C#  
COLLECTIONS AND GENERICS

## Objective

In this lab you will practice using ArrayList, Queues and in a separate task, you'll work with a HashMap.

## Part 1 – Use a List<Shape>

1. Open the bouncing shape exercise that you did in the previous lab.
2. Change the array of Shapes (in Game class) to List<Shape>.  
   Use the List's **Add** method in the constructor of the Game class to add the three shapes. You can add these during declaration of the List but better to initialize these in a constructor.
3. Try your code and see if it still works.

## Part 2 - Using a Queue and a Stack

1. Back in the **labs** Console project, add a new static method to the Program class called **Lab5()**
2. Call the Lab5() method from Main. Comment out the other calls so we can test the code in this lab
3. Add another class to the **Labs** Console app called **ShoppingBasket**
4. Add the following fields to the **ShoppingBasket** class  
    **string productName;  
   int quantity;  
   double price;**
5. Create a constructor for the **ShoppingBasket** class to set the fields
6. Create a method called **DisplayDetails()** to display the fields' values.
7. In the Program class and above the Main() method, create a static Queue of   
    **ShoppingBasket**s called **baskets**.  
     
   Since we are going to use this queue in Lab5(), it needs to be static.
8. Create a **static** method in the Program class called **Buy**().
9. Call the **Buy()** method several times from the Lab5() method.
10. Write code in the **Buy()** method to add a few **ShoppingBasket**s to the **baskets** Queue. We will process these in another method.
11. Create another static method called **ProcessBaskets**().  
    In this method, write code to remove items from the **baskets** Queue and call their **DisplayDetails**() method just after they are removed.

This method simulates processing of shopping baskets for payment and shipping. For simplicity we just investigate the queue actions.

1. Call the **ProcessBaskets()** method in **Lab5()** once.
2. Run and test your code.

## Part 3 Using Dictionary<K,V>

**Scenario**: A Zoo has several animals and is expecting new arrivals soon. They wish to keep track of which animal types they have and record the count of each animal type.

You will create **Dictionary<string, int>**. The key (String) stores an animal’s type and a value (Integer) holds the count of the animal in the zoo (Like, **Lion 3**, **Zebra 2** etc).

### Step by step instructions.

1. Create a class called **Zoo**.
2. Create an instance of Zoo in **Lab5().**
3. Create two methods in the Zoo class called **Open()** and **DisplayAnimalData**()   
   both with no parameters or return value.
4. Declared and initialise the following fields in the **Zoo** class.

The following String arrays contain the names of the existing animals and the new animals we wish to add to our zoo.   
The **Disctionary** will keep track of the animals and their count.

**Disctionary** <string, Integer> ***animals*** = **null**;

**string[ ]** ***originalAnimals***= {"Zebra", "Lion", "Buffalo"};

**string[ ]** ***newAnimals***= {"Zebra", "Gazelle", "Buffalo", "Zebra"};

1. Instantiate the **Dictionary** of animals (new it) in the class constructor of the Zoo class or in the Open() method.
2. Create a method in Zoo as **void** **AddAnimal(string** animal**)**
3. In the Open() method, for each name in the ***originalAnimals,*** Call AddAnimal().
4. Do the same thing as above with the ***newAnimals***.
5. **void** **AddAnimal(string** animal**)** method should detect if the animal already exists in the ***animals*** Dictionary. If it does, it must increase its count by **1**, otherwise its count must be set to 1.

**Tip:** use the **ContainsKey** method of the Dictionary class to see if animal exists in the Dictionary.

You will also use the **Add**() method of the Dictionary class to put an entry back in the collection and use the **Dictionary**’s indexer (use it like an array) to get the count of an animal type using the name of an animal as the key. For example **animals[“Lion”]++;** will increase the count of Lions in the zoo.

1. Call the **DisplayAnimalData**(). Method from withing **Lab5()** to show all the animal names and their count.

Tip: Best use and enhanced for loop.

This method is going to display key/value pairs in 2 columns like this.

3, Zebra

1, Gazelle

1, Lion

2, Buffalo

1. Run and test your code

**\*\* End \*\***