OOP - Lesson 3

Using objects

Displaying our objects

- What if I want to display all the info about my object? For example, I want to know my seamstress's experience, talent, and whether she has creativity without specifying each of those attributes in my print statement.
- Well, why don't you try print(my_seamstress)? That doesn't work. Yet.
- Add a method **__str__()** to your Seamstress class and in that method, code all the information that you would like to display about your seamstress. The python interpreter will use that method whenever you try to print() a Seamstress object.
- Go ahead and add it yourself.

Default attribute values

If you would like there to be a default value for one or more of your instance attributes, this is how you would do it:

```python
class Student:
    def __init__(self, name="Guest", age=25)
        self.name=name
        self.age=age
```

Instantiating an object in another file

- You can use the objects that you define in a separate file.
- Don't forget the import statement
- For example, your file name might be Professions.py. In that file, you might define a class Seamstress and a class Ganenet. You would reference those objects one of 2 ways:

```
import Professions as P

seamstress_sara = P.Seamstress(True, "one year")
ganenet_chavi = P.Ganenet(15)#15 kids max
```

```
from Professions import Seamstress, Ganenet

seamstress_sara = Seamstress(True, "one year")
ganenet_chavi = Ganenet(15)#15 kids max
```

Best Practices

- Name instances of objects by including the type of object it is. For example, instead of calling my seamstress instance sara_cohen, call it seamstress_sara.
- COMMENTS!
- Name your variables in a way that indicates the datatype. For example, instead of "talent", use "has_talent".
- Methods should be named like commands. For example, instead of washing_dishes, it should be wash_dishes.
- Variables should be named nouns
- Functionality that belongs to the class should remain inside the class. For example, if every seamstress must follow the protocol of take_measurements, then cut_material, then sew_garment, those 3 methods should be wrapped in another method inside the seamstress class.

Class Assignment

- Using the credit card bill file, define a class called CreditCardStatement.
- Plan out what your class will need to do. Write a list of variables you think you will need and methods you may want to use when reading a credit card statement.
- 2 required methods:
    - Write a method to read the data from the file. (note: The file credit_card_bill.txt contains the name of the store, amount charged , and the date. Each one is separated by a comma. The separate charges are separated by a carriage return/enter. Dates are in the format dd/MM/yyyy)
    - Write a method that will display your credit card statement in a nice format.
- In another file, instantiate the object and call the method(s).
- Bonus: Include validation. For example, cannot be a future date.
- Bonus 2: define an additional class. Use it inside the CreditCardStatement class.
- Have fun with it! Include concepts that you've learned in other classes.