17 Functions and string methods

Monday, May 9, 2022 10:03 PM

Announcements

Tuesday, November 1, 2022 11:10 PM

November Is Native American Heritage Month!

November is **Native American Heritage Month (NAHM)**, and all month long we are celebrating the history and contributions of Native American and Indigenous people throughout the United States. Check out **Western's NAHM website**for a full list of events and programs!

Native American Heritage Month Events:

Staff Picks Library Display
November 1

"The Healing Heart of the First People of This Land"

Hacherl Research & Writing Studio

MCC Lunch & Learn with Dr. Brandon Joseph
November 2 @ 12 PM
MCC Multi-use Space (VU 735)

MCC Educational Session with Dr. Anna Lees
November 8 @ 3 PM
MCC Multi-use Space (VU 735)

November 9 @ 7:30 PM
PAC Concert Hall

MCC Lunch & Learn with Dr. Natalie Welch
November 16 @ 12 PM
VU 565A

MMIW Discussion
November 21 @ 10 AM
MCC Multi-use Space (VU 735)

More NAHM @ WWU Events + Info

How to function

Monday, February 14, 2022 1:08 PM

L-15-2

How to function

① list

1. evaluate all arguments
2. Draw a local "box" for local variables
3. Assign argument values to parameter variables inside box
4. Execute function body

How to return → 2 things? wait!

for all variables look local 1st, Global second

5. Return and erase box

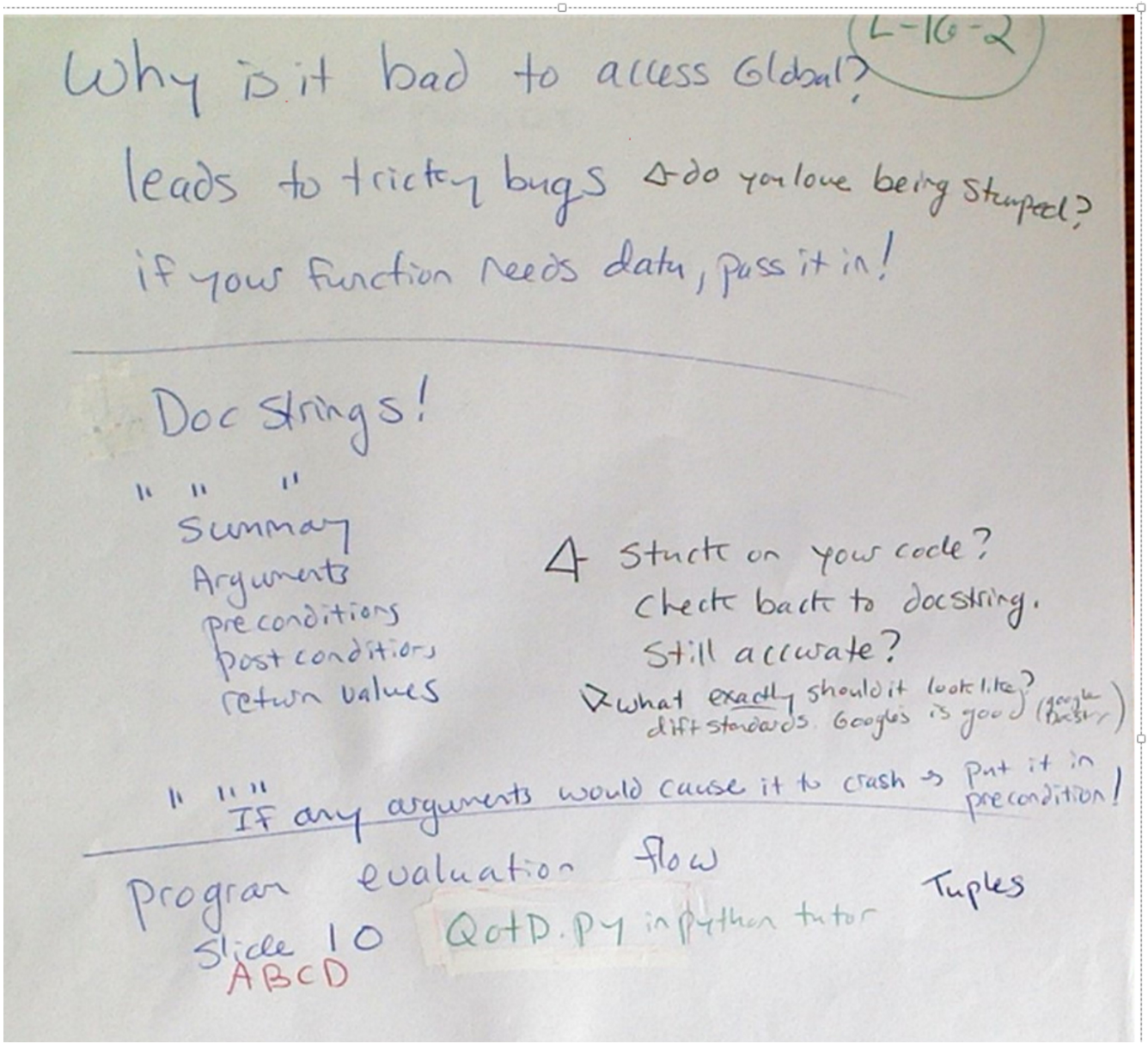
② example

x1 = 2
x2 = 4
y1 = -3
y2 = 7
num = 2

```
def midxy(x1,x2, y1, y2):
    midpointx= (x1+x2) /num
    midpointy= (y1+y2)/num
    return midpoint x
    return midpoint y
midxy(x1,x2, y1, y2)
```

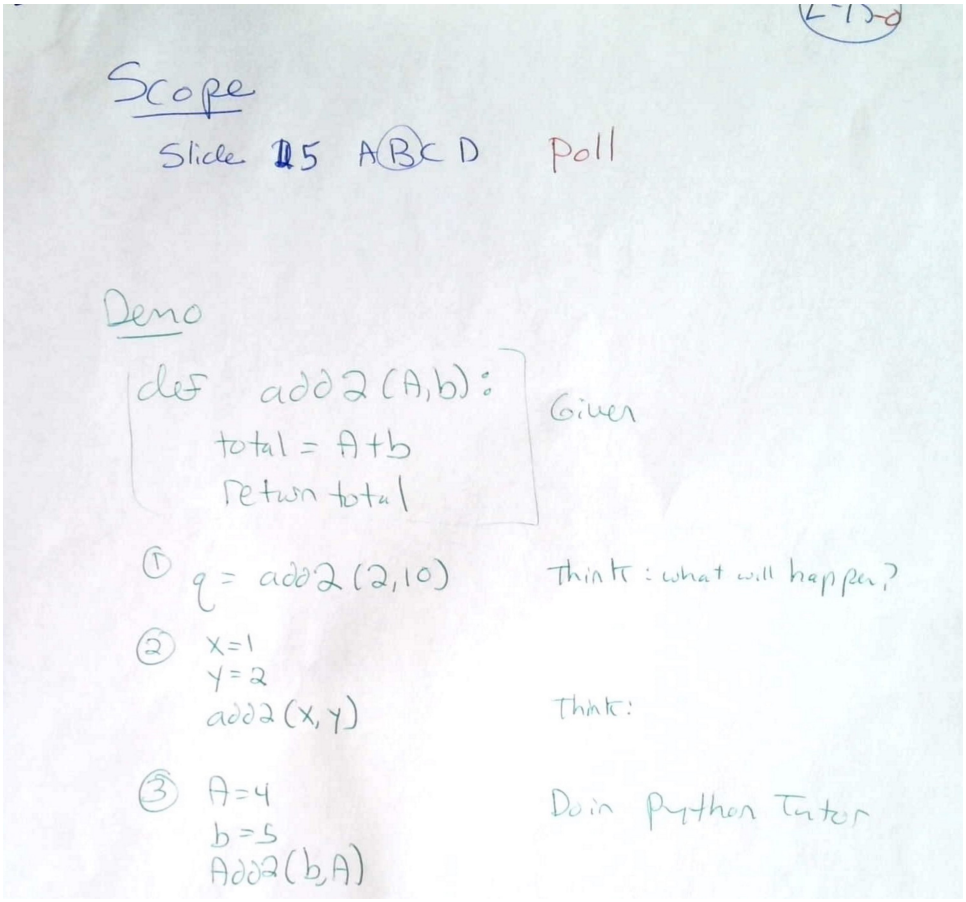
Doc Strings

Tuesday, April 26, 2022 10:22 PM



Scope

Tuesday, April 26, 2022 10:14 PM



```
2  def add2(a,b):
3      a = a+2 # 4
4      b = b+2 # 12
5      total = a+b
6      return(total)
7  a = 2
8  b = 10
9  w = add2(a, b)
10 print(a, b)
11 #what is in a and b on line 10
12 #A 2 and 10
13 #B 4 and 12
14 #C "a" and "b"
15 #D none of the above
```

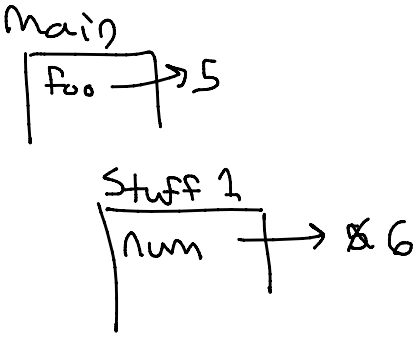
Mutability and functions

Friday, March 4, 2022 12:29 AM

var 1 → an object
var 2 → object ✓

```
def Stuff1(num):
    num = num + 1

foo = 5
Stuff1(foo)
```

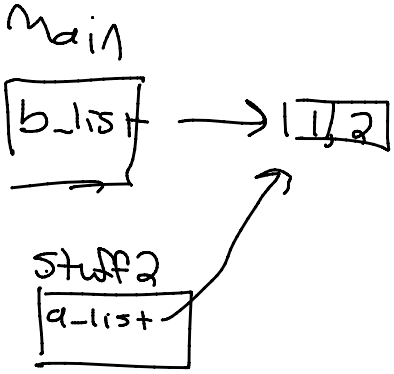


immutable objects obey scope

Mutable objects don't

```
def Stuff2 (a_list)
    a_list.append ("x")

b_list = [1, 2]
Stuff2 (b_list)
```



String enumeration

Thursday, March 4, 2021 2:02 PM

Demo

```
1  a_str = ["one", "two", "three", "four"]
2  #if we have a list of strings,
3  #we can turn it into a single string
4  #with .join
5
6  a_str = " ".join(a_str)
7
8  # but how do we do this if we
9  #are starting with a list of
10 #ints?
11 b = [0,1,2,3,4]
12
13 #" ".join(a) #nope error
```

breakout rooms: how turn a list of ints into a list of strings?

```
15 #we could use a loop!
16 for i in range(len(b)):
17     b[i] = str(b[i])
18
19 b_str = " ".join(b)
20
21 c = [0,1,2,3,4]
22 #but dang that was kinda a messy loop
23 #enumerate makes this easier:
24 for i, v in enumerate(c):
25     c[i] = str(v)
```

String Methods

Monday, February 21, 2022 9:59 PM

answer = input("What do?").lower()

Try!

ythonScratchpad.py *

```
1  address = "516 High Street Bellingham, WA 98229"
2
3  #print it all upper case
4
5  #print it all lower case
6
7  #remove any spaces
8
9  #Count how many letter 'A's there are
10
11 #replace 'WA' with "Washington"
12
13 #find the '9822' then make sure following digit is '5'
14
15
```

```
1  a = "quick brown fox jumps over the lazy dog"
2  b = a.upper() #makes it all upper case
3  c = a.lower() #makes it all lower case
4
5  d = a.replace("lazy", "!@#@")
6  #replace finds the first argument and replaces it with
7  #the second argument
8  print(d)
9
10 f = a.find("o")
11 print("f:",f)
12 #find finds the first instance of the string
13 #and returns the index
14 #want to find the next instance?
15 #you can specify where you start looking
16 g = a.find("o", 9)
17 print("g:",g)
```

```python
c = input("color?")
#type "    red   "
#users type extra spaces all the time
print(c)

#we dont' want that because then
#when we try to do comparisons

if c == "red":
    print("yay red!")

#it doesn't work

c_fixed = c.strip()
#so strip removes the leading and trailing whitespace
#(you can pass in something to remove something else,
#but this is the most common use)

print(c)
print(c_fixed)
```

```
z = " " #could do a variable
a = z.join(l_o_s)

#but normally just put the divider string like this
a = " ".join(l_o_s)

print(a)

#for split we usually use a variable because
#typically we have the string from something like
#input
a = input("Tell me a story")
b = a.split(" ")
print(b)

#but you could do it like this too
b = "Tell me a story".split(" ")

#takeaway: split and join are methods which belong to
#strings. So ask a string to do it, and save the result
#because they return strings, they don't change strings
```

Exercise 1: Write a function to print a given string with all vowels removed.

```
def remove_vowels(string):
    """ Print string, but with no vowels.
        Don't count y as a vowel. """
```

Exercise 1: Write a function that returns its argument string, but with any text after (and including) a # symbol removed.

Exercise 1 Answer

```
def remove_comments(string):
    """ Return a copy of string, but with
        all characters starting with and following
        the first instance of '#' removed. If there
        is no # in the string, return input unchanged. """
```

Exercise 2: The code the box below sets `word` to the string "BellinghaM", where the last letter, M, is now capitalized. Write a function that does the same thing to any input using a loop . The function should work on any length input, and should return the resulting string.

Exercise 2: Demo Code

```
word = "Bellingham"
word = word[:9] + word[9].upper()
```

Exercise 2 Function

```
def capitalize_last(in_str):
    """ Return a copy of in_str with its last character
        capitalized. Precondition: last character is a letter.
    """
```

Exercise 3: Rewrite the function from Exercise 1 without using a loop.

Exercise 3 Code

```
def remove_comments(string):
    """ Return a copy of string, but with
        all characters starting with and following
        the first instance of '#' removed. If there
        is no # in the string, return input unchanged. """
```