Lab 4 – Class & Inheritance Practice

(Fall 2024)

Lab Description:

In this lab, you will write three Python classes: a base class and two subclasses. Given the class descriptions provided below and the sample output at the end of the document, create code to satisfy these requirements.

Class Description:

Class #1: Sandwich:

Create a Sandwich class with the following features:

- Accepts arguments for **order number**, **bread type**, and **meat type** (if you don't like meat, just change it to something else appropriate).
- Contains getters for **order number, bread type,** and **meat type**.
- Contains setters for **bread type** and **meat type**.
- Contains a list of toppings with methods to **add** and **remove** toppings from that list.
- Contains a getter for **price**, which is calculated as $3.75 + $0.50 per topping.
- Contains an **info** method to print all of the relevant information to the screen in an organized manner.

Class #2: Meal:

Create a Meal class with the following features:

- Inherits from the **Sandwich** class.
- Accepts additional arguments for **drink** and **side**, with getters and setters for these values.
- Meal **price** should equal $6.75 + $0.50 per topping.
- Updated **info** method to reflect new features of the Meal class.

Class #3 KidsMeal:

Create a KidsMeal class with the following features:

- Inherits from the **Meal** class.
- Accepts an additional argument for a **toy**, with a getter and setter for this value.
- KidsMeal **price** should equal $4.75 + $0.30 per topping.
- Updated **info** method should reflect new features of the KidsMeal class.

Lab Instructions:

1. Create a file named **sandwich.py**

2. In this file, create the classes for **Sandwich**, **Meal**, and **KidsMeal** as defined in the above Class Description. Ensure that these classes adhere to the specified features.

3. When defining your getter and setter methods, use the **@property** and **@[name].setter** decorators accordingly.

4. When defining constructors for your subclasses, you may use super() or ClassName, whichever you prefer.

5. Create an object of each class and run that object's **.info()** method. Running this method should demonstrate that your code is working correctly.

Notes:

If you have trouble, the following notes may help you in completing the assignment:

- Use a single underscore (_) rather than double underscores (_ _) to precede your attribute names, as this will prevent mangling from interfering with your inheritance.

- Basic list methods such as **len()**, **.append()** and **.remove()** may be useful for your implementation.

- Also consider reviewing the syntax and keyword arguments for **print()**, as this could help you format your output more cleanly.

- Yes, I was hungry when I made this, and I was even hungrier when I finished it.

Submission

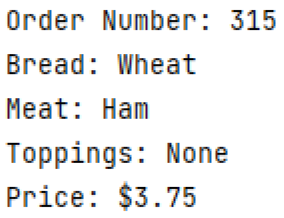
Submit your **sandwich.py** file to the Lab 4 Submission dropbox in iLearn. Although your code will contain three classes, you can write them all in the same file along with your object creation. If you do write multiple files, be sure to include a driver and submit them in a zip file called **[user_id]_lab04.zip**, where **[user_id]** is your Tech username.

Sample Output:

Your output does not need to be in the same format as this, nor do your attributes need to contain the same values; just display the same categories of information in some easily-readable format.

<u>Sandwich:</u>

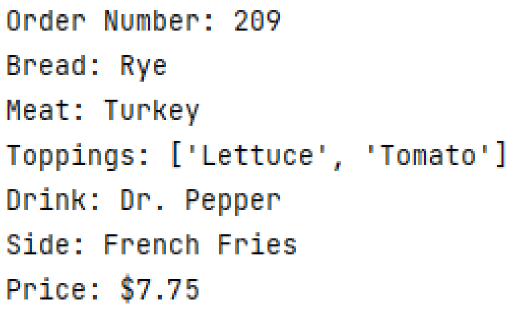
sandwich_object.info()



```
Order Number: 315
Bread: Wheat
Meat: Ham
Toppings: None
Price: $3.75
```

<u>Meal:</u>

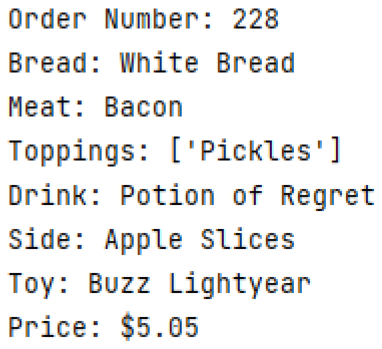
meal_object.info()



```
Order Number: 209
Bread: Rye
Meat: Turkey
Toppings: ['Lettuce', 'Tomato']
Drink: Dr. Pepper
Side: French Fries
Price: $7.75
```

<u>KidsMeal:</u>

kidsmeal_object.info()



```
Order Number: 228
Bread: White Bread
Meat: Bacon
Toppings: ['Pickles']
Drink: Potion of Regret
Side: Apple Slices
Toy: Buzz Lightyear
Price: $5.05
```