1. Constructor Linking/Constructor Initializers

Create a class called Game which has the following properties

* Name:string
* Price:decimal
* ReleaseDate:DateTime

Create a constructor which has three parameters, one for each property

Through constructor linking create two other constructors which link to this main constructor. These constructors are shown below.

public Game(string name, decimal price):this(name, price, DateTime.Now)

public Game() : this("", 0)

Add a ToString() method which displays all properties.

1. Readonly

In the above code, change name so that it has a backing readonly attribute.

1. Object Initializer

In your Program.cs file create an object of type Game but use an object initializer in addition to one of the constructors you have provided.

Game g1 =

new Game("Monopoly", 19.99m, new DateTime(1970, 01, 31));

Game g2 =

new Game() { Price = 10.99m, ReleaseDate = new DateTime(2000, 6, 15) };

1. Protected

Extend the game class to a ComputerGame class. Add a PEGI rating property. Update the ToString() method to display PEGI rating. Add a method called GetDiscountPrice in the ComputerGame class

public decimal GetDiscountPrice()

{

return Price \* .9m;

}

Change Price to be private in the Game class. The method will no longer work as the Property is now not visible. Change Price to protected. You should now be able to access the property. Does this cause any problems in Program.cs?

1. Base calls to constructor and methods

In the ComputerGame class create a constructor which takes all values as parameters. Link this constructor to the main constructor in Game using the base keyword.

1. Override and virtual

Add a method called UpdatePrice in the Game class which will take a % and amends the price by this amount.

public void UpdatePrice(decimal percentageIncrease)

{

Price \*= (1 + percentageIncrease);

}

Override this method in the ComputerGame class. In addition to the behaviour of the base class this overridden method must also write to the console that it is updating the price from the ComputerGame class. Why will it not allow you to override, what needs to be changed?

1. Implicit conversion

Create two objects of type Game and two of type ComputerGame. Create a method in Program.cs with Game as a parameter. Pass the Game objects and ComputerGame objects to this method calling the ToString methods.

static void DisplayGame(Game game)

{

Console.WriteLine(game);

}

1. Abstract Class

Amend Game so that it is an abstract class. Amend the UpdatePrice method so that it is an abstract method and move functionality to the child class. How does this affect Program.cs

1. Extension method

In a separate Project create a program and write an extension method. The extension method should count the number of words in a sentence. Test that this works using several sentences of text**. Look at the recording for this at 28m 45s to understand what is meant.**