CS 162P
Class Composition

Classes can contain variables of any other type within them. For example, they can have integers, strings, lists, or dictionaries. They can also include instances of other classes. When a class contains an instance of another class, this is referred to as a "has a" relationship. For example, a car "has a" engine.

Object Composition

Adding an object of another class to a given class as private variable is shown in the example below. In this example, the class Car "has a" driver that is an instance of the class Person.

```
class Person:
    def __init__(self, name = ""):
        self.__name = name

    def getName(self):
        return self.__name

    def setName(self, name):
        self.__name = name

class Car:
    def __init__(self, make, name):
        self.__make = make
        self.__driver = Person(name)

    def getDriver(self):
        return self.__driver.getName()

    def getMake(self):
        return self.__make

    def setDriver(self, name):
        self.__driver.setName(name)

    def setMake(self, make):
        self.__make = make
```

Composition and Initialization

When an object of type Car is instantiated, the __init__ method for Car is called. Since each Car contains an instance of type Person, it is necessary for the __init__ method for Person to be called to store any required information inside the Person. This is done when defining the variable __driver.

```
self.__driver = Person(name)
```

If no information was to be passed, then the default __init__ method is invoked with no parameters.

Composition and Member Access

Since the member variables of the included object (Person) are private, the class including it (Car) cannot access these varibles directly and must use the public accessor methods as shown in the getDriver method that calls getName on the __driver object.

```
    def getDriver(self):
        return self.__driver.getName()
```

The member methods of the Person object are accessed using the normal member access method (dot notation).

Composition Summary

When including an object of another class, it is possible to do this by defining a variable of that type. Since it is necessary to pass information to the included object, its __init__ can be called when creating it in the __init__ method for the including class. Finally, to access private member variables inside the included object, its public methods must be used and this requires the use of dot notation referencing the included object.