# Using the List Collection

A list is an object that can hold objects. You may have a list of ints, or chars or even a number of custom objects. It is possible to have duplicates in a list. The list may grow or shrink as items are adding or removed. A list is a generic collection type. Other collections include **SortedList**, **LinkedList**, **Stacks**, **Queue** and **Dictionary**. These collections will be covered later in the course.

List is a generic collection that means at declaration you specify what type the collection will work with.

### Declaration and initialization of a list:

//Creating an empty list of ints

List<int> numbers = new List<int>();

//Creating a list with three doubles

List<double> radii = new List<double>(){ 1.0, 2.1, 3.6 };

//Creating a list with five strings

List<string> pms = new List<string>(){ "Trudeau", "Harper", "Martin", "Chretien", "Campbell" };

//Creating an list of rectangle

List<Rectangle> rectangles = new List<Rectangle>();

### Adding an item to a list:

The new item will be appended to the end of the list

numbers.Add(3); //numbers will have a single item

pms.Add("Mulroney"); //pms will now have six items

### Inserting an item to a list

This adds an item at a specified position

pms.Insert(2, "William Lyon Mackenzie King");

### Removing an item from a list

This method will remove an item from the list if it exists. If it does not exist, nothing happens.

pms.Remove("Mulroney"); //pms will now have six items

You may also remove an item at a specified location in a list.

radii.RemoveAt(1);

### Removing all items from a list

This method will remove all of the items from a list.

numbers.Clear(); //numbers is now empty

### Checking for the presence of an item in a list

This method will return true or false depending if the item is present or not.

pms.Contains("Narendra Pershad"); //returns false

pms.Contains("Trudeau"); //returns false

### The number of items in a list

The Count property will return the number of items in a list. This is similar to the Length property of an array.

numbers.Count;

### Traversing a list

You may use any type of loop to traverse a list, however the preferred way is using a foreach loop.

The foreach traversal is read-only. i.e. You may not insert or remove items in the loop body

for(int i = 0;i < numbers.Count;i++)  
{   
 Console.Write(numbers[i] + " ");  
}

foreach(int x in numbers)  
{   
 Console.Write(x + ", ");  
}

Console.WriteLine("\n\nDisplay names that are longer than 6 letters");  
foreach(string pm in pms)  
{  
 if(pm.Length > 6)  
 {  
 Console.Write(pm + ", ");//displays name that are longer than 6 letters  
 }  
}

Console.WriteLine("\n\nDisplay names that starts with C");  
foreach (string pm in pms)  
{  
 if (pm.StartsWith("C"))  
 {  
 Console.Write(pm + ", ");//displays name that are longer than 6 letters  
 }  
}