Exercise 2

Finger Exercises due Aug 5, 2020 20:30 -03

Exercise 2

4/4 points (graded)

ESTIMATED TIME TO COMPLETE: 8 minutes

1. Consider the following code:

```
class Clock(object):
    def __init__(self, time):
        self.time = time
    def print_time(self):
        time = '6:30'
        print(self.time)

clock = Clock('5:30')
clock.print_time()
```

What does the code print out? If you aren't sure, create a Python file and run it.

5:30

✓ **Answer:** 5:30

Explanation:
`5:30` prints out because we printed out the attribute `self.time`, not the local variable `time`.

2. Consider the following code:

```
class Clock(object):
    def __init__(self, time):
        self.time = time
    def print_time(self, time):
        print(time)

clock = Clock('5:30')
clock.print_time('10:30')
```

What does the code print out? If you aren't sure, create a Python file and run it.

10:30

✓ **Answer:** 10:30

Explanation:
What does this problem tell us about giving parameters the same name as object attributes?
In short, it is needlessly confusing. It is less confusing if you give parameters, local variables, and attributes different, distinct names to avoid the confusion that arises in this problem.

3. Consider the following code:

```
class Clock(object):
    def __init__(self, time):
        self.time = time
    def print_time(self):
        print(self.time)

boston_clock = Clock('5:30')
paris_clock = boston_clock
paris_clock.time = '10:30'
boston_clock.print_time()
```

What does the code print out? If you aren't sure, create a Python file and run it.

10:30

✓ **Answer:** 10:30

Are `boston_clock` and `paris_clock` different objects?

Yes [✓ No]

✓

Explanation:
`boston_clock` and `paris_clock` are two names for the same object. This is called aliasing.

Enviar

ℹ Answers are displayed within the problem

Exercise 2

Ocultar discussão

Topic: Lecture 9 / Exercise 2

Add a Post

‹ All Posts

Can someone explain #3?

question posted about 21 hours ago by anônimo

I'm having a hard time understanding it even though I ran it through my ide.

This post is visible to everyone.

Add a Response

4 responses

phibeer
about 20 hours ago

Let's first talk about the class definition:

```
class Clock(object):
    def __init__(self, time):
        self.time = time
    def print_time(self):
        print(self.time)
```

The **init**-method creates an instance with an attribute of the name time. This attribute is only set when the **init** function is called during the instance creation. The print_time() method is defined as such that it takes this instance attribute and prints the value of it when called.

Now, to what I guess is your actual question. What happens here:

boston_clock = Clock('5:30') --> creates an instance of the class Clock with the instance attribute time that holds the value of 5:30

paris_clock = boston_clock --> (this is where the magic happens) setting boston_clock to paris_clock creates a pointer to the same location in memory (or in other words the space in memory gets a second name associated with it, i.e. paris_clock and bosten_clock point to the same thing

paris_clock.time = '10:30' --> sets the variable time of the instance (which now has two names) to 10:30

boston_clock.print_time() --> returns that value of the instance attribute time

How can you confirm this:

After you have created both objects and run the code, check for their addresses by calling the objects, like so:

```
boston_clock
Out[17]: <__main__.Clock at 0x7f7ffb914f50>

paris_clock
Out[18]: <__main__.Clock at 0x7f7ffb914f50>
```

And you'll notice that point to the same address in memory.

Add a comment

Spenstine
about 15 hours ago

paris_clock is an alias to boston_clock; the two names refer to the same object. Therefore, changing the time attribute of paris_clock would have the same effect to boston_clock.

Add a comment

Emekadavid2020
about 8 hours ago

The way I thought of it was that paris_clock and boston_clock were referring to the same frames inherited from the class Clock. So, they are sharing the same attributes in that frame. Either of them can change the values of those attributes but that does not mean they are the same objects because paris_clock can decide to point or reference another Clock frame in the future.

Add a comment

poogen
about 3 hours ago

You might want to explore #3 in pythontutor: Clock class.

Add a comment

Exibindo todas as respostas

Add a response:

Pré-visualizar

Enviar