# 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

### Declaration and initialization of a list:

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

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

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);

pms.Add("Mulroney");

### 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.

numbers.Remove(3);

### Removing all items from a list

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

numbers.Clear();

### 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");

### 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.

for(int i = 0;i < numbers.Count;i++)

{

Console.Write(numbers[i] + " ");

}

foreach(int x in numbers)

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

{

Console.Write(x + " ");

}