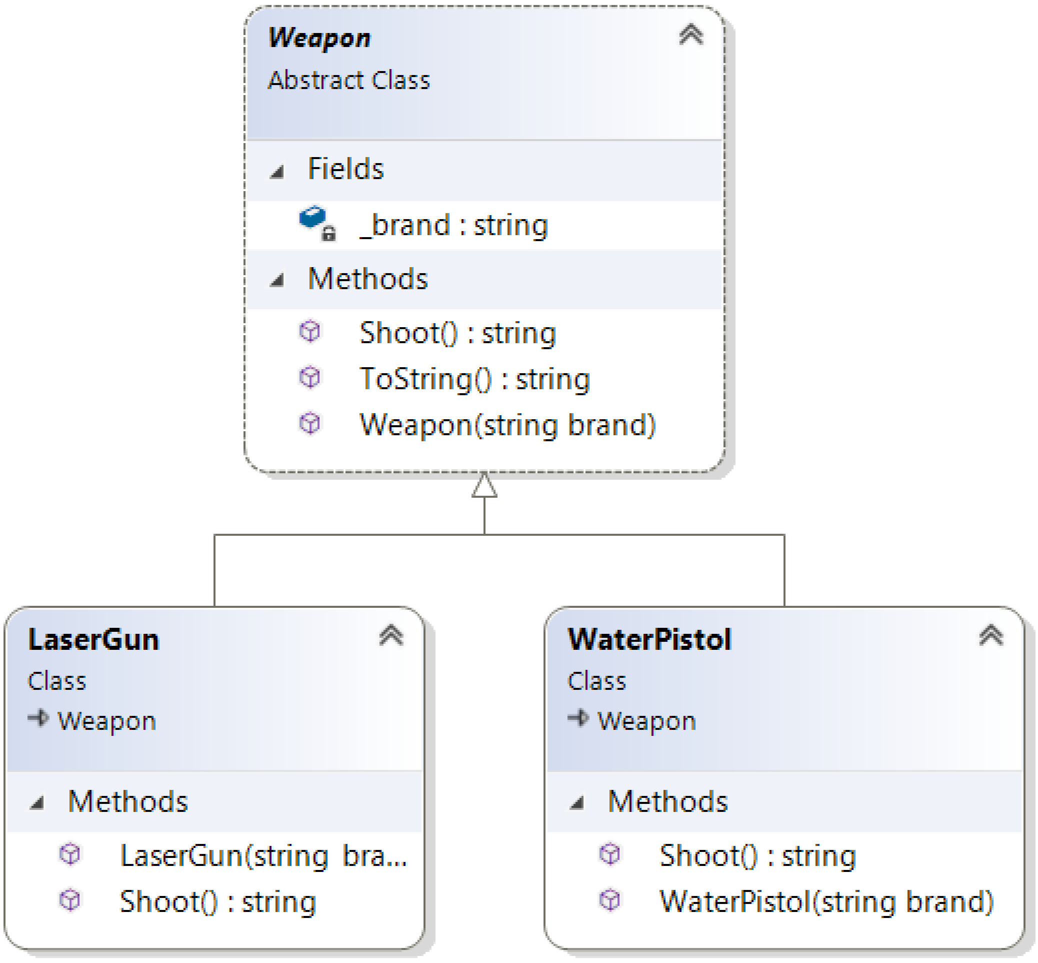
Exercise: Polymorphism

Goal: Implement a polymorphic shootout as discussed in the lecture.

Step 1:

Implement an abstract Weapon class with at least two subclasses as follows:



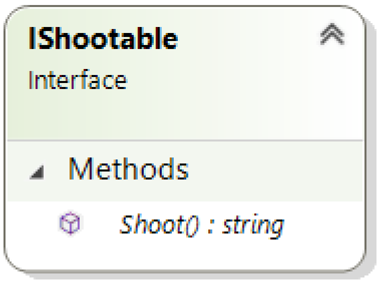
Example implementation of Shoot method (in LaserGun):

```
public override string Shoot()
{
    return $"Zing!! {base.Shoot()}";
}
```

Create a List of type <Weapon>, and construct and add some Weapons.

Loop through the list, calling Shoot() on each weapon.

Step 2:



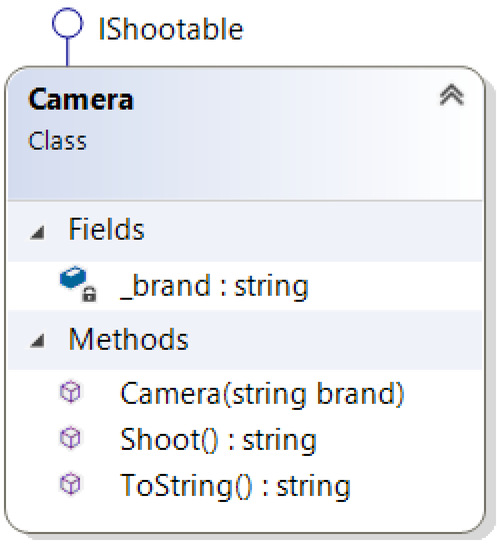
Create an interface called IShootable which conforms to the class diagram. Modify your Weapon class so it implements IShootable. Change the type of your List to be <IShootable>. Run your program again – the output should be unchanged.

Step 3:

Modify your Hunter class (not Person) to implement the IShootable interface. Construct a few Hunter objects and add them to be list. Run your program again and verify that it works.

Step 4:

Create a Camera class to conform to this class diagram. It should implement the IShootable interface. Construct and add a Camera to your list, verify that it works.



Step 5

Modify the Hunter class so that it no longer has a string field _camera, but instead has a public property Shooter, with default get; and set;

Modify the Hunter constructor so that it takes an IShootable object as the third parameter, rather than a string

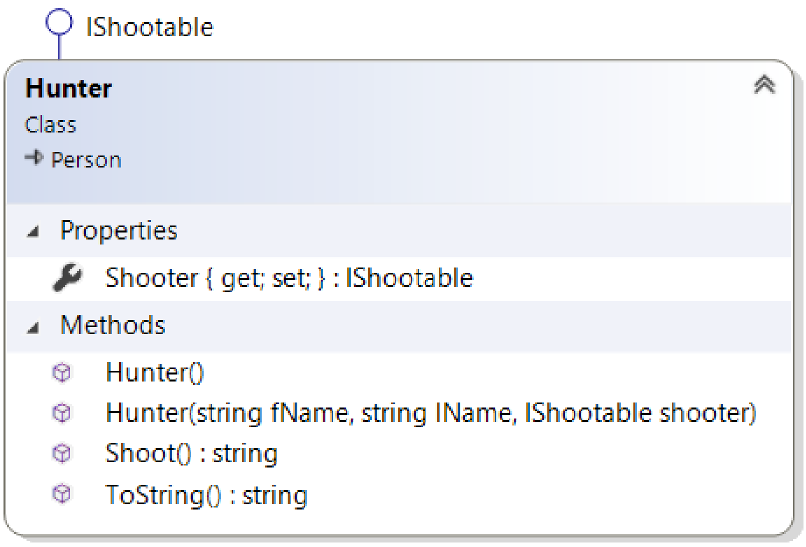
```
public Hunter(string fName, string lName, IShootable shooter) : base(fName, lName)
{
    Shooter = shooter;
}
```

Some of the existing code in your main method may break – either update it or comment it out.

Change the Hunter Shoot method so the Hunter calls the Shooter Shoot() method. (Hunters aren't born with the power to shoot, they need to use an object that can).

Run your program again, it should still work.

Step 6: Now you should be able to do a polymorphic shoot-out, as discussed in the lecture. Add code to your Main method to make it happen!


IShootable
Hunter
Class
Person
Properties
Shooter { get; set; } : IShootable
Methods
Hunter()
Hunter(string fName, string lName, IShootable shooter)
Shoot() : string
ToString() : string