C# CH.8

Evan Blanton

1. Shallow A new object is created that has an exact copy of the values in the original object.  
   Deep: A deep copy copies all fields, and makes copies of dynamically allocated memory pointed to by the fields.
2. To be able to refer back to what you want to declare
3. Int I;
4. Int pizza = new pizza();
5. no
6. Yes because NULL doesn’t reference anything
7. Stack adds and takes off the last, heap takes out what came first   
   Heap memory is like a large pile of boxes strewn around a room rather than stacked neatly on top of one another. Each box has a label indicating whether it is in use. When a new object is created, the runtime searches for an empty box and allocates it to the object. The reference to the object is stored in a local variable on the stack. The runtime keeps track of the number of references to each box. (Remember that two variables can refer to the same object.) When the last reference disappears, the runtime marks the box as not in use, and at some point in the future it will empty the box and make it available.
8. D
9. When using a *ref* parameter, anything you do to the parameter you also do to the original argument because the parameter and the argument both reference the same data
10. you want the method itself to initialize the parameter.
11. Make a value a type, unbox you get the value
12. you can specify that, *in your opinion*, the data referenced by an object has a specific type and that it is safe to reference the object by using that type. The key phrase here is “in your opinion.”