Lab 1: Working with Generic Queue.

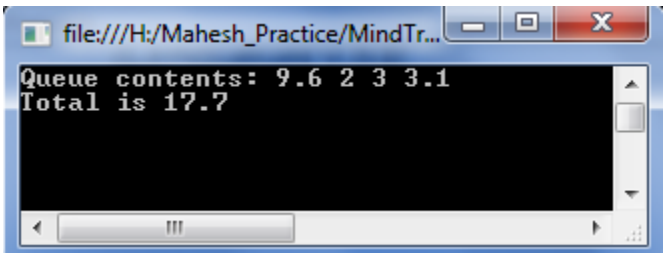
In this lab we will see how to make use of the Generic Queue.

Step 1: Open VS2008 and create a Console Application, name it as 'CS_GenericQueue'. The Queue provides Enqueue method to add elements in Queue and Dequeue method to read data from Queue.

Step 2: Write the below code for Main Method:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace CS_GenericQueue
{
    class Program
    {
        static void Main(string[] args)
        {
            //Create a Q and Enqueue data in it.
            Queue<double> q = new Queue<double>();
            q.Enqueue(9.6);
            q.Enqueue(2.0);
            q.Enqueue(3.0);
            q.Enqueue(3.1);

            double sum = 0.0;
            Console.Write("Queue contents: ");

            //Print the data from Queue using Deququq and calculate SUm
            while (q.Count > 0)
            {
                double val = q.Dequeue();
                Console.Write(val + " ");
                sum += val;
            }

            Console.WriteLine("\nTotal is " + sum);
            Console.ReadLine();
        }
    }
}
```

Step 3: Run the application, the result should be as below:



Lab 2: Working with List<T> for performing Insert, Update, Delete and Get operations.

In this lab we will see how the, Generic List class can be used for storing the data and how to perform the data manipulation and read operation on it.

Note: Please Read the GREEN Comments on the code. This comments the role and use of every class, method and the line in the program.

Step 1: Open VS2008 and create a new Console Application, name it as 'CS_GenericList'. In this class add the following Employee class:

```
/// <summary>
    /// The Data Class for Employee Information
    /// </summary>
    public class Employee
    {
        int _EmpNo;

        public int EmpNo
        {
            get { return _EmpNo; }
            set { _EmpNo = value; }
        }
        string _EmpName;

        public string EmpName
        {
            get { return _EmpName; }
            set { _EmpName = value; }
        }
        decimal _Salary;

        public decimal Salary
        {
            get { return _Salary; }
            set { _Salary = value; }
        }
        string _DeptName;

        public string DeptName
        {
            get { return _DeptName; }
            set { _DeptName = value; }
        }
    }
```

Step 2: Add a new IEmployeeBizAction interface in the project as below:

```
/// <summary>
    /// Interface for Employee Actions
    /// </summary>
    public interface IEmployeeBizAction
    {
```

```
        void CreateEmployee(Employee Emp);
        List<Employee> GetEmployees();
        Employee GetEmployeeByEmpNo(int EmpNo);
        void UpdateEmployee(Employee Emp);
        void DeleteEmployee(int EmpNo);
    }
```

Step 3: Add a new class 'EmployeeBizAction' in the project. This class implements 'IEmployeeBizAction' interface. This class define List<Employee> object to act as a data store for the Employee information. This class performs operations for performing data manipulation and read operations on the List<Employee>. The code is as below:

```
/// <summary>
    /// The Biz Action Class for Employee
    /// </summary>
    public class EmployeeBizAction : IEmployeeBizAction
    {
        //The Generic List<T> object used to store Employees.
        List<Employee> EmployeeList = new List<Employee>();
        /// <summary>
        /// Method to Create a new Employee in the List<Employee>
        /// </summary>
        /// <param name="Emp"></param>
        public void CreateEmployee(Employee Emp)
        {
            EmployeeList.Add(Emp);
        }

        /// <summary>
        /// Method to Return All Employees using List<Employee>
        /// </summary>
        /// <returns></returns>
        public List<Employee> GetEmployees()
        {
            return EmployeeList;
        }

        /// <summary>
        /// Method to Update Employee. This  accepts the Employee to be
Updated
        /// </summary>
        /// <param name="Emp"></param>
        public void UpdateEmployee(Employee Emp)
        {
            //Loop to Check wheather the matching EmpNo is present in the
List<Employee>
            //If yes the previous Employee record will be removed from the
List and the
            //Updated record will be added
            foreach (Employee Ep in EmployeeList)
            {
                if (Ep.EmpNo ==Emp.EmpNo)
                {
                    //Remove the matching EmpNo from the list
```

```
                    EmployeeList.Remove(Ep);
                    break;
                }
            }
            //Add the Updated Employee
            EmployeeList.Add(Emp);
        }

        /// <summary>
        /// Method to Delete the Employee based upon the EmpNo.
        /// It used the Remove method
        /// </summary>
        /// <param name="EmpNo"></param>
        public void DeleteEmployee(int EmpNo)
        {
            foreach (Employee Emp in EmployeeList)
            {
                if (Emp.EmpNo == EmpNo)
                {
                    EmployeeList.Remove(Emp);
                    break;
                }
            }
        }

        /// <summary>
        /// Method to check wheather the Employee us present in the
List<Employee>
        /// or not based upon EmpNo
        /// </summary>
        /// <param name="EmpNo"></param>
        /// <returns></returns>
        public Employee GetEmployeeByEmpNo(int EmpNo)
        {
            Employee emp = new Employee();
            foreach (var item in EmployeeList)
            {
                if (item.EmpNo == EmpNo)
                {
                    emp.EmpNo = item.EmpNo;
                    emp.EmpName = item.EmpName;
                    emp.Salary = item.Salary;
                    emp.DeptName = item.DeptName;
                    break;
                }
            }
            return emp;
        }
    }
```

Step 4: At the program class level declare the static instance of the EmployeeBizAction class as below:

```
static EmployeeBizAction objEmpBiz = new EmployeeBizAction();
```

Step 5: Write the following code in Program class and Main method:

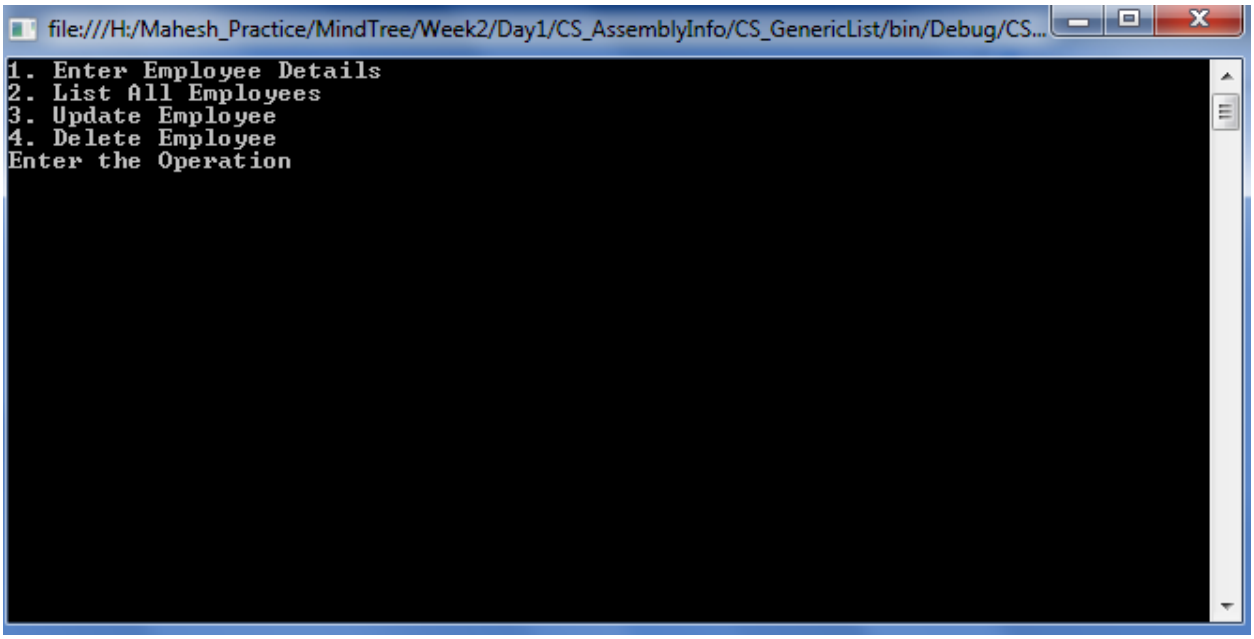
```
class Program
    {
        static EmployeeBizAction objEmpBiz = new EmployeeBizAction();
        static void Main(string[] args)
        {

            bool canContinue = true;
            while (canContinue)
            {
                Console.Clear();
                Console.WriteLine("1. Enter Employee Details");
                Console.WriteLine("2. List All Employees");
                Console.WriteLine("3. Update Employee");
                Console.WriteLine("4. Delete Employee");
                Console.WriteLine("Enter the Operation");
                int chioce = Convert.ToInt32(Console.ReadLine());

                switch (chioce)
                {
                    case 1:
                        Employee Emp = new Employee();
                        Console.WriteLine("Enter Employee Info");
                        Console.WriteLine();
                        Console.WriteLine("Enter EmpNo");
                        Emp.EmpNo = Convert.ToInt32(Console.ReadLine());
                        Console.WriteLine("Enter EmpName");
                        Emp.EmpName = Console.ReadLine();
                        Console.WriteLine("Enter Salary");
                        Emp.Salary = Convert.ToDecimal(Console.ReadLine() );
                        Console.WriteLine("Enter DeptName");
                        Emp.DeptName = Console.ReadLine();
                        objEmpBiz.CreateEmployee(Emp);

                        break;
                    case 2:
                        Console.WriteLine();
                        Console.WriteLine("List of All Employees");
                        List<Employee> lstEmp = objEmpBiz.GetEmployees() ;

Console.WriteLine("EmpNo\tEmpName\tSalary\tDeptName");
                        foreach (Employee ep in lstEmp)
                        {
                            Console.WriteLine(ep.EmpNo + "\t" + ep.EmpName +
"\t" + ep.Salary + "\t" + ep.DeptName);
                        }
                        break;
                    case 3:
                        Console.WriteLine("");
                        Console.WriteLine("Enter Employee No to be Updated");
                        int Eno = Convert.ToInt32(Console.ReadLine());
                        Employee Ep = objEmpBiz.GetEmployeeByEmpNo(Eno);
                        if (Ep.EmpName == string.Empty)
                        {
                            Console.WriteLine("Sorry! This employee is not
```

```
present please select the option again");
                        }
                        else
                        {
                            Console.WriteLine("The Old Details for EmpNo " +
Ep.EmpNo + " are as below");
                            Console.WriteLine("EmpNo " + Ep.EmpNo + " EmpName
" + Ep.EmpName + " Salary " + Ep.Salary + " DeptName " + Ep.DeptName);
                            Console.WriteLine("Please enter new values from
Salary and DeptName to Update");
                            Console.WriteLine("Enter New Salary");
                            Ep.Salary =
Convert.ToDecimal(Console.ReadLine());
                            Console.WriteLine("Enter new DeptName ");
                            Ep.DeptName = Console.ReadLine();
                            objEmpBiz.UpdateEmployee(Ep);
                            Console.WriteLine("Please List all Employees
(Option 2) to Make sure that Employee Updated");
                        }
                        break;
                    case 4:
                        Console.WriteLine();
                        Console.WriteLine("Enter EmpNo to Delete");
                        int eno = Convert.ToInt32(Console.ReadLine());
                        objEmpBiz.DeleteEmployee(eno);
                        Console.WriteLine("Please List all Employees (Option
2) to Make sure that Employee Deleted");
                        break;
                }
                Console.WriteLine("Press y or Y to Continue");
                string ct = Console.ReadLine();
                if (ct == "y" || ct == "Y")
                {
                    canContinue = true;
                }
                else
                {
                    canContinue = false;
                }
            }

            Console.ReadLine();
        }
    }
```

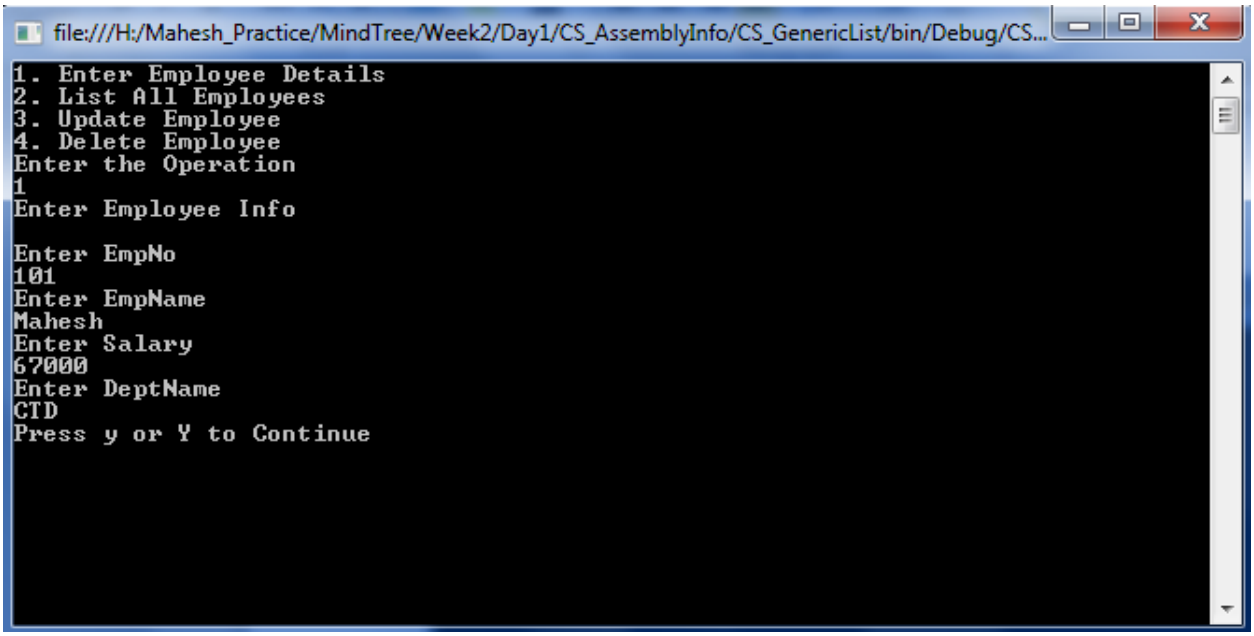
Step 6: Run the Application by entering your choices.

(**NOTE: The below screen-shots has the Sample DATA. If you Enter different EmpNo, EmpName, DeptName and Salary then the result will vary.**)

Press F5 the result will be as below:

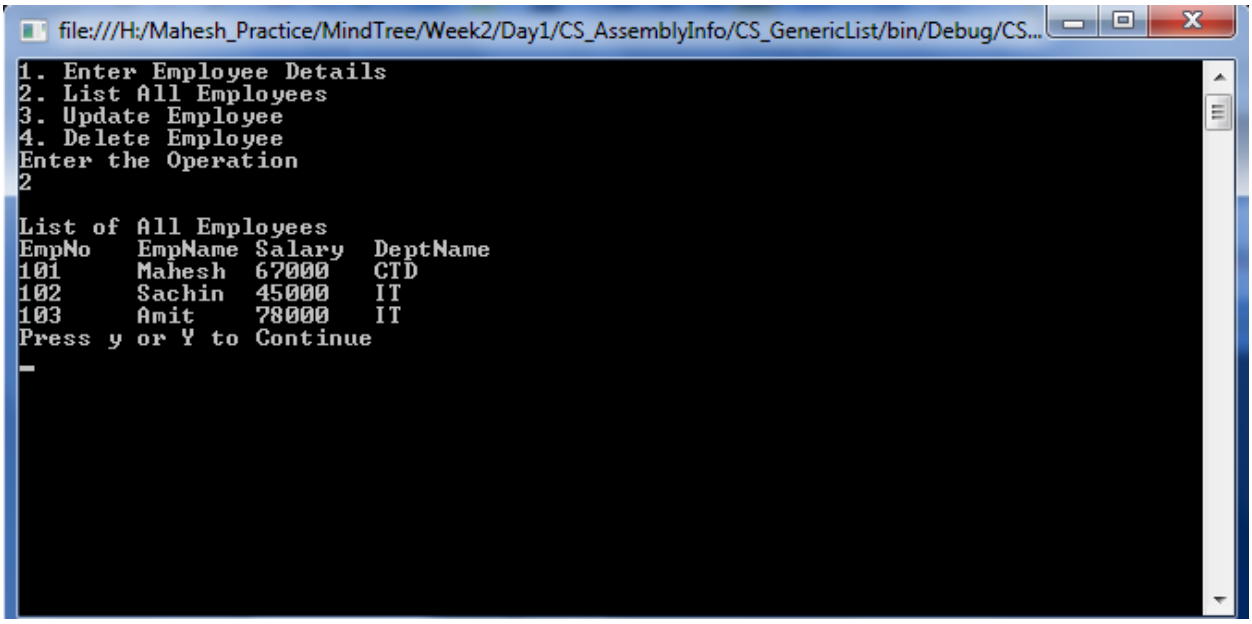


Enter 1, and enter Employee details as EmpNo, EmpName, Salary and DeptNo as below:



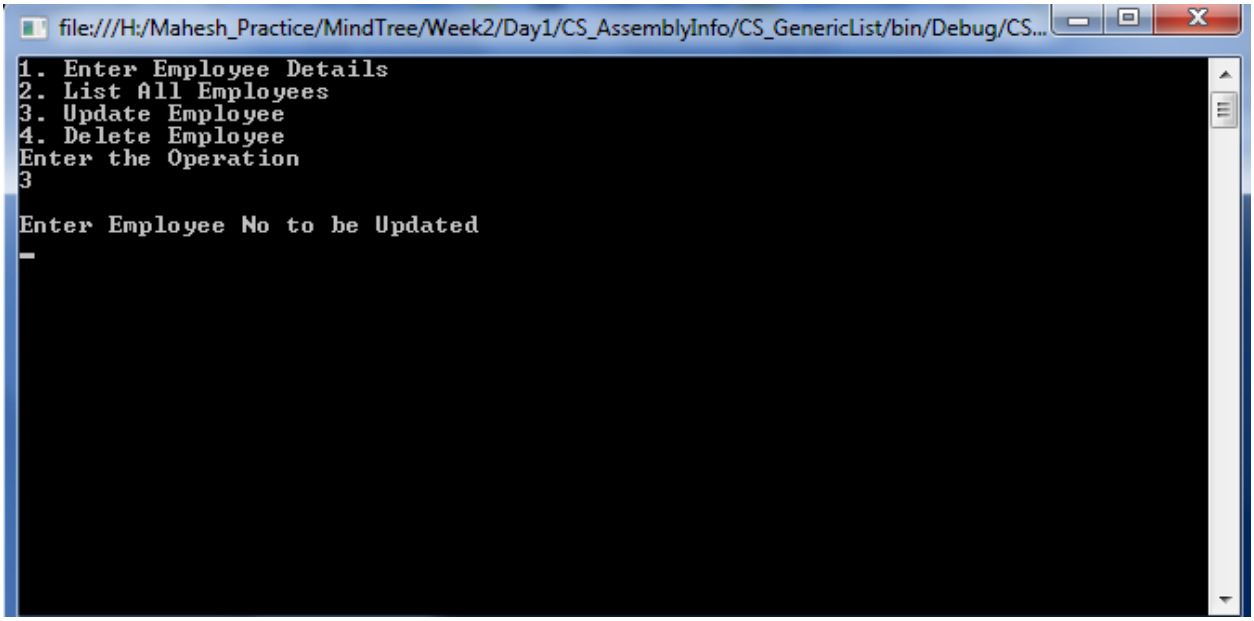
Enter “y” or “Y” to Continue and enter 3 to 4 Records.

Enter 2 to List All Employees the Result will be as bleow:

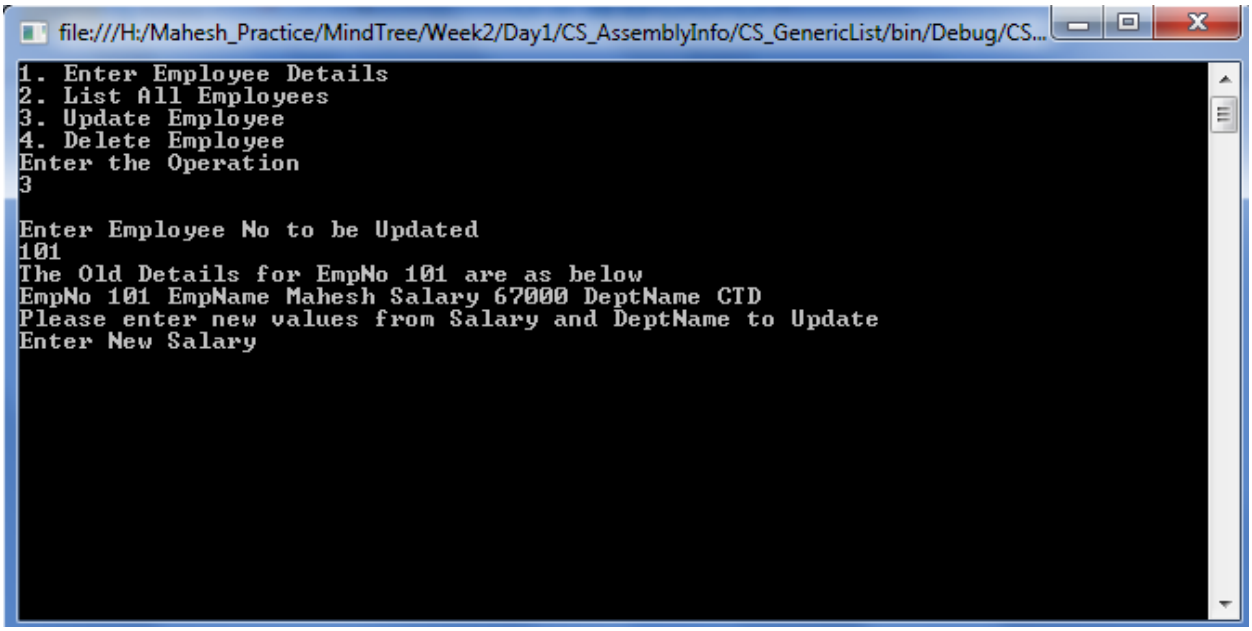


Enter “y” or “Y” to Continue

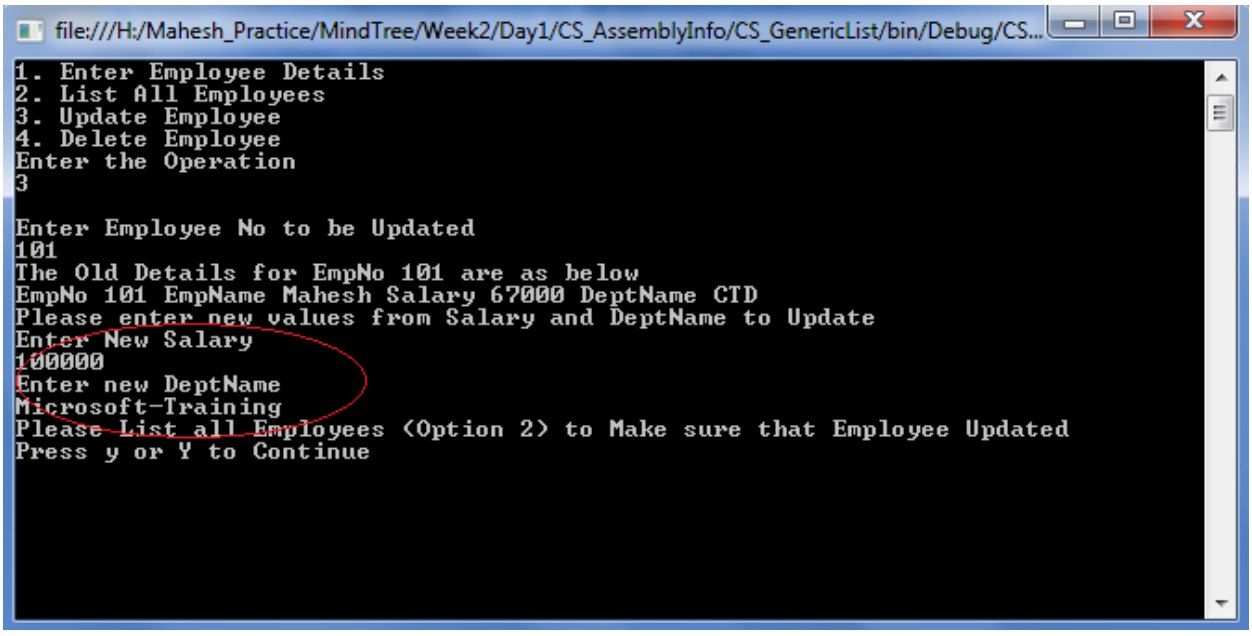
Enter 3 to Update an Employee, the result will be as below:



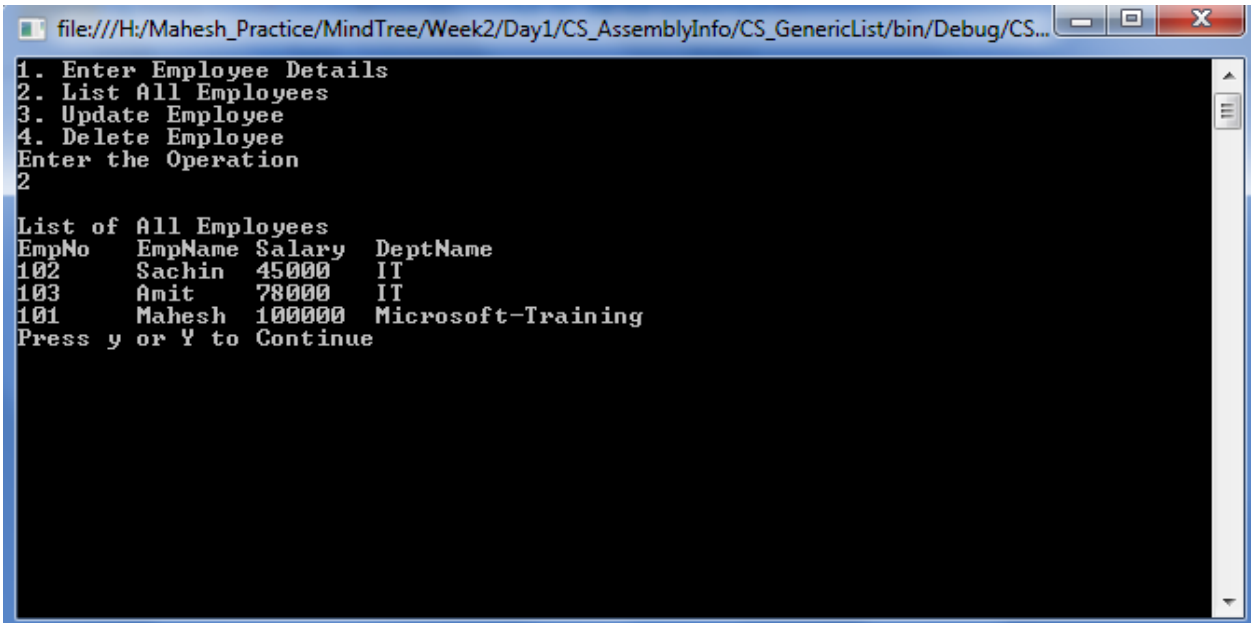
After Entering an EmpNo the Result will be as below:



Now Enter New Salary and DeptName as below:

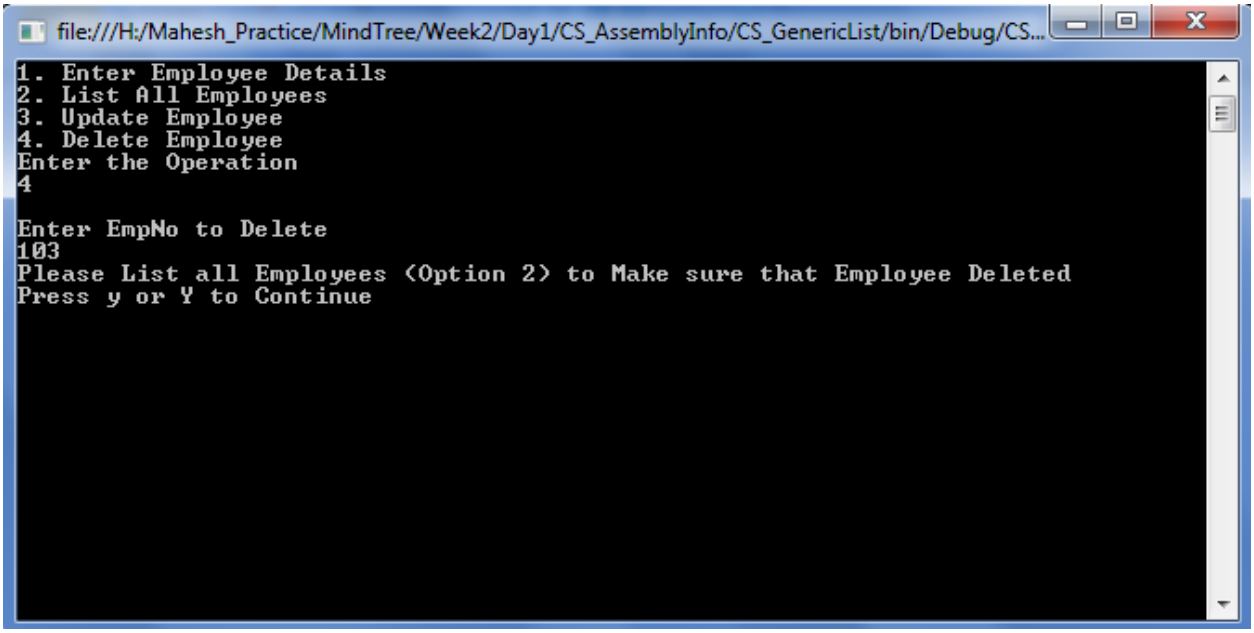


Enter ‘Y’ or ‘y’ to continue and press 2 to List all Employees to check wheather the record is updated or not as below:



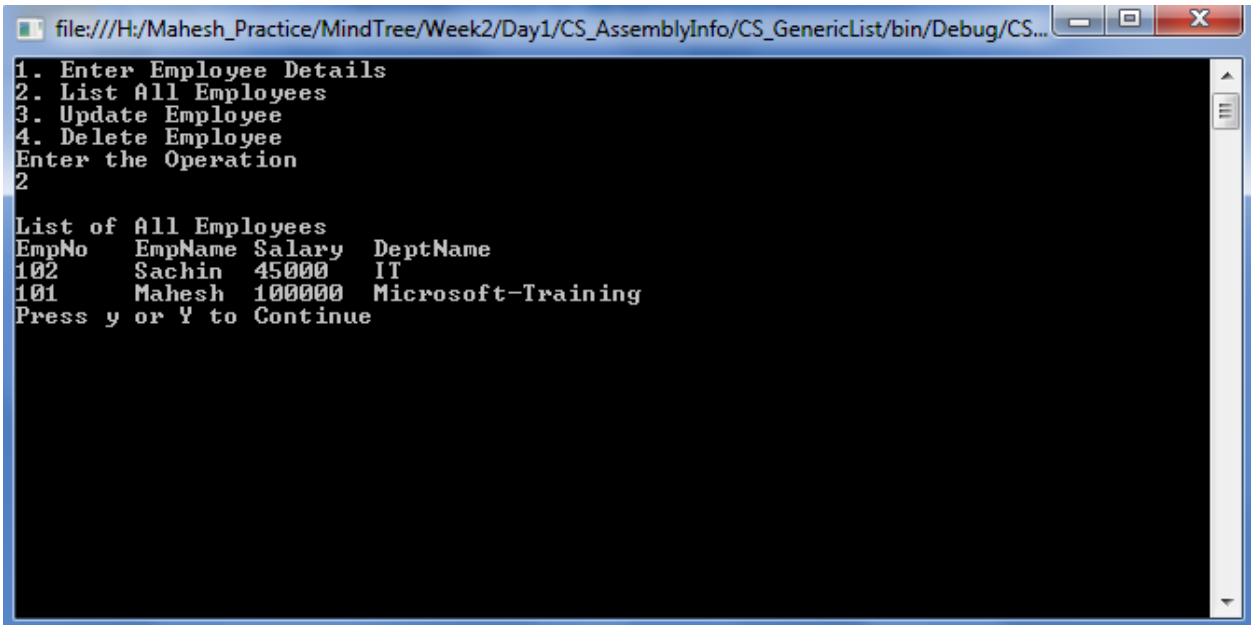
Enter ‘Y’ or ‘y’ to continue

Enter 4 to Delete a Record. To Delete the record you need to enter EmpNo as below:



Enter ‘Y’ or ‘y’ to continue

Enter 2 to Check the Record is Deleted or not. If deleted then the result will be as below:



Lab 3: Using Standard Dictionary class.

In this lab we will see how to make use of the Dictionary Class it is used for storing name value pair.

Step 1: Open VS2008 and create a Console Application, name it as 'CS_GenericCollection_Dictionary".

Step 2: Write the below code for Main Method:

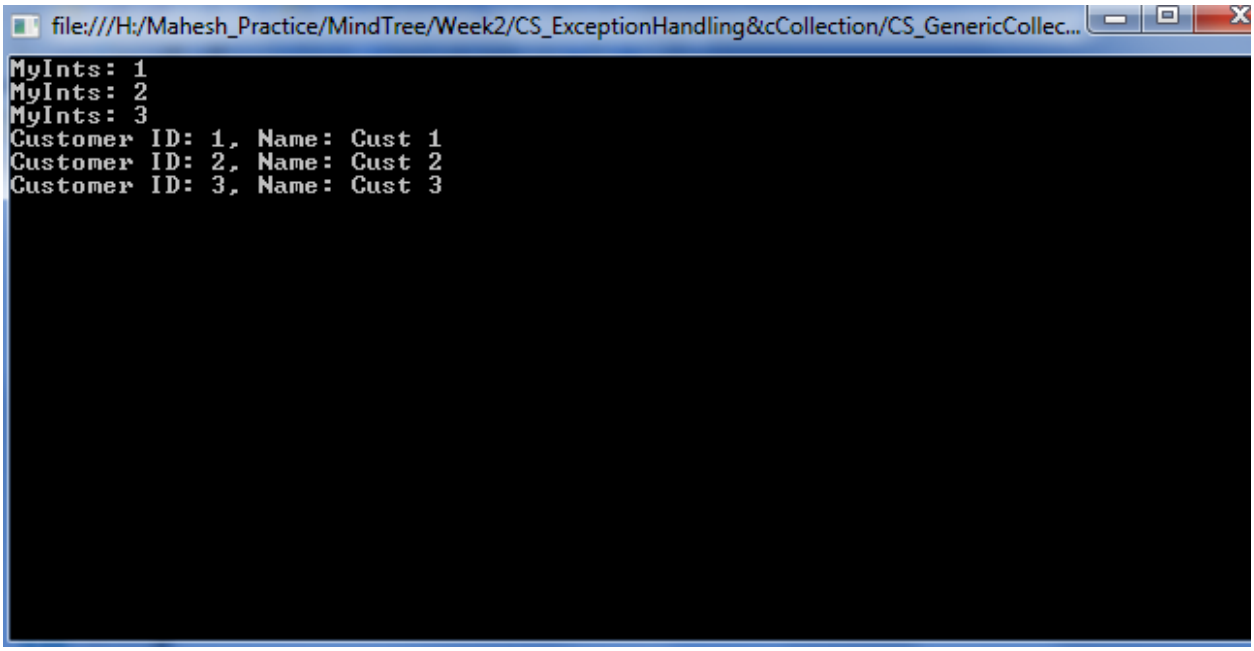
```
class Program
    {
        static void Main(string[] args)
        {
            List<int> myInts = new List<int>();

            myInts.Add(1);
            myInts.Add(2);
            myInts.Add(3);

            for (int i = 0; i < myInts.Count; i++)
            {
                Console.WriteLine("MyInts: {0}", myInts[i]);
            }

            Dictionary<int, Customer> customers = new Dictionary<int,
Customer>();

            Customer cust1 = new Customer(1, "Cust 1");
            Customer cust2 = new Customer(2, "Cust 2");
            Customer cust3 = new Customer(3, "Cust 3");
```

```
            customers.Add(cust1.ID, cust1);
            customers.Add(cust2.ID, cust2);
            customers.Add(cust3.ID, cust3);

            foreach (KeyValuePair<int, Customer> custKeyVal in customers)
            {
                Console.WriteLine(
                    "Customer ID: {0}, Name: {1}",
                    custKeyVal.Key,
                    custKeyVal.Value.Name);
            }

            Console.ReadKey();
        }
    }
```

Step 3: Create Class Customer

```
public class Customer
    {
        public Customer(int id, string name)
        {
            ID = id;
            Name = name;
        }

        private int m_id;

        public int ID
        {
            get { return m_id; }
            set { m_id = value; }
        }

        private string m_name;

        public string Name
        {
            get { return m_name; }
            set { m_name = value; }
        }
    }
```

Step 4 : Run the Press F5 the result will be as below:



Lab 4: Using Multi-Type Name value pairs.

In this lab we will see how to make use of the Multi Type Name value Pairs

Step 1: Open VS2008 and create a Console Application, name it as 'CS_Generic EmployeePairs".

Step 2: Write the below code for Main Method:

```
class Program
    {
        static void Main(string[] args)
        {

            EmployeePairs [] objEmp = new EmployeePairs[2];
            objEmp[0] = new EmployeePairs();
            objEmp[0].AddCollection(1001, "Mahesh");
            objEmp[1] = new EmployeePairs();
            objEmp[1].AddCollection(1002, "Ravi");


            for (int i = 0; i < 2; i++)
            {
                NameValuePair<int, string> EmpData =
objEmp[i].GetCollection();
                Console.WriteLine(EmpData.GetKey() + EmpData.GetValue());
            }
```

```
            Console.ReadLine();
        }
    }
```

Step 2 : Create a class NameValuePair <K,V>

```
public class NameValuePair<K, V>
    {
        private K key;
        private V value;


        public void SetKey(K key)
        {
            this.key = key;
        }

        public K GetKey()
        {
            return key;
        }

        public void SetValue(V value)
        {
            this.value = value;
        }

        public V GetValue()
        {
            return value;
        }
    }
```

The above class helps to store the data in a table format where the **key** part is used to identify a record which is stored using **value**.

Step 3 : Create a Class EmployeePairs

```
  public class EmployeePairs
    {
        NameValuePair<int, string> EmpKeyValue = new NameValuePair<int,
string>();
        public void AddCollection(int EmpNo,string EmpName)
```

```
        {
            EmpKeyValue.SetKey(EmpNo);
            EmpKeyValue.SetValue(EmpName);
        }

        public NameValuePair<int, string> GetCollection()
        {
            return EmpKeyValue;
        }
    }
```

Step 4 : Run the Press F5 the result will be as below: