PYTHON PROGRAMMING TUTORIAL – 2 –NUMBERS:

+ addition  
- subtraction  
\* multiplication  
/ division (result is floating point number)  
Python has order of regular operations. 8 + 2 \* 10 = 28 vs (8 + 2) \* 10 = 100  
// integer division rounds it down to lower number  
% „modulus”, gets only the remainder  
\*\* calculate powers 5 \*\* 5 = 25

You can have variables:  
start it with a letter and later you can have numbers in it too.  
>>> tuna = 5  
variable tuna equals 5.  
You can add numbers + variables:   
>>> 20 + tuna  
25  
>>> bacon =18  
>>> bacon / tuna  
3.6

STRINGS:

A text in programming. You gotta include it in quotation marks – double or single.  
>>> ’I don’t think she’s 18’  
If you have a sentence that uses single quote, put the whole sentence in double quotes and vica versa.  
>>> „I don’t think she’s 18”  
„I don’t think she’s 18”  
>>> ’She said, „What part of the cow is the meatloaf from?” ’  
’She said, „What part of the cow is the meatloaf from?”

*Escaping a character* whenever python thinks the string came to the end, use the \’ escape character:  
>>> ’I don\’t think she’s 18’  
„I don’t think she’s 18”

To print sg out on the screen:  
>>>print(„Hey now brown cow”)  
hey now brown cow

\n means go to a new line:  
>>>print(C:\User\Vera\nevermind’)  
C:\User\Vera  
evermind  
When you wanna print out a raw string, put r before a string, so it wouldn’t use escape characters:  
>>>print(r’C:\User\Vera\nevermind’)  
C:\User\Vera\nevermind

You can add together strings:  
>>> firstname = „Rocky”  
>>>firstname + „Balboa”  
’Rocky Balboa  
>>>firstname + „McLovin”  
’Rocky McLovin’  
>>>firstname \* 5  
’Rocky Rocky Rocky Rocky Rocky’

SLICING UP STRINGS:

>>>user =”Tuna McFish”  
Computers start counting from zero!  
>>>user[0]  
T  
>>>user[-1]  
h  
>>>user[-3]  
i  
>>>user[2:7]  
’na Mc’  
It doesn’t involve the seventh character.  
>>>user[:7]  
’’Tuna Mc’  
>>>user[2:]  
’na McFish’  
>>>user[:]  
’Tuna McFich’  
>>>len(’afhhfhff’)  
8  
>>>len(user)  
11  
Counts spaces too

LIST:

>>>players = [21, 58, 66, 71, 87]  
>>>players[2]  
66  
>>>players[2] =68  
[21, 58, 68, 71, 87]  
>>>players + [90, 91, 98] -> this statement will not modify the original list  
[21, 58, 68, 71, 87, 90, 91, 98]  
>>>players  
[21, 58, 68, 71, 87]  
>>>players.append(120) ->modifies permanently  
[21, 58, 68, 71, 87, 120]  
>>>players[:2]  
[21, 58]  
>>>players[:2] = [0, 0] ->changing multiple items  
>>>players  
[0, 0, 68, 71, 87, 120]  
>>>players[:2] = [] ->removing multiple items  
[68, 71, 87, 120]  
>>>players[:] = [] -> deleting a lost  
>>>players  
[]

IF ELIF ELSE:

age = 27

if age < 21:  
 print(„No beer for you”)

name = „Lucy”

if name is „Bucky”:  
 print(„Hey there Bucky”)

elif name is „Lucy”:  
 print(„What up Lucy?”)

elfi name is „Samy”:  
 print(„Yo Samy”)

else:  
 print(„Please sign up for the site”)

What up Lucy?

FOR LOOPS:

foods = [’bacon’, ’tuna’, ’ham’, ’sausages’, ’beef’]

for f in foods:  
 print(f)  
 print(len(f))

bacon  
5  
tuna  
4  
ham  
3  
sausages  
8  
beef  
4

for f in foods[:2]: -> loop through the items, stop at position no.2  
 print(f)  
 print(len(f))

bacon  
5  
tuna  
4

RANGE AND WHILE:

for x in range(10): ->loop 10 times x is equivalent to 0 to 9  
 print(’lol’)

lol  
lol  
lol  
lol  
..\* 4  
lol

for x in range (5, 12):  
 print(x)

5  
6  
7  
..  
11

for x in range (10, 40, 5):  
 print(x)

10  
15  
20  
25  
..  
35

while loop goes on until the test equals false.

buttcrack = 5

while buttcrack < 10:  
 print(buttcrack)  
 buttcrack += 1 -> first prints five then adds 1 to it at every iteration up until it reaches 10.

COMMENTS AND BREAK:

Commenting in Python:

* on a single line: # this is a comment
* on multiple lines: ’’’ this is  
   a multiline comment ’’’

magicNumber = 26  
print(’Bucky’ + 9) -> ERROR. cannot add strings to numbers. Plus is only for adding to things of the same data type together.  
print(9, ’Bucky’) -> gonna print 9 Bucky

for n in range(101):

if n is magicNumber:  
print(n, „ is the magic number”)

Gonna print 26 is the magic number

for n in range(101):

if n is magicNumber:  
 print(n, „ is the magic number”)  
 break ->stop the loop

else:  
 print(n)

Gonna print every number up until 26. after it print 26 is the magic number and stops.

GETTING USER INPUT:

x = input(’What is your name? ’) -> put a space at the end so it’s pretty  
print(’Hello’, x)

INPUT FUNCTION

prompts the user to enter some unique information.  
>>> a = input()  
100  
>>>a  
’100’ - > this is a string  
>>>b = input(’Enter your name: ’)  
Enter your name: badass  
>>> b  
’badass’  
>>> c = int(input(’Enter a number: ’))  
Enter a number: 25  
>>> c  
25 -> this is an integer  
>>>c +5  
30  
>>> d = float(input(’enter any number: ’))  
enter any number 123.5  
>>> d  
 123.5  
>>>123.5 + c  
148  
>>> e = int(input(’enter an integer: ’))  
enter an integer: 123.4  
ERROR

DAY 03:  
FUNCTIONS:  
def beef():  
 print(„Dayum, functions are cool!”)

Whenever you want to use it, type:  
beef()

Functions are badass for reusability.

def bitcoin\_to\_usd():  
 amount = btc \* 527  
 print(amount)

beef()  
bitcoin\_to\_usd(3.85)

Console:  
Dayum, functions are cool!  
2023.95

RETURN VALUES:

return is useful for storing information (in cases we don’t simply need printing straight away).

def allowed \_dating\_age(my\_age):  
 girls\_age = my\_age / 2 + 7  
 return girls\_age

buckys\_limit = allowed\_dating\_age(27) ->nothing prints  
creepy\_joe\_limit = allowed\_dating\_age(49)  
print(„Bucky can date girls”, buckys\_limit, „or older”)  
print(„Creepy Joe can date girls”, creepy\_joe\_limit, „or older”)

DEFAULT VALUES FOR ARGUMENTS:

def get\_gender(sex=’Unknown’):  
 if sex is ’m’:  
 sex=”Male”  
 elif sex is ’f’:  
 sex= „Female”  
 print(sex)

get\_gender(’m’)  
get\_gender(’f’)  
get\_gender()

Console:  
Male  
Female  
Unknown

VARIABLE SCOPE:

a = 726

def corn():  
 print(a)

def furdge():  
 print(a)

Console:  
corn()  
fudge()

If the variable is created outside the function and above it, the function can access the variable.

let’s say, variable a is created inside function corn():

def corn():

a = 726  
 print(a)

Console:  
726 ->function corn prints it out  
ERROR on calling function fudge() and jumping back to the print statement of function fudge()  
„NameError: name ’a’ is not defined

Whenever a variable is created inside a function, only that function can access it.

KEYWORD ARGUMENTS:  
def dumb\_sentence(name=’Bucky’, action=’ate’, item=’tuna’):  
 print(name, action, item)

dumb\_sentence()  
dumb sentence(„Sally”, „farts”, „gently”)

CONSOLE:  
Bucky ate tuna

When we don’t pass in any argument, function uses default arguments. Wheneer we’re passing parameters, it’s gonna accept them in the order we pass them in.  
Whenever we wanna pass in a limited amount of arguments or pass them in a limited order, we’ll use key words.:

1. dumb\_sentence(item=’awesome’) ->On the console, it prints Bucky ate awesome
2. dumb\_sentence(item=’awesome’, action=’is’) ->Bucky is awesome