**1) What is the difference between deep copy and shallow copy?**

-A deep copy is the process of creating a new object and then copying the fields of the current object to the newly created object to make a complete copy of the internal reference types. A shallow copy is creating a new object and then copying the value type fields of the current object to the new object.

**2) What is the value of a reference after you declare and initialize it?**

-If my understanding is correct, the value of a reference will have the same value as the original object after you declare and initialize it. IT HOLDS A SPOT IN MEMORY

**3) How do you declare a value type?**

-You declare a value type by using the associated keyword for the type of data that will be stored in that variable. These common data types are all examples of value types: char, int, float, long, double, so to declare them you state the data type followed by the variable name you are assigning to that data type. Variables of these types directly contain values. An example is “int first = 34;”.

**4) How do you declare a reference type?**

-You declare a reference type in the same way that you declare value types. The difference is that the variables do not actually contain any value, but a pointer to another memory location that holds the data. String, Array, class, and delegate are all reference data types. An example is “string firstName = “Tony”;”.

**5) Does C# allow you to assign NULL to a value type?**

-No, you can’t assign NULL to a value type.

**6) Can you assign a nullable value type to a non-nullable variable of the same type? Why or why not?**

-You cannot assign a nullable value type to a non-nullable variable of the same type because if you did it would cause a compiling error.

**7) What is the difference between the stack and the heap?**

-The stack is used for storing value data types and the heap is used for storing reference data types. The stack is an array of memory, which means that it is a LIFO structure. The heap is an area of memory where blocks are allocated to store certain kinds of data objects.

**8) What does it mean when we say that all classes are specialized types?**

-When we say that “all classes are specialized types” it means that all classes are actually objects that belong to the system.object class and that you can use system.object to create a variable that can refer to any reference type.

**9) What does ref do?**

-The *ref* keyword causes an argument to be passed by reference and not by value.

**10) What does out do?**

-The out keyword has two uses. As a parameter modifier it enables you to pass an argument to a method by reference rather than by value. It can also be used in generic type parameter declarations for interfaces and delegates, which specifies that a type parameter is covariant.

**11) Describe boxing and unboxing in your own words.**

-Boxing is the process of converting data types. For example, if you are converting a value type, which is referenced on the stack, to an object type whose data type is referenced on the heap, the system will allocate heap memory to copy the value of the value type to, and then refers to the new address on the heap, rather than on the stack. Unboxing is the inverse of this process; converting a value that resides on the heap to one that can be referenced on the stack from a value type variable.

**12) What does cast do?**

-Cast is an operation that checks whether converting an item of one type to another is safe before actually making the copy.