24

Tuesday, February 22, 2022 10:45 PM

Blue Blood by Foals

From <https://discord.com/channels/@me/1043248466115186728>

Nested objects traps

Monday, November 21, 2022 3:03 PM

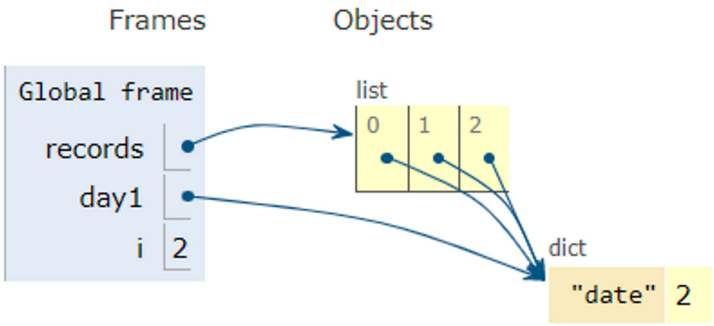
Write code in Python 3.8

```python
######## doesn't work

records = []
day1 = {"date": -1}

for i in range(3):

    day1["date"] = i
    records.append(day1)

print(records)



############## works!

records2 = []
day2 = {"date": -1}

for i in range(3):
    day2= {}
    day2["date"] = i
    records2.append(day2)

print(records2)
```

https://pythontutor.com/visualize.html#code=%23%23%23%23%23%23%23%20doesn't%20work%0A%0Arecords%20%3D%20%5B%5D%0Aday1%20%3D%20%7B%22date%22%3A%20-1%7D%0A%0Afor%20i%20in%20range%283%29%3A%0A%20%20%20%20%0A%20%20%20%20day1%5B%22date%22%5D%20%3D%20i%0A%20%20%20%20records.append%28day1%29%0A%20%20%20%20%0Aprint%28records%29%0A%0A%0A%0A%23%23%23%23%23%23%23%23%23%23%23%23%23%20works!%0A%0Arecords2%20%3D%20%5B%5D%0Aday2%20%3D%20%7B%22date%22%3A%20-1%7D%0A%0Afor%20i%20in%20range%283%29%3A%0A%20%20%20%20day2%3D%20%7B%7D%0A%20%20%20%20day2%5B%22date%22%5D%20%3D%20i%0A%20%20%20%20records2.append%28day2%29%0A%20%20%20%20%0Aprint%28records2%29&cumulative=false&heapPrimitives=true&mode=edit&origin=opt-frontend.js&py=3&rawInputLstJSON=%5B%5D&textReferences=false

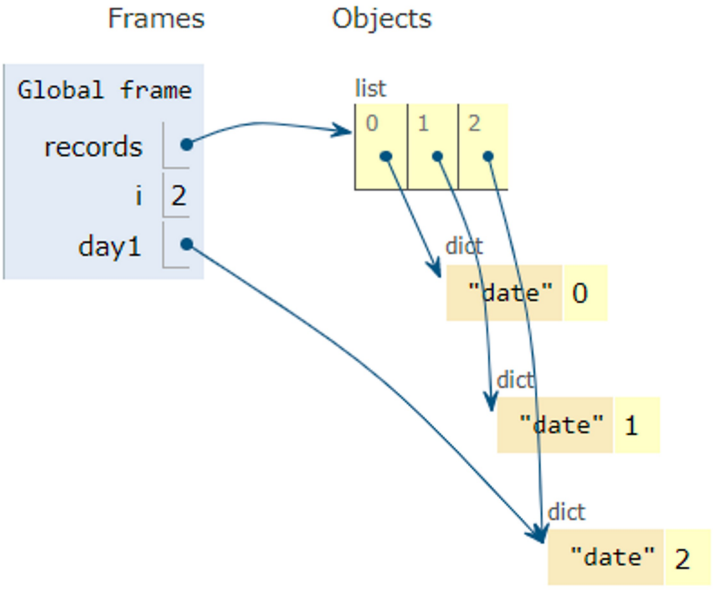
Print output (drag lower right corner to resize)

```
[{'date': 2}, {'date': 2}, {'date': 2}]
```



Print output (drag lower right corner to resize)

```
[{'date': 0}, {'date': 1}, {'date': 2}]
```



Worksheets on mutability

Friday, March 4, 2022 12:36 AM

Problem 2: Execute the following, drawing and updating the memory diagram for each variable and object involved.

Problem 2 Code	Problem 2 Diagram

`a = []`
`b = [1]`
`a.insert(0, b)`
`b[0] = 4`
`a.insert(0, b)`
`print(a)`	

Problem 3: Implement the following function to create a true copy of a list. Hint: one possible approach uses a loop and the append method.

```
def copy_list(in_list):
    """ Return a new list object containing the same elements
    as in_list.
    Precondition: in_list's contents are all immutable. """
```

Problem 4: Implement this function, which removes half the elements from the given list.

```
def snap(avengers):
   """ Remove a randomly chosen half of the
       elements from the given list of avengers """
```

```
#write a fuction that takes in alist
#and randomly removes half of the elements

import random

def snap(items):
    for i in range(len(items)//2):
        del items[random.randint(0,len(items)-1)]

a = [1,2,3,4,5,6,7]
snap(a)
```

List challenge

Thursday, March 4, 2021 1:50 PM

think / breakout room

w 22 Just Demo

prep slides

```
a = [3,4,5]
a.insert(0,4)
a[2:] = a[1:4]
a.remove(4)
a.append(a.index(5))
del a[1]
print(len(a))
print(4 not in a)
print(a[-2] )
```

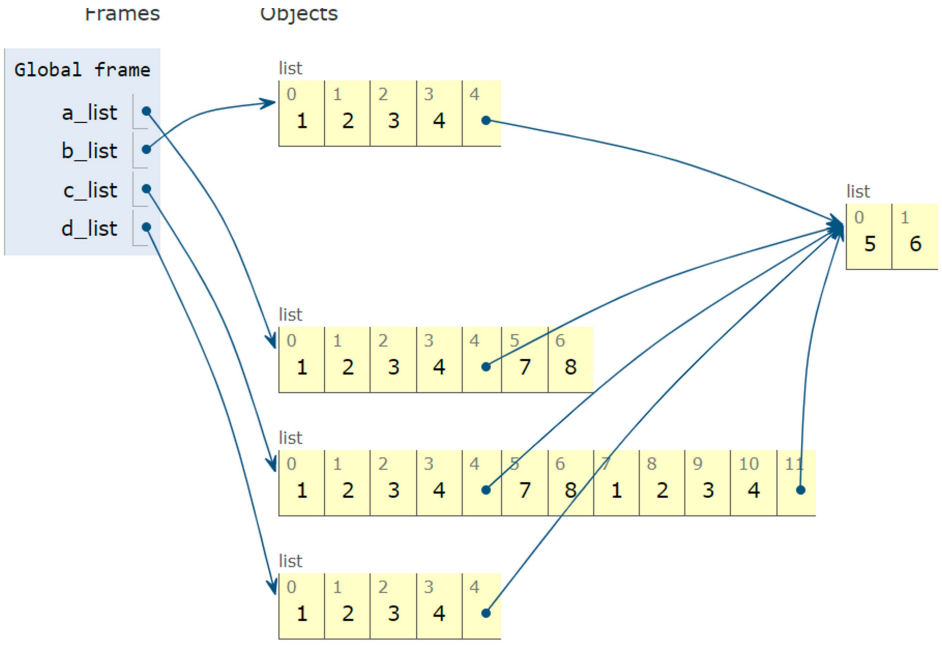
```
a = [3,4,5]              #[3,4,5]
a.insert(0,4)            #[4,3,4,5]
a[2:] = a[1:4]           #[4,3,3,4,5]
a.remove(4)              #[3,3,4,5]
a.append(a.index(5))     #[3,3,4,5,3]
del a[1]                 #[3,4,5,3]
print(len(a))            #4
print(4 not in a)        #False
print(a[-2] )            #5
```

List append vs extend

Friday, March 25, 2022 4:07 PM

(drag lower right corner to resize code editor)

```
1  a_list = [1,2]
2  a_list.extend([3,4])
3  a_list.append([5,6])
4  b_list = a_list
5  a_list = a_list + [7,8]
6  c_list = a_list + b_list
7  d_list = c_list[7:]
8
```



Mutable vs immutable in vars

Wednesday, May 25, 2022 9:50 AM

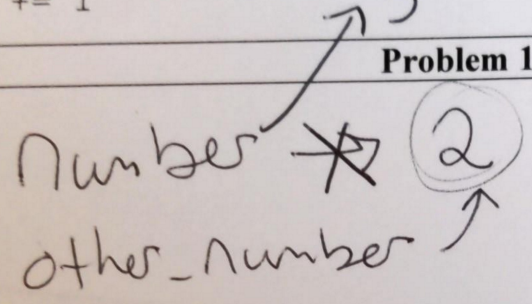
CSCI141 Lecture 22

Problem 1: Execute the following, drawing and updating the memory diagram for each variable and object involved.

Problem 1 Code

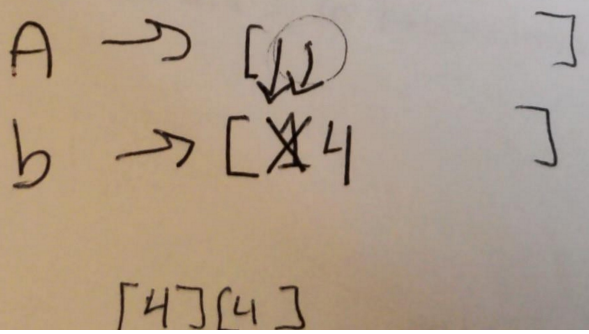
`number = 2`
`other_number = number`
`number += 1`

Problem 1 Diagram



Problem 2: Execute the following, drawing and updating the memory diagram for each variable and object involved.

Problem 2 Code	Problem 2 Diagram

`a = []`
`b = [1]`
`a.insert(0, b)`
`b[0] = 4`
`a.insert(0, b)`
	

```
a.insert(0, b)
b[0] = 4
a.insert(0, b)
print(a)
```

[4][4]

Functions & lists

Wednesday, May 25, 2022 9:52 AM

In breakout rooms

CSCI141 Lecture 23

To the right of each program, draw the memory diagram for the program as it is executed.

```
def z1(a_list):
    a_list[0] = 0

a = [1, 1, 1]
z1(a)
print(a)
```

```
def z2(a_list):
    a_list = []

a = [1, 1, 1]
z2(a)
print(a)
```

```
def z3(x):
    a_list = [x, x, x]
    return a_list
```

```
    return a_list

b = 2
a = z3(b)
print(a)
```