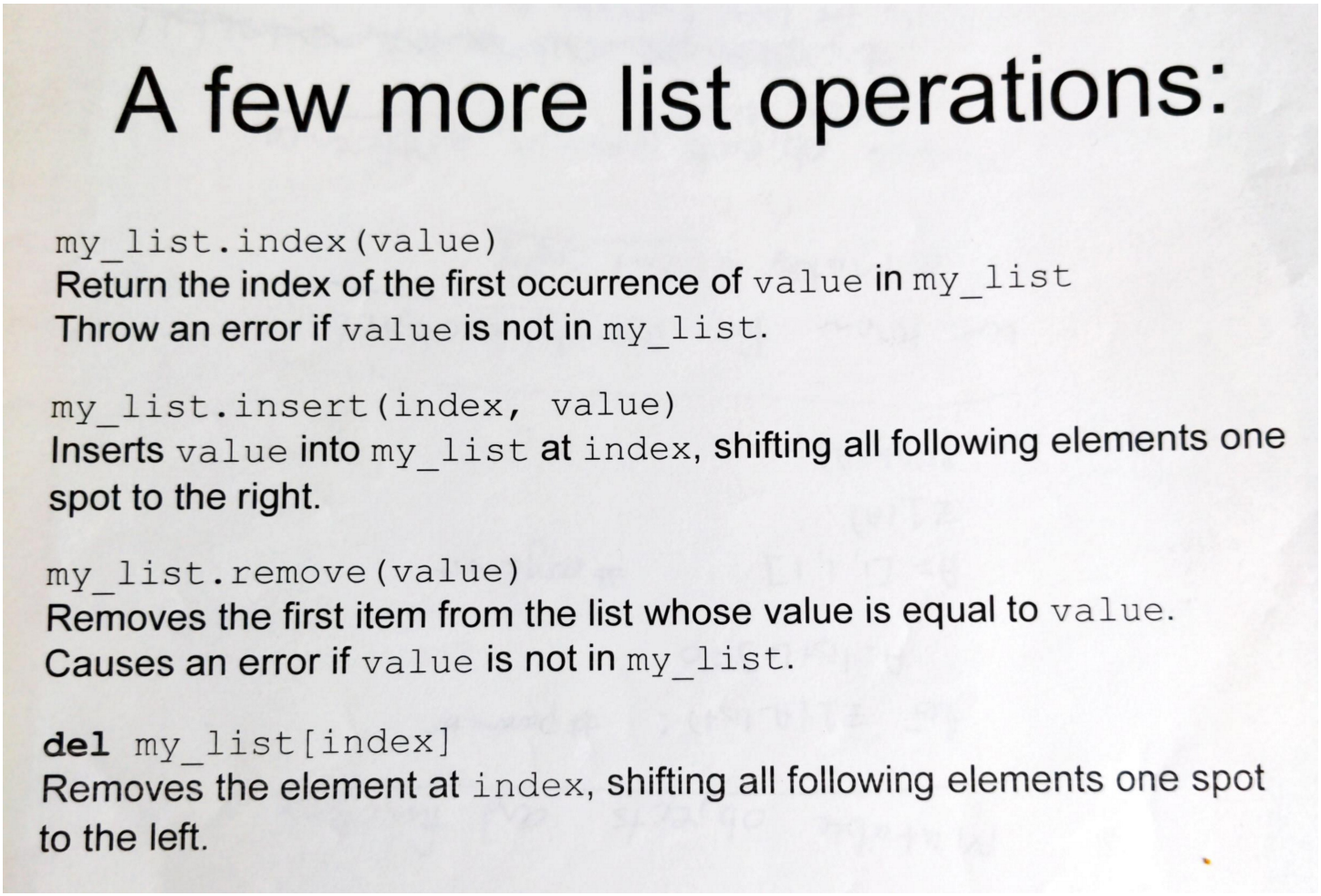
18 Lists

Wednesday, March 3, 2021 11:34 AM

List index insert remove del

Wednesday, May 25, 2022 1:39 PM



```
2  my_list= [5,6,7,8]
3
4
5  print(my_list.index(6))
6  #return the index of the first occurence of (valule)
7  #in the list. Will crash with an error if not in list
8
9  my_list.insert(2,100)
10                 #index, value
11 #inserts at the first number the second number, shifting
12 #all following elements to the right
13
14 my_list.remove(5)
15 #removes the first item from the list whose value is
16 #equal to (value). Crashes with error if not in list
```

```
14 del my_list[2]
15 #removes the element at index, shifting all following
16 #elements to the right
```

Lists append insert extend

Thursday, October 8, 2020 11:07 AM

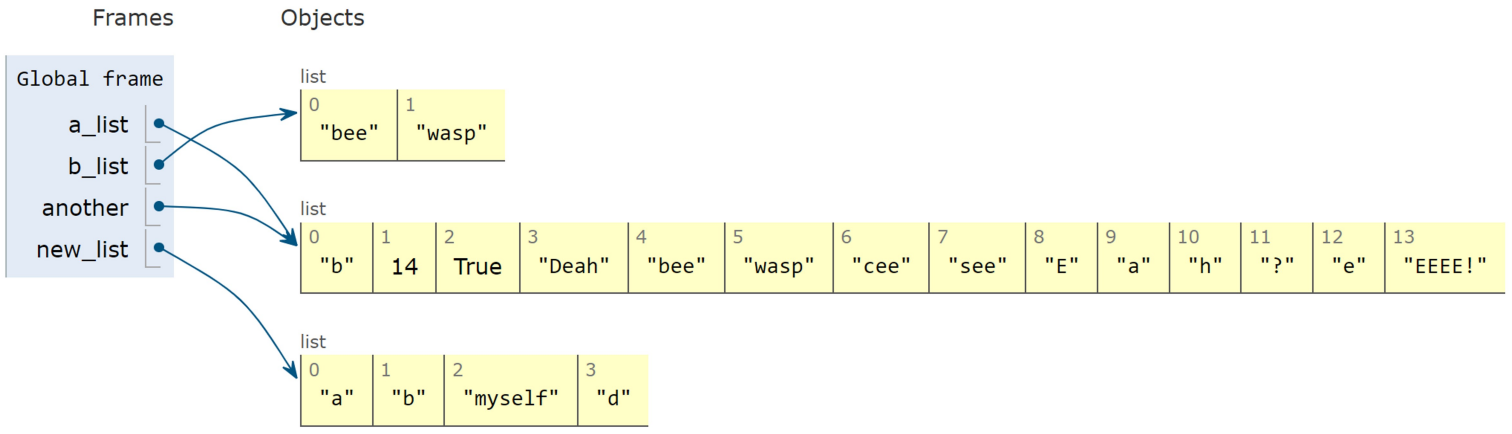
Demo:

```
a_list = ["hat", 34, 27.7] #notice: any data type
print(a_list[0])
print(a_list[-1])
print(a_list[:])
```

1 min: predict what will happen. (Is it on your notesheet?)

```
#lists are mutable!
a_list = ["a", 14, True]

#let's change a value
a_list[0] = "b"

#what if we want to add something to the list?

#use the append method
a_list.append("Deah")

# or we could make a new list and concatenate it
b_list = ["bee","wasp"]
a_list = a_list + b_list

#or do it even without a variable
a_list = a_list + ["cee", "see"]
```

```
another = a_list

a_list.extend("Eah?")
#extend adds a list. In this case, it turns the string into a list
a_list.extend(["e", "EEEE!"])

#append is not the same as extend!
new_list = [44,55]
new_list.append([55,66])

new_list = ["a", "b", "d"]

#what if you wanted to insert in the the middle of a list?
new_list.insert(2, "myself")
                #position, value
```



Link to code

Start

```
a = len(a_list)
b = len(["abc"])
c = len([])
d = 34 in a_list
e = "34" not in a_list
f = a_list + ["Vikings", "WWU"]
g = a_list.append("hidden figures")
h = a_list.append(["Katherine", "Johnson"])
i = a_list.insert(0, "Dorothy")
```

✓ Self work or breakout rooms
1. discuss with group
2. Poll: what would you like me to go over?

Lists: Lightning Round!

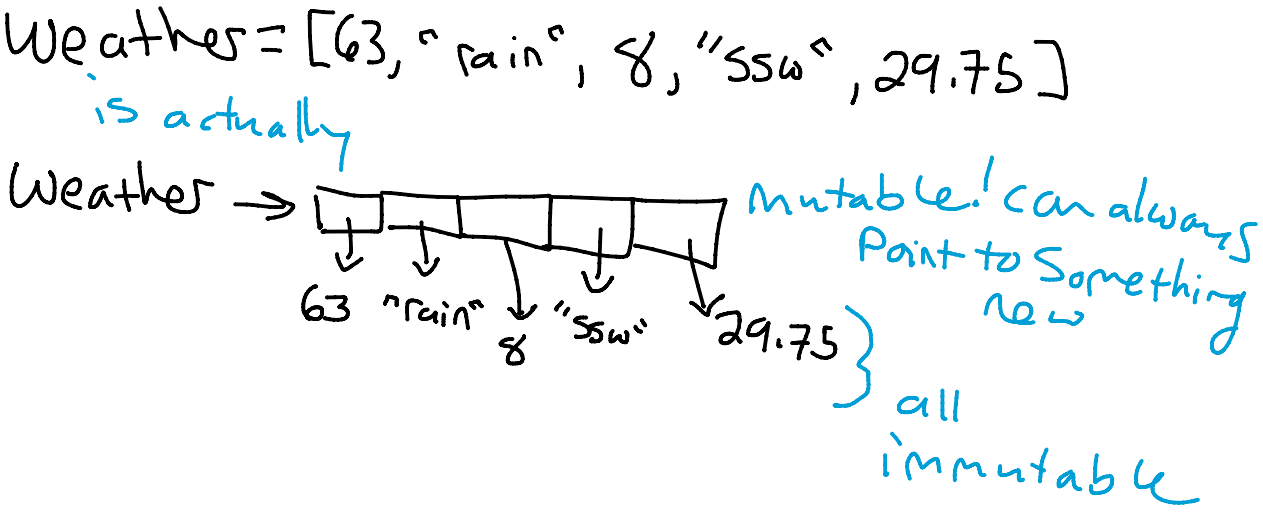
True or False?

```
season12 = ["Brita", "Gigi", "Jackie", "Nicky"]
```

```
"Gigi" in season12

"Nicky" in season12[1:3]

len(season12[1:4]) == 3

"Brita" in (season12 + ["Jan"])[2:]

len(season12[1:2] * 4) == 8
```

Lists and references

Tuesday, March 1, 2022 11:02 PM



Implications of Mutability

```
weather = [63, "light rain"]
tomorrow_weather = weather
tomorrow_weather[0] = 68
print(weather[0])
```

ABCD: What does the above code print?

A B C D

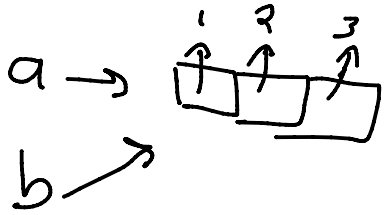
A. "light rain"
B. Error
C. 63
D. 68

Lists and copies

Friday, March 4, 2022 12:11 AM

What if you wanted a copy? So

```
a = [1,2,3]
b = a
```



actually

Python makes the int objects and the list points to them.

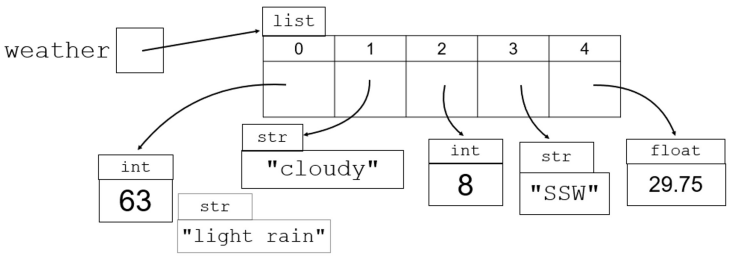
We've simplified as

because it's well, simpler

but for this part we're going to show all the arrows

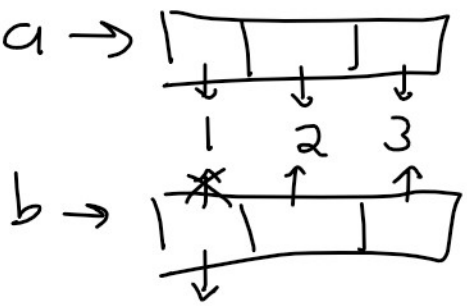
List elements store references to objects

```
weather = [63, "light rain", 8, "SSW", 29.75]
weather[1] = "cloudy"
```



To Make a copy of a list

`b = a.copy()`



`b[0] = 99`

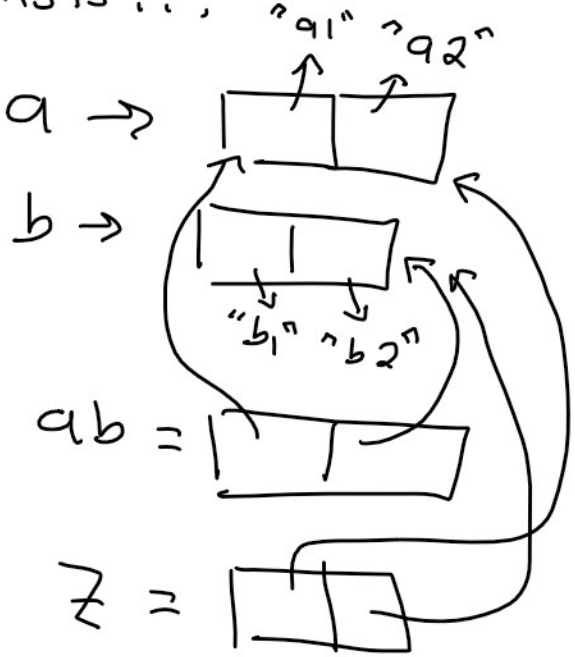
99

Note a[0]!

Watch out!

for lists inside lists...

```
a = ["a1", "A2"]
b = ["b1", "b2"]
```




```
ab = [a, b]
z = ab.copy()
```

What do... Soon

Demo

```
weather = [63, "light rain"]
tomorrow_weather = weather
tomorrow_weather[0] = 68
print(weather[0])
```

ABCD: What does the above code print? A B C D

A. light rain
B. Error
C. 63
D. 68

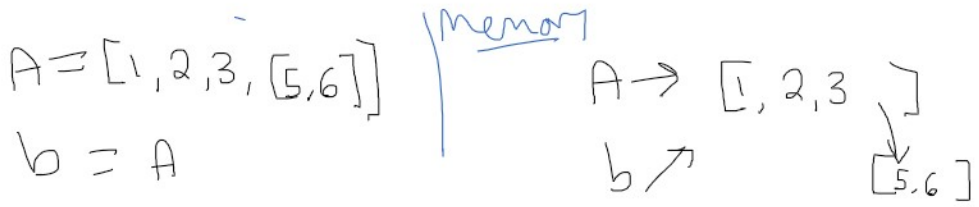
Quick fix for making a true copy of a list:

weather_2m = weather.copy()
#show in python tutor

Or
weather_next_week = weather[:] #see the text book

Lists: Shallow vs deep copy

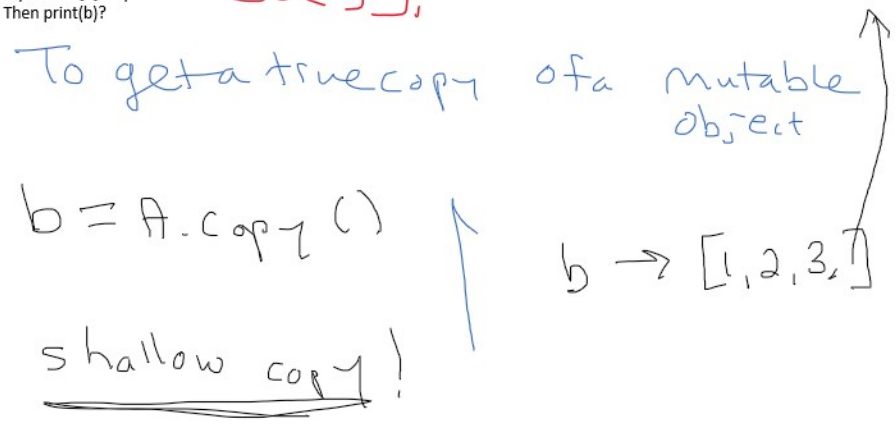
Wednesday, March 3, 2021 12:15 PM



What would print(b) print?

If you do a[1] = "y"
Then print(b)?

Guess!



Python 3.6
(known limitations)

```
1  a =[1,2,3]
2  b =a.copy() #shallow copy
3  b[0] = "b" #changes only b
4
5  a[0] = [4,5] #if there is an object nested
6  c = a.copy() #then copy only copies the reference
7
8  a[0][0] = "a"
9
10 print(c[0][0]) #c changed too!
```

Edit this code

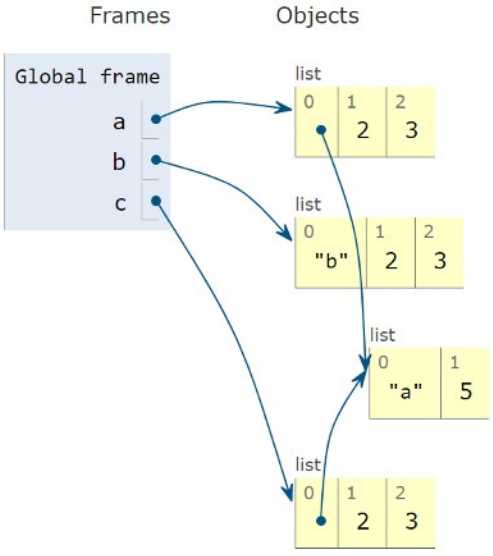
ıe that just executed
ext line to execute

<< First | < Prev | Next > | Last >>

Done running (7 steps)

Print output (drag lower right corner to resize)

```
a
```



Import copy

new_list = copy.deepcopy(old_list)

List assigning & slicing

Tuesday, March 1, 2022 11:21 PM

```
3  a_list = ["a",14, True, ["c", "d"]]
4
5
6  #you can slice lists!
7
8  b = a_list[3:4] #utbni
9  c = a_list[:3] #defaults to [0:3]
10 d = a_list[2:] #defaults to [2:len(a_list)]
11 print(b)
```

pythonScratchpad.py

```
1  #assign
2  a= [5,6,7,8]
3  a[0] = 10
4
5  #slice
6  print(a[0:3]) # [5,6,7]
7
8  #assign a slice!
9  a[:2] = [0,1]
10
11 #what about too small?
12 a[:3] = a[2:] # replaces and shortens
13 #[0,1,7, (8)] = [7,8]
14   #3 positions gets results of 2
15   #[7,8,8]
16
17 #too big?
18 a[:1]= a[:2] #makes it bigger!
19  #[7,8,8,8]
```

Q1: Implement the following function that returns the letters in a student's WWU username given their first and last name.

Code 1

```
def uun_letters(first_name, last_name):
    """ Return the letters in a student's WWU Universal
        Username given the student's first_name and last_name.
        The username begins with up to 6 characters of
        the last name, followed by the first letter of the
        first name. Return the username in all lower case.
        Examples: uun_letters("Scott", "Wehrwein") # => "wehrwes"
                  uun_letters("Ned", "Stark") # => "starkn"
    """
```

Deep copy

Friday, March 4, 2022 12:36 AM

```python
import copy
a = [1,2,3,[4,5]]
b= a.copy() #shallow copy

c =copy.deepcopy(a) #deep copy

a[3][0]= "Z"


print(a[3][0])
print(b[3][0])
print(c[3][0])
```