# ToString And Arrays

**The goal for this exercise** is to make sure that you can override the ToString method, which is a standard method that all C# objects inherit.

**What you need to do for this exercise:**

1. In the provided starter project, in the ToString\_And\_Arrays class, you will see that there is code provided to you. You need to finish implementing that class, as described below.
2. Take two classes that you've previously created (say, a **MyPoint**, and a **FordTruck**). For each class, if the class doesn’t already have it’s own ToString method (which overrides the one provided by the **Object** base class), then you should add the ToString method to it.
(this can also be phrased as “if the class doesn’t already have it’s own, overridden version of the ToString method”)
3. Next, you should be able to get code similar to the following to work:

Object []objs = new Object[4];
objs[0] = new MyPoint(1,2);
objs[1] = new IncandescentLightBulb(10) ; // amps, maybe?
objs[2] = new MyPoint(4,7);
objs[3] = new IncandescentLightBulb(20) ; // amps, maybe?

for(int i = 0; i < objs.Length; i++)
{
 string s = objs[i].ToString();
 Console.WriteLine(s);
}

What’s important here is that
	1. You have an array (of type **Object**)
		1. Since all classes inherit, directly or indirectly, from **Object**, knowing that we can create an array of **Object** references, and then throw any sort of object at all into it, is a handy trick, and good to know about.
	2. You’ve filled the array with objects, of two different types (in this case, some objects are of the **MyPoint** type, some are of the **IncandescentLightBulb** type, but you’re welcome to use any two classes that you’ve previously created, or any two new classes that you want to create for this exercise)
	3. You’ve given the objects sufficiently unique data that you can tell which ‘**ToString’** method is which in the output ☺
	4. You’ve called **ToString** on each of the objects in the array