Prob2 write up

The used of access modifiers in a parent class like private and protected can have an effect of the child class trying to the data that has either private or protected keyword. Since private made it so it’s only that object is available to the class that created it, it’s not possible for another class (even if this class is a child to the class which had private object) to access and use that object. On the other hand, if an object from the parent class is protected, it can still only be access by the class that created and only a child class from the same hierarchy (a class that inherit data from it’s parent) is allowed access to that object or as the saying goes “it stays in the family”. You can have interaction between parent and child class if the parent class has an abstract class modifier or keyword alongside a method created in its class for example, but the child class(s) in question need to have the override keyword when calling the method in question. When ever you set a property inside a class it’s set to that class and the access modifier is set to public meaning another class can have access to that data. If another class want to access the data in a property, they will have to create a new class object that share information from the class that data first came from in order to use it themselves.

The first constructor I called was from the Leader class have two string objects (Order and Location) that requires two strings as parameters for the constructor to execute and by using this class “GiveTheOrder” method to display the values of each string.

The second constructor from the Commander class have three string objects one is from the Commander class itself (Condition) while the other two string objects (Order and Location) are from its parent the Leader class, where the constructor requires three strings as parameters for the constructor to run. This is where base (Order, Location) comes into play as it taken from the Leader’s master constructor for the Commander’s constructor to use it without those objects being in the Commander’s field. However, since the object Order from the Leader’s class has an access modifier of private the Commander’s constructor can’t access it, so it won’t appear when you call the GiveTheOrder method from the Commander’s class. For the output it’ll display the values of Location and Condition. Even though this constructor requires three parameters, one of them (Order) won’t be used since it can’t be access due to its protection level otherwise the complier won’t run properly.

The third and final constructor from the Captain’s class requires four strings parameters in order this constructor to run. The object Rank came from the Captain’s class itself, while Condition object came from it’s parent class Commander while order and location came from the grandparent Leader. Again, since Order has the access modifier private no other class, including Captain, can access or even use it on their own class while Location, Condition, and Rank have a modifier of protected meaning any class in their hierarchy can access those objects. By using this class GiveTheOrder method, it’ll print out the value of Location, Condition, and Rank objects when called by Main() same with the other constructors.

Sources from the following website <https://stackoverflow.com/questions/3626690/protected-keyword-c-sharp>

Using the protected keyword allows the parent class to give access to its properties to the child class in a hierarchy where you want the child class(s) to share similar properties with the parent. This practice is good to facilitate reusing of codes through the protected keyword instead of copying code from one class to another which can be lots of work for typing the same code repeatedly. This helps minimize security risks as well when you find exposed code that was written wrong causing an error. Instead of just having code in one place where it’s easy to find and fix if it happens to go wrong, you would have to search for it which could take a lot longer.