**Part 83 - Generic stack collection class**

**Stack is a generic LIFO (Last In First Out) collection class** that is present in System.Collections.Generic namespace. The Stack collection class is analogous to a stack of plates. If you want to add a new plate to the stack of plates, you place it on top of all the already existing plates.If you want to remove a plate from the stack, you will first remove the one that you have last added. The stack collection class also operates in a similar fashion. The last item to be added (pushed) to the stack, will be the first item to be removed (popped) from the stack.

To insert an item at the top of the stack, use **Push**() method.  
  
To remove and return the item that is present at the top of the stack, use **Pop**() method.

A foreach loop iterates thru the items in the stack, but will not remove them from the stack. The items from the stack are retrieved in **LIFO**(Last In First Out), order. The last element added to the Stack is the first one to be removed.  
  
To check if an item exists in the stack, use **Contains**() method.  
  
**What is the difference between Pop() and Peek() methods?**  
Pop() method removes and returns the item at the top of the stack, where as Peek() returns the item at the top of the stack, without removing it.  
  
Let us undestand the Stack collection class with an example.  
public class Program  
{  
    public static void Main()  
    {  
        Customer customer1 = new Customer()  
        {  
            ID = 101,  
            Name = "Mark",  
            Gender = "Male"  
        };  
  
        Customer customer2 = new Customer()  
        {  
            ID = 102,  
            Name = "Pam",  
            Gender = "Female"  
        };  
  
        Customer customer3 = new Customer()  
        {  
            ID = 103,  
            Name = "John",  
            Gender = "Male"  
        };  
  
        Customer customer4 = new Customer()  
        {  
            ID = 104,  
            Name = "Ken",  
            Gender = "Male"  
        };  
  
        Customer customer5 = new Customer()  
        {  
            ID = 105,  
            Name = "Valarie",  
            Gender = "Female"  
        };  
  
        // Create a Stack  
        Stack<Customer> stackCustomers = new Stack<Customer>();  
        // To add an item to the stack, use Push() method.  
        // customer1 is inserted at the top of the stack  
        stackCustomers.Push(customer1);  
        // customer2 will be inserted on top of customer1 and now is on top of the stack  
        stackCustomers.Push(customer2);  
        // customer3 will be inserted on top of customer2 and now is on top of the stack  
        stackCustomers.Push(customer3);  
        stackCustomers.Push(customer4);  
        stackCustomers.Push(customer5);  
  
        // To retrieve an item from the stack, use Pop() method. This method   
        // removes and returns an object at the top of the stack. Since customer5

        // object is the one that is pushed onto the stack last, this object will be  
        // first to be removed and returned from the stack by the Pop() method  
        Customer c1 = stackCustomers.Pop();  
        Console.WriteLine(c1.ID + " -  " + c1.Name);  
        Console.WriteLine("Items left in the Stack = " + stackCustomers.Count);  
  
        Customer c2 = stackCustomers.Pop();  
        Console.WriteLine(c2.ID + " -  " + c2.Name);  
        Console.WriteLine("Items left in the Stack = " + stackCustomers.Count);  
  
        Customer c3 = stackCustomers.Pop();  
        Console.WriteLine(c3.ID + " -  " + c3.Name);  
        Console.WriteLine("Items left in the Stack = " + stackCustomers.Count);  
  
        Customer c4 = stackCustomers.Pop();  
        Console.WriteLine(c4.ID + " -  " + c4.Name);  
        Console.WriteLine("Items left in the Stack = " + stackCustomers.Count);  
  
        Customer c5 = stackCustomers.Pop();  
        Console.WriteLine(c5.ID + " -  " + c5.Name);  
        Console.WriteLine("Items left in the Stack = " + stackCustomers.Count);  
        Console.WriteLine("-----------------------------------------------------------");

        // After customer5 is removed, there will be no items left in the   
        // stack. So, let's push the five objects once again  
        stackCustomers.Push(customer1);  
        stackCustomers.Push(customer2);  
        stackCustomers.Push(customer3);  
        stackCustomers.Push(customer4);  
        stackCustomers.Push(customer5);  
  
        // If you need to loop thru items in the stack, foreach loop can be used in the same   
        // way as we use it with other collection classes. The foreach loop will only iterate

  // thru items in the stack, but will not remove them. Notice that the items from the   
        // stack are retrieved in LIFO (Last In First Out), order. The last element added to   
        // the Stack is the first one to be removed.  
        foreach (Customer customer in stackCustomers)  
        {  
            Console.WriteLine(customer.ID + " -  " + customer.Name);  
            Console.WriteLine("Items left in the Stack = " + stackCustomers.Count);  
        }  
        Console.WriteLine("-----------------------------------------------------------");  
  
        // To retrieve an item that is present at the top of the   
        // stack, without removing it use Peek() method.  
        Customer c = stackCustomers.Peek();  
        Console.WriteLine(c.ID + " -  " + c.Name);  
        Console.WriteLine("Items left in the Stack = " + stackCustomers.Count);  
        Console.WriteLine("-----------------------------------------------------------");  
  
        Customer c99 = stackCustomers.Peek();  
        Console.WriteLine(c99.ID + " -  " + c99.Name);  
        Console.WriteLine("Items left in the Stack = " + stackCustomers.Count);  
        Console.WriteLine("-----------------------------------------------------------");

     // To check if an item, exists in the stack, use Contains() method.  
        if (stackCustomers.Contains(customer1))  
        {  
            Console.WriteLine("customer1 is in stack");  
        }  
        else  
        {  
            Console.WriteLine("customer1 is not in stack");  
        }  
    }  
}  
  
public class Customer  
{  
    public int ID { get; set; }  
    public string Name { get; set; }  
    public string Gender { get; set; }  
}