# Inheritance: Constructors

**The goal for this exercise** is to make sure that you can constructors and inheritance together.

Let's say that you have a base class defined as follows:

public class Base

{

int x;

public Base(int newX)

{

Console.WriteLine(“Base Class Constructor”);

x = newX;

}

}

If you want to create a derived class with a constructor, that calls the constructor of the base class, you can do so quickly and easily, like so:

public class Derived **: Base**

{

int y;

int z;

public Derived(int newX, int newY, int newZ) **: base(newX)**

{

Console.WriteLine(“Derived Class Constructor”);

y = newY;

z = newZ;

}

}

Notice that the way that we initialize the derived class is to pass the derived class’s constructor all the information that the derived class needs, *plus all the information the base class needs*, and then have the derived class's constructor simply 'forward the information along' the to the base class's constructor.

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

1. Within the starter project, create three classes: HomeElectronicsDevice, TV, and GameConsole. All three will have a price, as well as weight. Make sure that you set up an appropriate inheritance hierarchy, and that the two derived classes make good use of (reuse) data fields from the base class, where possible. Before you proceed on with this exercise, you may want to double-check that your choices for the base & derived classes make sense, perhaps with other students taking this course (using the Google Group).
   1. For these attributes, you’ll need to provide the normal Get/SetPrice and Get/SetWeight methods.
   2. The TV has a **ScreenSize** (which may not be a whole number), and the GameConsole has a **CPUSpeed**, measured in MHz (both of these need the standard getter/setter methods).
2. Additionally, add a constructor onto the base class that will initialize the price and weight fields of the base class using parameters provided to that constructor. The first parameter to the base class constructor must be the price, the second must be the weight.
3. Add constructors onto each of the derived classes that forward the needed data onto the base class, and then initialize the derived-class-specific instance variables. So for the TV class’s constructor, the first parameter should be price, the second should be weight, and the third should be the screen size.
   1. Optionally, add Print (and PrintTV, and PrintGameConsole) methods onto the classes, so that you can confirm that your constructors (and get/set methods) work correctly.
      1. Because these are optional, you’re free to format the output however you wish.
4. Within Constructors\_Calling\_One\_From\_Another. RunExercise(), create (at least) one instance of each class, and (optionally) print out the data fields (instance variables) of each object, in order to make sure that your code all works.