Unit 1 / Lesson 3 / Project 4

Model the Bicycle Industry

Estimated Time: 4-5 hours

To close out this unit, you'll use your budding knowledge of Python classes and object oriented programming to create a simplified model of the bicycle industry. To succeed at this assignment, you'll draw on nearly everything we've learned about so far in this course: variables, strings, numbers, lists, control flow, functions, and classes.

This is the first project you'll create where there isn't a single correct answer. We would like you to think of your solution to this project not only as a chance to get more practice with Python syntax, but as an opportunity to start thinking about **software architecture** decisions. You will need to consider how the different parts of the model are best represented in code, and how they can be coupled together in a clean and flexible way.

As you advance as a programmer, you'll start to understand software architecture decisions in terms of tradeoffs. There are numerous ways to fulfill the project requirements, each with distinct advantages and disadvantages. Honing your ability to recognize and analyze these tradeoffs is a key skill that separates intermediate and advanced software developers from novices.

Once you've completed this project, be sure to send your solution to your mentor and plan on discussing it at your next session. Your mentor will be able to give you feedback on how you've architected your solution.

The Scenario

For this project, we'll imagine that you've been hired by a business analyst to build a system that models the bicycle industry. You need to be able to model bicycles, which have a fixed cost to produce, bike shops, which sell bicycles with an added margin on top, and customers, who have different budgets for buying a bicycle.

Basic Requirements

Design Python classes

You should create classes to represent each of the following parts of our model:

- Bicycle
  - Have a model name
  - Have a weight
  - Have a cost to produce
- Bike Shops
  - Have a name
  - Have an inventory of different bicycles
  - Sell bicycles with a margin over their cost
  - Can see how much profit they have made from selling bikes
- Customers
  - Have a name

  - Have a fund of money to buy a bike
  - Can buy and own a new bicycle

Add some code to use your classes

The code should:

- Create a bicycle shop that has 6 different bicycle models in stock. The shop should charge its customers 20% over the cost of the bikes.
- Create three customers. One customer has a budget of $200, the second $500, and the third $1000.
- Print the name of each customer, and a list of the bikes offered by the bike shop that they can afford given their budget. Make sure you price the bikes in such a way that each customer can afford at least one.
- Print the initial inventory of the bike shop for each bike it carries.
- Have each of the three customers purchase a bike then print the name of the bike the customer purchased, the cost, and how much money they have left over in their bicycle fund.
- After each customer has purchased their bike, the script should print out the bicycle shop's remaining inventory for each bike, and how much profit they have made selling the three bikes.

Refactor your solution

Once you have everything working, refactor your code to make it more **modular**. You should:

- Create a file named *bicycles.py* that contains each of your classes.
- Create a file named *main.py* that imports those classes, and uses them.

Run `main.py` and make sure your refactored code still works like it did before.

Good Luck!

We're going to leave you largely on your own to complete this assignment, and we expect that your solution will **not** be optimal because you're just starting to work with object oriented programming and it takes time to develop an instinctive sense of what makes for good object oriented design. As we said before, think of your solution to this project mainly as a conversation starter for you and your mentor.

Extra Challenge

If you found completing the basic requirements fairly straightforward then you should try to extend your code to model the bicycles in more detail.

Alter your classes

You should add new classes to represent the following bike parts:

- Wheels
    - Have a model name
    - Have a weight
    - Have a cost to produce
    - There should be a total of three different wheel types
- Frames
    - Can be made of aluminum, carbon, or steel
    - Have a weight

    - Have a cost to produce

Then you should modify your Bicycle class. The updated class should:

- Bicycle
    - Be comprised of two wheels of the same type and a frame
    - Have a weight equal to the sum of the weight of the frame and two wheels
    - Have a cost to produce equal to the sum of the two wheels' and frame's cost to produce

You may also need to update your testing script to reflect the changes that you have made here.

Extension Exercise

If the extra challenges were not a problem and you're running ahead of schedule then you could try to extend your model even further to add bicycle manufacturers.

Alter your classes

You should add one or more classes to represent:

- Bicycle Manufacturers
    - Have a name
    - Produce three models of bikes each
    - Have a percentage over cost which they sell bikes to bike shops at

Then you should modify your Bicycle class again. The updated class should:

- Bicycle

  - Have a manufacturer

Update your testing script

The testing script should be modified so that it:

- Creates two bicycle manufacturers, which both produce three different bicycle models
- Makes the bike shops stock their inventory by purchasing bikes from manufacturers

☆ ☆ ☆ ☆ ☆ · Report a typo or other issue

example.com/project

✓ Submit your project

Previous

Next