Polymorphism and Composition Homework

Polymorphism

1. What does the word 'polymorphism' mean?

*It is the ability to take on many forms, i.e. can pass more then one “is-a” test in Java.*

1. What does it mean when we apply polymorphism to OO design? Give a simple Java example.

*It is when an object can be thought of more than one form. If a car or a boat object implements an “IDrive” interface, then they can both be considered “IDrive” objects and be passed into an array accepting “IDrive” type.*

1. What can we use to implement polymorphism in Java?

*An interface.*

1. How many 'forms' can an object take when using polymorphism?

*As many as they want. An object can implement several interfaces therefore can be thought of as many different types.*

1. Give an example of when you could use polymorphism.

*If you want to have an array hold different types of transport such as planes, cars and boats we can implement the “IDrive” interface to them all and instantiate the array list to hold IDrive objects.*

Composition and Aggregation

1. What do we mean by 'composition' in reference to object-oriented programming?

*This is when a class is made up of other classes as it’s instance variables. i.e. Car “has-a” engine.*

1. When would you use composition? Provide a simple example in Java.

*Composition is used to make code cleaner, reusable and allows for alteration without affecting existing code. An example is Car class that is made up of an Engine class and a Tyre class. The Engine and Tyre class can be reused with other classes without affecting the Car class.*

1. Give a difference between composition and aggregation?

*Aggregation the child can exist without the parent however composition implies the child cannot exist without the parent. Students can still exist if the class is deleted, however a house cannot exist if the rooms are deleted.*

1. What is/are the advantage(s) of using composition/aggregation?

*Composition/Aggregation is more flexible than inheritance, you can be more selective with the properties and behaviours you want to pass, in addition you can implement several different interfaces therefore making code reusable and cleaner.*

1. When using composition, when an object is destroyed, what happens to all the objects it is composed of?

*The child class cannot exist without the parent class.*

1. When using aggregation, when an object is destroyed, what happens to all the objects it is composed of?

*The child class can still exist without the parent class.*