Object Oriented programming Exersizes

Simple: Class for Basic Geometry Shapes

Create a class hierarchy for basic geometry shapes: Shape (abstract base class), Circle, and Rectangle. Each shape should have attributes for its dimensions and a method to calculate its area. Use inheritance and abstraction.

Specifically, the goals of this exersize are as follows:

- Create an abstract class Shape with an abstract method `calculate_area()`.
- Create subclasses `Circle` and `Rectangle` that inherit from `Shape`.
- Implement the `calculate_area()` method in each subclass.
- Demonstrate the use of inheritance and abstraction to calculate the areas of different shapes.

Intermediate: Online Shopping System

Design an online shopping system. Create classes for Customer, Product, ShoppingCart, and Order. Implement methods for adding products to the cart, placing orders, and calculating the total order amount. Use encapsulation and demonstrate a clear class relationship.

Specifically, the goals of this exersize are as follows:

- Create a class `Customer` with attributes like `name` and `email`.
- Create a class `Product` with attributes like `name`, `price`, and `quantity`.
- Create a class `ShoppingCart` to add `products` and calculate the total cart amount.
- Create a class `Order` to place an `order`, `update inventory`, and generate an `order summary`.
- Showcase the ability to add products to a cart, place an order, and calculate the total order amount.

Difficult: Library Management System

Develop a library management system. Create classes for Library, Book, Member, and Transaction. Implement features like borrowing and returning books, tracking book availability, and handling late returns. Include features like fine calculation and membership status management. Showcase complex class interactions and behaviors.

Specifically, the goals of this exersize are as follows:

- Create a class Book with attributes like title, author, and availability status.
- Create a class Member with attributes like name, membership status, and borrowing history.
- Create a class Transaction to manage borrowing and returning books, and calculate late fines.
- Create a class Library to manage the overall system, including book inventory and member management.

- Showcase the ability to borrow and return books, calculate late fines, manage book availability, and handle member interactions.

Please note, this final exersize is more difficult. It is intentionally left more ambigious in order for you to use your knowledge and ideas on how to make this idea work.

As a next step of progression, please review the Intro to Genetic Algorithms Repository, your task is to develop your own Genetic Algorithm to solve the same problem. The repository provides the steps and knowledge and a working version for use when you get stuck. Here is the link: https://github.com/trav-d13/Genetic-Algorithm-Intro (https://github.com/trav-d13/Genetic-Algorithm-Intro)

Goodluck!!