Lab2(b) Object Oriented Python

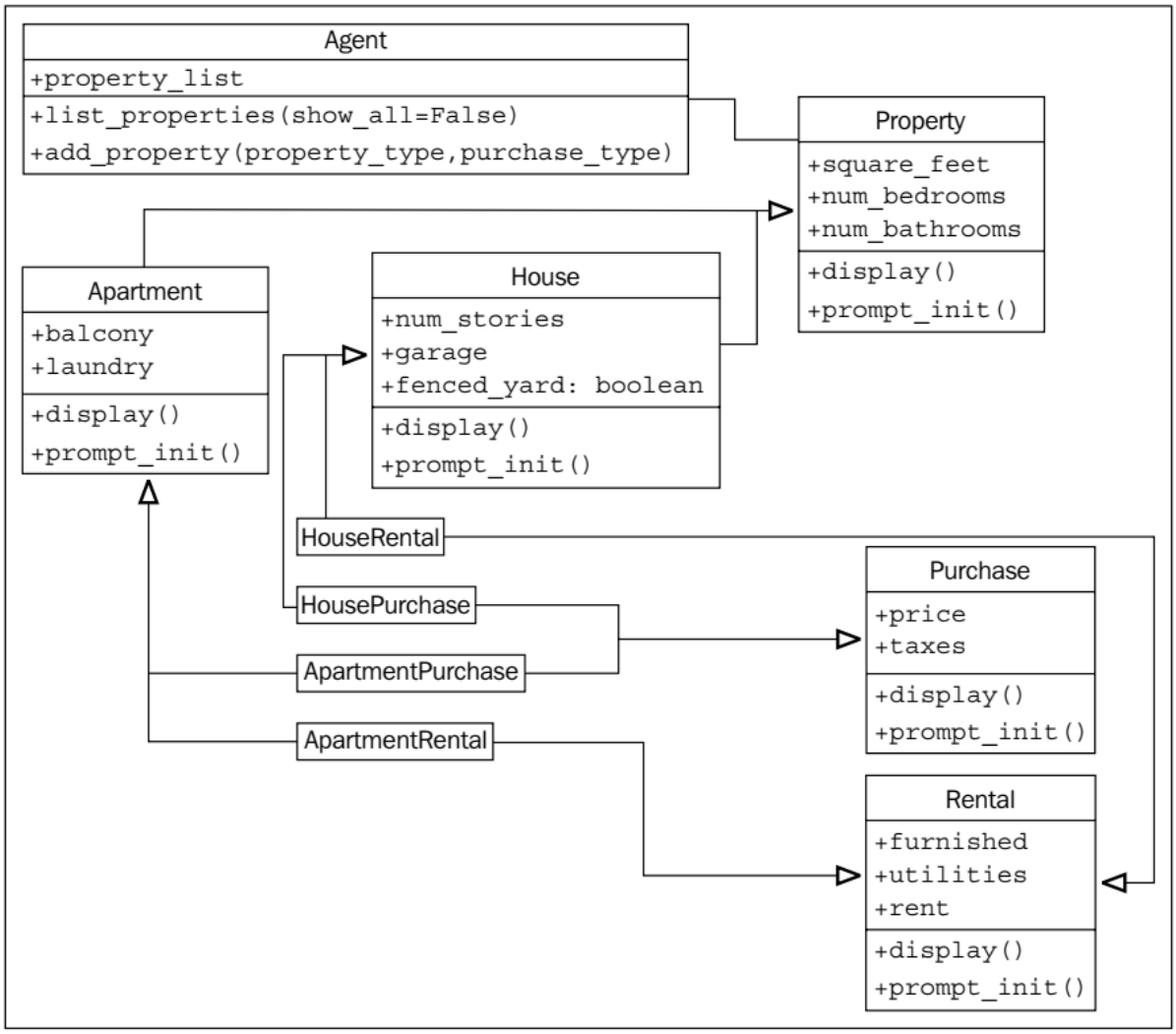
References:

https://pynative.com/python/object-oriented-programming/
https://www.geeksforgeeks.org/python-classes-and-objects/
https://www.programiz.com/python-programming/object-oriented-programming
https://realpython.com/python3-object-oriented-programming/

Task : Design a simple real estate application that allows an agent to manage properties available for purchase or rent. There will be two types of properties: apartments and houses. The agent needs to be able to enter a few relevant details about new properties, list all currently available properties, and mark a property as sold or rented. For brevity, we won't worry about editing property details or reactivating a property after it is sold.

Looking at our requirements, it seems like there are quite a few nouns that might represent classes of objects in our system. Clearly, we'll need to represent a property. Houses and apartments may need separate classes. Rentals and purchases also seem to require separate representation. Since we're focusing on inheritance right now, we'll be looking at ways to share behavior using inheritance or multiple inheritance. House and Apartment are both types of properties, so Property can be a superclass of those two classes. Rental and Purchase will need some extra thought; if we use inheritance, we'll need to have separate classes, for example, for HouseRental and HousePurchase, and use multiple inheritance to combine them. This feels a little clunky compared to a composition or association-based design, but let's run with it and see what we come up with. Now then, what attributes might be associated with a Property class? Regardless of whether it is an apartment or a house, most people will want to know the square footage, number of bedrooms, and number of bathrooms. (There are numerous other attributes that might be modeled, but we'll keep it simple for our prototype.) If the property is a house, it will want to advertise the number of stories, whether it has a garage (attached, detached, or none), and whether the yard is fenced. An apartment will want to indicate if it has a balcony, and if the laundry is ensuite, coin, or off-site. Both property types will require a method to display the characteristics of that property. At the moment, no other behaviors are apparent. Rental properties will need to store the rent per month, whether the property is furnished, and whether utilities are included, and if not, what they are estimated to be. Properties for purchase will need to store the purchase price and estimated annual property taxes. For our application, we'll only need to display this data, so we can get away with just adding a display() method similar to that used in the other classes.

Finally, we'll need an Agent object that holds a list of all properties, displays those properties, and allows us to create new ones. Creating properties will entail prompting the user for the relevant details for each property type. This could be done in the Agent object, but then Agent would need to know a lot of information about the types of properties. This is not taking advantage of polymorphism. Another alternative would be to put the prompts in the initializer or even a constructor for each class, but this would not allow the classes to be applied in a GUI or web application in the future. A better idea is to create a static method that does the prompting and returns a dictionary of the prompted parameters. Then, all the Agent has to do is prompt the user for the type of property and payment method, and ask the correct class to instantiate itself. That's a lot of designing! The following class diagram may communicate our design decisions a little more clearly:


Agent
+property_list
+list_properties(show_all=False)
+add_property(property_type,purchase_type)
Property
+square_feet
+num_bedrooms
+num_bathrooms
+display()
+prompt_init()
Apartment
+balcony
+laundry
+display()
+prompt_init()
House
+num_stories
+garage
+fenced_yard: boolean
+display()
+prompt_init()
HouseRental
HousePurchase
ApartmentPurchase
ApartmentRental
Purchase
+price
+taxes
+display()
+prompt_init()
Rental
+furnished
+utilities
+rent
+display()
+prompt_init()