1. **Can a struct be part of an inheritance hierarchy?**

No, inheritance applies only to classes, not to structures. You cannot define your own hierarchy with structures, and you cannot define a structure that derives from a class or another structure.

1. **Why is it reasonable to declare a reference to a base class which refers to an object of a derived class (e.g., Mammal myMammal = new Horse();)?**

It is reasonable to declare a reference to a base class which refers to an object of a derived class because in this example, all Horses are Mammals, and the Horse is simply a special type of Mammal. It has everything that a mammal has with a few extra bits defined by any methods and fields added to the Horse class.

1. **What is the difference between calling a method declared new and a method declared override in a derived class when referring to an object polymorphically by its base class (as in the last question)?**

Overriding a method is a mechanism for providing different implementations of the same method—the methods are all related because they are intended to perform the same task, but in a class-specific manner. Hiding a method is a means of replacing one method with another—the methods are usually unrelated and might perform totally different tasks. Overriding a method is a useful programming concept; hiding a method is often an error. For example, in the last question, the compiler generates a warning message informing you that Horse.Talk hides the inherited method Mammal.Talk. Using the “new” keyword does not change the fact that the two methods are completely unrelated and that hiding occurs, but it tells the compiler “I know what I’m doing so stop showing me these warning messages.”

1. **In what circumstance would you prefer an extension method over inheritance?**

Sometimes using inheritance is not the most appropriate mechanism for adding new behaviors, especially if you need to quickly extend a type without affecting existing code. Using an extension method, you can extend an existing type (a class or structure) with additional static methods. These static methods become immediately available to your code in any statements that reference data of the type being extended.

1. **Declare an array named myArray of type int[] and length 64.**

int[] myArray = new int[64];

1. **Using a for loop, iterate through myArray assigning each element of the array to its index (e.g., index 0 should have value 0).**

for (int i = 0; i < myArray.Length; i++)

{

myArray[i] = i;

}

1. **Using a foreach loop, iterate through myArray printing each array element to the console.**

foreach (int num in myArray)

{

Console.WriteLine(num);

}

