1.Create following types ofarrays

* 1. Integer
  2. String

Use System.Arrayclass to perform following operations on them

Copy, Sort, Clear, Reverse

using System;

using System.Collections.Generic;

using System.Text;

namespace que1

{

class ArrayOperations

{

public void integer()

{

Console.WriteLine("Enter Size of an Array:");

int size = int.Parse(Console.ReadLine());

int[] arr = new int[size];

int[] arr2 = new int[size];

Console.WriteLine("Enter Elements to the array:");

for (int i = 0; i < size; i++)

{

arr[i] = int.Parse(Console.ReadLine());

}

Array.Copy(arr, arr2, size);

Console.WriteLine("Copy Elements:");

foreach (int array in arr)

{

Console.WriteLine(array);

}

Array.Sort(arr);

Console.WriteLine("Sort Elements");

foreach (int a in arr)

{

Console.WriteLine(a);

}

Array.Clear(arr, 2, 2);

Console.WriteLine("Clear Elements:");

foreach (int n in arr)

{

Console.WriteLine(n);

}

Array.Reverse(arr);

Console.WriteLine("Reverse Elements:");

foreach (int b in arr)

{

Console.WriteLine(b);

}

}

public void String()

{

Console.WriteLine("Enter Size of an Array:");

int size = int.Parse(Console.ReadLine());

string[] str = new string[size];

string[] str2 = new string[size];

Console.WriteLine("Enter Strings to the array:");

for (int i = 0; i < size; i++)

{

str[i] = Console.ReadLine();

}

Array.Copy(str, str2, size);

Console.WriteLine("Copy Strings:");

foreach (string array in str)

{

Console.WriteLine(array);

}

Array.Sort(str);

Console.WriteLine("Sort Strings:");

foreach (string a in str)

{

Console.WriteLine(a);

}

Array.Clear(str, 1, 2);

Console.WriteLine("Clear Strings:");

foreach (string n in str)

{

Console.WriteLine(n);

}

Array.Reverse(str);

Console.WriteLine("Reverse Strings:");

foreach (string b in str)

{

Console.WriteLine(b);

}

}

public static void Main(string[] args)

{

ArrayOperations p = new ArrayOperations();

p.integer();

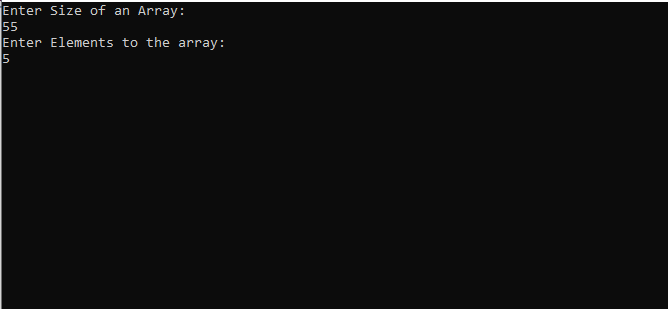
Console.WriteLine("---------------");

p.String();

}

}

}



2. Use collection class such as ArrayListto hold more than one employee objects in Employee Management application. Display all Employee details which are stored in collection.

using System;

using System.Collections.Generic;

namespace EmployeeManagementApplication

{

public class Demo

{

public static void Main()

{

List<object> arr1 = new List<object>();

arr1.Add("1");

arr1.Add("mona");

arr1.Add("4000");

Console.WriteLine("Initial List ...");

foreach (object i in arr1)

{

Console.WriteLine(i);

}

object[] arr2 = new object[4];

arr2[0] = 2;

arr2[1] = "Shital";

arr2[2] = 5000;

arr2[3] = 80;

arr1.InsertRange(3, arr2);

Console.WriteLine("After adding elements ...");

foreach (object i in arr1)

{

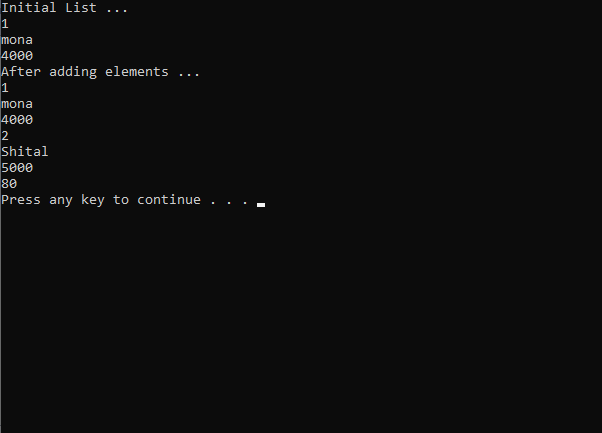
Console.WriteLine(i);

}

}

}

}



3 Write a console based program to create a linked list of Employee objects using the generic class List<>.Perform following operations on thelist:

1. Add a newemployee
2. Display the list ofemployees.
3. Total number of employees in thelist

using System;

using System.Collections.Generic;

namespace que2

{

class Program

{

static void Main(string[] args)

{

Employee employee1 = new Employee()

{

EmpID = 1,

EmpName = "Krushna",

EmpSalary = 35000

};

List<Employee> employees = new List<Employee>(1);

employees.Add(employee1);

foreach (Employee c in employees)

{

Console.WriteLine("ID={0}, Name={1}, Salary={2}", c.EmpID, c.EmpName, c.EmpSalary);

}

again:

Console.WriteLine("do you want to add emoployee---yes or no");

string choice = Convert.ToString(Console.ReadLine());

if (choice.ToUpper() == "YES")

{

Employee employeen = new Employee();

Console.WriteLine("enter your employee id");

employeen.EmpID = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("enter employee name");

employeen.EmpName = Console.ReadLine();

Console.WriteLine("enter employee salary");

employeen.EmpSalary = Convert.ToInt32(Console.ReadLine());

employees.Add(employeen);

goto again;

}

else

{

Console.WriteLine("total no.of employees =" + employees.Count);

}

Console.WriteLine("total no.of employees =" + employees.Count);

foreach (Employee c in employees)

{

Console.WriteLine("ID={0}, Name={1}, Salary={2}", c.EmpID, c.EmpName, c.EmpSalary);

}

}

}

class Employee

{

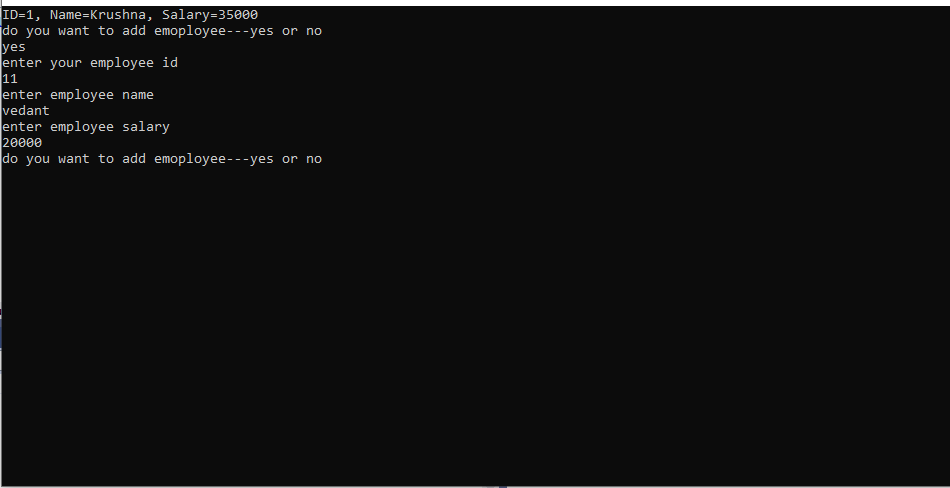
public int EmpID { get; set; }

public string EmpName { get; set; }

public int EmpSalary { get; set; }

}

}



4. Write Custom Generic class MyStackbased on assignment of previous session,with

Push()and Pop()methods to store any kind of .NET Type.

using System;

using System.Collections.Generic;

namespace GenericStack

{

class MyStack

{

public static void Main()

{

try

{

Stack<object> numbers = new Stack<object>();

numbers.Push("one");

numbers.Push("2");

numbers.Push("three");

numbers.Push("4");

numbers.Push("five");

// A stack can be enumerated without disturbing its contents.

foreach (object number in numbers)

{

Console.WriteLine(number);

}

Console.WriteLine("\nPopping '{0}'", numbers.Pop());

Console.WriteLine("Peek at next item to destack: {0}",

numbers.Peek());

Console.WriteLine("Popping '{0}'", numbers.Pop());

// Create a copy of the stack, using the ToArray method and the

// constructor that accepts an IEnumerable<T>.

Stack<object> stack2 = new Stack<object>(numbers.ToArray());

Console.WriteLine("\nContents of the first copy:");

foreach (string number in stack2)

{

Console.WriteLine(number);

}

// Create an array twice the size of the stack and copy the

// elements of the stack, starting at the middle of the

// array.

string[] array2 = new string[numbers.Count \* 2];

numbers.CopyTo(array2, numbers.Count);

// Create a second stack, using the constructor that accepts an

// IEnumerable(Of T).

Stack<object> stack3 = new Stack<object>(array2);

Console.WriteLine("\nContents of the second copy, with duplicates and nulls:");

foreach (object number in stack3)

{

Console.WriteLine(number);

}

Console.WriteLine("\nstack2.Contains(\"four\") = {0}",

stack2.Contains("four"));

Console.WriteLine("\nstack2.Clear()");

stack2.Clear();

Console.WriteLine("\nstack2.Count = {0}", stack2.Count);

}

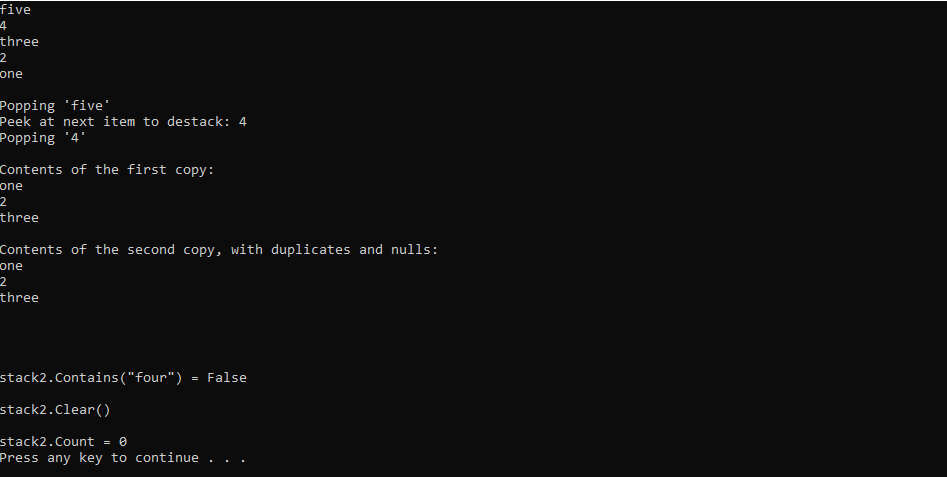
catch (Exception ex)

{ Console.WriteLine(ex.Message); }

}

}

}



5. Create a class named Player that contains Player name, runs scored as data members. Create a class named Teamthat contains an array of Player. Implement IEnumerableinterface for classTeam.

using System;

using System.Collections;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace IEnumerableExample

{

class Player

{

public string Name { get; set; }

public int Run { get; set; }

public Player(string name, int run)

{

Name = name;

Run = run;

}

}

class Team : IEnumerable

{

private Player[] playerArray = new Player[4];

public Team()

{

playerArray[0] = new Player("Virat", 48);

playerArray[1] = new Player("Dhoni", 90);

playerArray[2] = new Player("Rohit", 84);

playerArray[3] = new Player("Yuvraj", 39);

}

public IEnumerator GetEnumerator()

{

foreach (Player player in playerArray)

{

Console.WriteLine("Name is {0} and Run is {1}", player.Name, player.Run);

}

return playerArray.GetEnumerator();

}

}

internal class Program

{

public static void Main(string[] args)

{

Team India = new Team();

India.GetEnumerator();

Console.ReadLine();

}

}

}

