

inheritance_exercise_clothing

November 30, 2019

1 Inheritance Exercise Clothing

The following code contains a Clothing parent class and two children classes: Shirt and Pants.

Your job is to code a class called Blouse. Read through the code and fill out the TODOs. Then check your work with the unit tests at the bottom of the code.

```
In [4]: class Clothing:
```

```
    def __init__(self, color, size, style, price):
        self.color = color
        self.size = size
        self.style = style
        self.price = price
```

```
    def change_price(self, price):
        self.price = price
```

```
    def calculate_discount(self, discount):
        return self.price * (1 - discount)
```

```
    def calculate_shipping(self, weight, rate):
        return weight*rate
```

```
class Shirt(Clothing):
```

```
    def __init__(self, color, size, style, price, long_or_short):

        Clothing.__init__(self, color, size, style, price)
        self.long_or_short = long_or_short
```

```
    def double_price(self):
        self.price = 2*self.price
```

```
class Pants(Clothing):
```

```
    def __init__(self, color, size, style, price, waist):
```

```

        Clothing.__init__(self, color, size, style, price)
        self.waist = waist

    def calculate_discount(self, discount):
        return self.price * (1 - discount / 2)

# TODO: Write a class called Blouse, that inherits from the Clothing class
# and has the the following attributes and methods:
#     attributes: color, size, style, price, country_of_origin
#     where country_of_origin is a string that holds the name of a
#     country
#
#     methods: triple_price, which has no inputs and returns three times
#     the price of the blouse
#
#

# TODO: Add a method to the clothing class called calculate_shipping.
# The method has two inputs: weight and rate. Weight is a float
# representing the weight of the article of clothing. Rate is a float
# representing the shipping weight. The method returns weight * rate
class Blouse(Clothing):
    def __init__(self, color, size, style, price, country_of_origin):
        Clothing.__init__(self, color, size, style, price)
        self.country_of_origin = country_of_origin

    def triple_price():
        return self.price**3

```

In [5]: *# Unit tests to check your solution*

```

import unittest

class TestClothingClass(unittest.TestCase):
    def setUp(self):
        self.clothing = Clothing('orange', 'M', 'stripes', 35)
        self.blouse = Blouse('blue', 'M', 'luxury', 40, 'Brazil')
        self.pants = Pants('black', 32, 'baggy', 60, 30)

    def test_initialization(self):
        self.assertEqual(self.clothing.color, 'orange', 'color should be orange')
        self.assertEqual(self.clothing.price, 35, 'incorrect price')

        self.assertEqual(self.blouse.color, 'blue', 'color should be blue')
        self.assertEqual(self.blouse.size, 'M', 'incorrect size')
        self.assertEqual(self.blouse.style, 'luxury', 'incorrect style')
        self.assertEqual(self.blouse.price, 40, 'incorrect price')
        self.assertEqual(self.blouse.country_of_origin, 'Brazil', 'incorrect country of

```

```

def test_calculateshipping(self):
    self.assertEqual(self.clothing.calculate_shipping(.5, 3), .5 * 3,\
        'Clothing shipping calculation not as expected')

    self.assertEqual(self.blouse.calculate_shipping(.5, 3), .5 * 3,\
        'Clothing shipping calculation not as expected')

tests = TestClothingClass()

tests_loaded = unittest.TestLoader().loadTestsFromModule(tests)

unittest.TextTestRunner().run(tests_loaded)

..
-----
Ran 2 tests in 0.006s

OK

Out[5]: <unittest.runner.TextTestResult run=2 errors=0 failures=0>

In [ ]:

```