Eric Tweedie

CSC 6301

Module 3 Project

Code Analysis

The code that I used was generated by Google AI generative search when I had searched for multilevel inheritance with visibility java example. The code that was produced by the search was not overly extensive but it had the concepts over what the project was asking for.

The code seemed to adhere to the guidelines of encapsulation because it did make use of the modifiers (public, private, protected) within the code. The classes themselves were public since they are being inherited from parent to child. Although within the class Animal the main attribute that was used in this class was a protected attribute so it is only able to be accessed by the class itself and any other subclass that inherits the Animal class. In the Mammal class its main attribute was private and can only be accessed by the class and its methods. There is also a protected method within the Mammal class, which makes it only available to the class and then any other subclasses(child) of that class. The final class, Class Dog, also has a private attribute inside of it. The Dog class has a display info method that displays an output of what the name, breed, and if it can give birth when it is called. Now because the Mammal class has a private method it can not be accessed by the Dog class because it is only available to the Mammal class, so that method is only able to return its output from the Mammal class. Overall, the program appears to have a good sense of encapsulation because it is keeping the data variables that are specific to each class available to only itself or only available to the classes or methods that should have access to them. I believe this code has good modularity aspects because of the encapsulation of the code within its classes. I believe this to be true because the classes should be able to be broken down into their own modules because the data is contained within each class.