Homework 1: Answers to Questions

Jade Pearl

CMSC 335-7381 Object-Oriented and Concurrent Programming

Professor Andrew Seely

**Question 1**:

The program provided gives an error because the main method calls a no-argument constructor which does not exist in the MyClassA class. The class has a constructor with a parameter of int pV and there is no default/no-argument constructor. Here is a screenshot of the error:  
A screen shot of a computer program

Description automatically generated

The programmer may have intended to provide a default constructor and just forgot to do so. This can be fixed by providing a default constructor with no arguments:  
A computer screen shot of a program

Description automatically generated

Or the programmer can pass an integer argument in the method call:  
A screen shot of a computer program

Description automatically generated

**Question 2**:

In this program, the constructor is incorrectly defined:

A computer screen shot of a program

Description automatically generated  
It is written as public void MyClassB which is not how you should declare a constructor. The programmer probably thought of writing a different method. To fix this, delete the keyword void and there will no longer be errors:  
A screen shot of a computer program

Description automatically generated

**Question 3**:

In this initial program, MyClassC defines two variables, int v = 12 class variable and a local variable int v = pV in the constructor. This is an issue because the programmer most likely wished to change the value of the class variable, so creating a new int v locally in the constructor will cause the value of the class variable to not change. This could potentially lead to logic errors:

A screen shot of a computer program

Description automatically generated

To fix this, delete the int keyword before the v in the constructor which assigns the value of the class member variable itself as intended:

A screen shot of a computer program

Description automatically generated

**Question 4**:

In this problem, MyClassE carries out a normal constructor call. However, there is an error that says the MyClassF constructor that it calls is not visible. This problem occurred because the constructor is declared as private:

A computer screen shot of a program

Description automatically generated

The program intended for this constructor to be visible to and used by MyClassE. Declaring a method or constructor as private means that it can only be accessed within the class it is contained. In order to fix this, the private property of the constructor must be changed to public:

A screen shot of a computer program

Description automatically generated

**Question 5**:

The reason the following code works without any errors or warnings as compared to the previous programs is because it does not break the rules that the former programs did:

A computer screen shot of a program code

Description automatically generated

1. The program calls only constructors that exist (as opposed to question 1)
2. All constructors and methods are declared correctly (as opposed to question 2)
3. The class member variables are correctly defined and not duplicated as local variables in methods (as opposed to question 3)
4. Constructors and methods are assigned private or public properly depending on the use. The needed constructors/methods are accessible outside of the class it is defined in (as opposed to question 4)

**Question 6**:

The class hierarchy of

* DefenseDepartment
  + General
    - Private

is not a reasonable hierarchy. Parent/child relationships in classes as seen above typically indicate an “is-a” relationship. The ranks of General and Private are separate ranks in the DefenseDepartment, so it does not make sense to give General and Private a parent/child hierarchy. Looking at this logically as a real-life example, no-one realistically in military ranks can be a private AND a general at the same time. It would not make sense. The class hierarchy becomes reasonable when General and Private are separate entities and are both subclasses of DefenseDepartment:

* DefenseDepartment
  + General
  + Private

**Question 7**:

The following fields are reasonable for each of the following classes in the hierarchy. The fields are represented in bold.

* Vehicle – **manufacturer**: every vehicle has a person or company that manufactures it
  + Car – **gas**: a Boolean variable that dictates if a car can take gas or is electric and cannot take gas
  + Airplane – **maxAltitude**: the maximum altitude an airplane can reach
    - Passenger – **maxPassengers**: the max number of passengers that can fit on the passenger plane
    - Fighter – **gunPayLoad**: type and number of guns a fighter plane has
    - Bomber – **bombPayLoad**: type and number of bombs a bomber plane can carry
  + Spaceship – **launchDate**: can be a future or past date of when a spaceship is set to launch on a mission to space

**Question 8**:

The following methods are reasonable for each of the following classes in the hierarchy. The methods are represented in bold.

* Vehicle – **startVehicle()**: starts the vehicle’s engine
  + Car – **setAlarm()**: prepares the alarm to go off incase someone tries to break into the car
  + Airplane – **gainAltitude(height)**: makes the plane gain altitude/fly higher up
    - Passenger – **bookFlight(date, time, numPassengers, seats [])**: allows the user to book a flight
    - Fighter – **shootGuns(gunPayLoad)**: shoots the plane’s guns
    - Bomber – **dropBombs(bombPayLoad)**: drops the plane’s bombs
  + Spaceship – **blastOff()**: launches the spaceship into space

**Question 9**:

Encapsulation is defined by the use of data variables and methods/functions only in the class it is declared in, i.e., private variables and methods. An example I developed is a class Student:

A screen shot of a computer program

Description automatically generated

Inheritance is when a class, commonly a child class, inherits properties such as variables and methods from another class whether the variables are private or not. I made a class Undergraduate that gains inheritance from the Student class:

A computer screen shot of code

Description automatically generated

In spite of the name, year, and id variables of Student being private (only accessible by Student), the Undergraduate class extends the Student class which allows it to inherit the properties of the student class. This means the Undergraduate class has access to all the aspects of the Student class whether they are private or not.

**Question 10**:

For this question, I decided to make the Tree class the parent class because there are different types of trees. The class hierarchy I chose is as follows:

* Tree
  + Deciduous
  + Coniferous

This is a reasonable parent-child hierarchy of classes because there are different categories of trees and so having the class of Tree being a parent class is a good choice since there are also subcategories of trees. The children are Deciduous and Coniferous because they are the two main categories of trees that we commonly know of. Parent/child relationships typically represent an “is-a” relationship. Deciduous and Coniferous classes are perfect child classes to the Tree class because they are types of trees that have their own unique traits. For example, Tree can have the field of height because all trees have a height. When we get to Deciduous and Coniferous, we can go more into specific aspects. The field leafColor can be a unique identifier for Deciduous trees since they have leaves that change colors and fall in the cold seasons. Coniferous trees are trees that do not change color and have needles that fall over time. The field needleType can describe the kind of needles a tree bears such as pine needles.

**Resources**:

*Coniferous vs. Deciduous Trees Explained*. (2019, May 20). Clean Cut Tree & Landscape. <https://www.cleancuttrees.com/coniferous-vs-deciduous-trees/>