "Kevin" #string
age = 29 #int (integer, has no decimal point)
gpa = 3.7 #float (has decimal point)
```

```
print (int(age)) #Integer does not round
print (float(29))

#print ("My name is " + name + " and I am " + age + "years old.")
#cannot concat a string (above^) with an integer fix is below
print ("My name is " + name + " and I am " + str(age) + "years old.") #fix by str(age)

age += 1 #adding a year to our age
print(age)

birthday = 1
age += birthday #adding variable "birthday" to 'age"
print(age)
```

```
#Variables and Methods
quote = "All is fair in love and war."
print(quote.upper()) #method for making quote uppercase
print(quote.lower()) #method for making quote lowercase
print(quote.title()) #title case

print(len(quote)) #gives us all characters (length).

name = "Kevin" #string
age = 29 #int (integer, has no decimal point)
gpa = 3.7 #float (has decimal point)

print (int(age)) #Integer does not round
print (float(29))

#print ("My name is " + name + " and I am " + age + "years old.")
#cannot concat a string (above^) with an integer fix is below
print ("My name is " + name + " and I am " + str(age) + "years old.") #fix by str(age)

age += 1 #adding a year to our age
print(age)

birthday = 1
age += birthday #adding variable "birthday" to 'age"
print(age)

print('\n')
```

```
#Functions
print ("Here is an example function")

def who_am_i(): #this is a function
name="Kevin"
age=29
print ("My name is " + name + " and I am " + str(age) + "years old.")

who_am_i()
```

if we type print(age) now, it will pop up as 31 because the defined age of

```
#29 only exists within the above function. Whereas the age += is before the
#funtion.
```

```
#adding parameters
def add_one_hundred(num): #num=number
print(num + 100)

add_one_hundred(100)
```

```
#multiple parameters
def add(x,y):
print(x+y)

add (7,7)
```

```
#more mini programs
def multiply(x,y):
return x * y #doesnt print to the screen. only returns the number

multiply
#or
print(multiply(7,7)) #call to print later after the return

def square_root(x):
print(x ** .5) # ** is for exponents

square_root(64)
```

```
#creating a new line

def new_line():
print('\n')

new_line()
```

```
#Boolean Expressions (True or False)
print("Boolean Expressions:")

bool1 = True
bool2 = 33 == 9
bool3 = False
bool4 = 33 != 9

print (bool1,bool2,bool3,bool4)
print(type(bool1))
```

```
#Relational and Boolean operators
greater_than = 7 > 5
less_than = 5 < 7
greater_than_equal_to = 7 >= 7
less_than_equal_to = 7 <= 7

test_and1 = (7 > 5) and (5 < 7) #True
test_and2 = (7 > 5) and (5 > 7) #False
test_or1 = (7 > 5) or (5 < 7) # True
test_or2 = (7 > 5) or (5 > 7) #True

test_not = not True #False
test_not = not False #True
#Look up Truth Tables in Python
```

```
#Conditional Statements
new_line()

def drink(money):
if money >= 2:
return "You've got yourself a drink!"
```

```
else:
return "No drink your you!"

print(drink(3))
print(drink(1))

def alcohol(age,money):
if (age >= 21) and (money >= 5):
return "Were getting a drink!"
elif (age >= 21) and (money < 5):
return "Come back with more money."
elif (age < 21) and (money >= 5):
return "nice try, kid!"
else:
return "youre too poor and too young"

print(alcohol(21,5))
print(alcohol(21,4))
print(alcohol(20,4))
```

```
#Lists (Lists have brackets [])
new_line()
print('\n')

movies = ["When harry met sally", "The hangover", "The perks of being a wallflower","the exorsist"]

print(movies[1])#returns 2nd item in the list
print(movies[0]) #returns first item in the list
print(movies [0:4])#prints all items in the list
print(movies[1:]) #Grabs all items in the list after one.
print(movies[:1])#Grabs all items in the list before 1
print(movies[-1])#Grabs the last items on the Lists

print(len(movies))#counts the items in the Lists

movies.append("JAWS") #adds item to the end of the List
print(movies)

movies.pop()#deletes the last items on the List
print(movies)

movies.pop(0)#deletes the first items in the List
print(movies)

print('\n')
```

#Tuples Do not change (Tuples have Parenthesis, and cannot be changed once defined)

```
grades = ("a","b","c","d","e")
print(grades[1])

print('\n')
```

#Looping

```
#For loops - start to finish of an iterate
vegetables = ["cucumber", "spinach", "cabbage"]
for x in vegetables:
print(x) #prints each item on the list. Iterates through the list.
```

```
#while loops - execute as long as they are true.

i = 1

while i < 10:
print(i)
i += 1 #loops and prints 1-9
new_line()
new_line()
new_line()
new_line()
new_line()
################## NEW SECTION, NEW SCRIPT #################

#sys - deals with anything related to system funtions and parameters.
#There are modules in python that need to be imported

import sys #imports the module "sys" that is needed to print "sys.version"
print(sys.version)

from datetime import datetime #imports a specific thing from the module "datetime"
print(datetime.now())

new_line()
from datetime import datetime as dt #creats an alias from imported modules
#and imports module with alias
```

```
#Ex.
print(dt.now())
```

```
new_line()
#Advanced Strings
```

```
my_name = "Kevin"
print(my_name[0])#prints first letter of my_name
print(my_name[-1])#prints last letter of my_name

sentence = "This is a sentence"
print(sentence[:4]) #prints the word "This" but you need to know the amount of
#characters

print(sentence.split()) #splits based on a delimiter

sentence_split = sentence.split()

sentence_join = ' '.join(sentence_split) #joins a sentence and adds our own
#delimiter
print(sentence_join)
```

quote = "He said, 'give me all your money'" #single quotes inside of double quotes

```
#allows quotes in your Strings
quote = "He said, "give me all your money"" # forward slash allows "character escaping"
print(quote)

too_much_space = " hello "
print(too_much_space.strip()) #removes extra spaces by using .strip()

print("A" in "Apple") # looks for A in the word apple, but it is case sensitive.

letter = "A"
word = "Apple"
print(letter.lower() in word.lower()) #improved by making everything lowercase and then searching for
"A"

movie2 = "The Hangover"
print ("My favorite movie is {}.".format(movie2)) #works as a placeholder {} with .format()
#placeholding is better than concatenation
```

#Dictionaries - key/value pairs uses curly braces {}
new_line()
drinks = {"White Russian": 7, "Old Fashion": 10, "Lemon Drop": 8}
print(drinks) #drink is the key, price is the value "key/value pairs"

#you can have multiple values to a key
employees = {"finance": ["bob", "Linda", "Tina"], "IT": ["Gene", "Louise", "Teddy"], "HR": ["Rita", "Margaret"]}
print(employees)

#Adding a new Key/Value Pair (new department called "legal")
employees["legal"] = ["Mr.Frond"]
print(employees)

#or
new_line()

employees.update({"sales": ["andie", "ollie"]})
print(employees)

#update drinks
drinks["White Russian"] = 8
print(drinks) #updates price of white Russian

print(drinks.get("White Russian")) #pulls from a dictionary

SOCKET SCRIPT

#!/bin/python3

#A socket is used to connect two nodes together. To connect to an open port and IP address
#This is a simple socket script and port connection
import socket

HOST = '127.0.0.1'
PORT = 7777

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #we want s to equal this
#AF_INET = ipv4
#sock_stream = port
s.connect((HOST,PORT))

#use netcat = nc
#listening port = -nvlp
```

nc -nvlp 7777

PORT SCANNER SCRIPT

```
#!/bin/python3

import sys
import socket
from datetime import datetime

if len(sys.argv) == 2:
target = socket.gethostbyname(sys.argv[1])
else:
print("Invalid amount of arguments.")
print("syntax: python3 scanner.py ")

print("-" * 50)
print("Scanning Target: "+target)
print("Time started: "+str(datetime.now()))
print("-" * 50)

try:
for port in range(50,85):
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
socket.setdefaulttimeout(1)
result = s.connect_ex((target,port))
if result == 0:
print("port {} is open".format(port))
else:
```

```
print("port {} not open".format(port))
s.close()

except KeyboardInterrupt:
print("\nExiting Program.")
sys.exit()

except socket.gaierror:
print("Hostname could not be resolved")
sys.exit()

except socket.error:
print("Couldnt connect to server.")
sys.exit()
```