CPS 109 - Lab 7

Agenda

1. Classes
2. Testing

Agenda

Both concepts bring us back to this. You know how we've been droning on about using:

```
if __name__ == "__main__":
```

Well now we have a use for it!

Classes

What's a class? I know your profs haven't gone over this, but if we're introducing testing, we'll introduce classes.

A class is simple: it's a way to consistently define an object.

Classes

Recall: an object is a piece of data in your program. Built in objects are things like ints, lists, etc.

A class is a set of interlinked objects that make up a new object!

Classes

Let's make a basic class:

```
class car:
    def __init__(self, make, colour, price):
        self.make = make
        self.colour = colour
        self.price = price
```

Notice that we use the __init__ here? It's short for initialize! We use this for when we want to make a new object.

Classes

We can also give it methods! (Just built in functions):

```
def age_price(self):
    self.price = self.price - 500
```

Unit Testing

What is unit testing? Simply put: it’s a way to automate testing your code so you don’t have to run it a million times.

Unit Testing

So why did we introduce classes before showing testing? What does this have to do with that “__main__” business?!

Unit Testing

So why did we introduce classes before showing testing? What does this have to do with that “__main__” business?!

Unit Testing

Let's look at an example from your prof:

```
import unittest
import ExampleOne

class myTests(unittest.TestCase):
    def test1(self):
        self.assertEqual(ExampleOne.mostfrequent([5, 2, 9, 2, 9, 1, 18, 9, 3]), 9)
    def test2(self):
        self.assertEqual(ExampleOne.mostfrequent(['cat', 'dog', 'dog', 'cat', 'cat']), 'cat')
    def test3(self):
        self.assertEqual(ExampleOne.mostfrequent([5]), 5)
    def test4(self):
        self.assertEqual(ExampleOne.mostfrequent([1, 2, 3, 3, 2, 1]), 1)
    def test5(self):
        self.assertEqual(ExampleOne.mostfrequent([(5, 5, 5), (3, 2, 1), (5, 5, 5)]), (5, 5, 5))


if __name__ == '__main__':
    unittest.main(exit=True)
```

Unit Testing

That’s weird. The mostfrequent function is in a different file!

That’s because we’re importing it at the top. We can import from different files like we do with libraries. Speaking of libraries, notice how we import “unittest”?

Unit Testing

Long story short: make classes, call methods and classes outside of files you've already made, and then test them!