

Day 6 Morning Assignment

By [Manoj Karnatapu](#) - NBHealthTech

Assignment 1

Write a C# Code, Create an Array List with 5 items and Find Sum

Code

```
using System;
using System.Collections;

// Author: Manoj.Karnatapu
// Purpose : To Declare ArrayList, assign some Values & Find Sum/

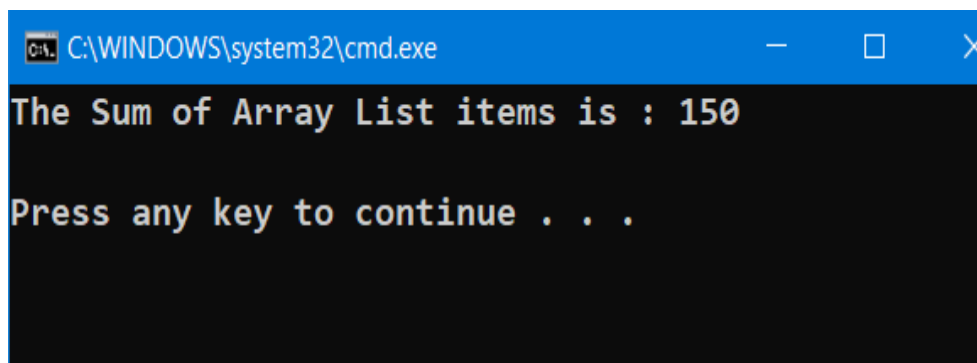
namespace Day6Project1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            ArrayList data = new ArrayList();

            int sum = 0;

            data.Add(10);
            data.Add(20);
            data.Add(30);
            data.Add(40);
            data.Add(50);

            foreach(var d in data)
            {
                sum = sum + (int)d;
            }
            Console.WriteLine("The Sum of Array List items is : {0}", sum);
            Console.ReadLine();
        }
    }
}
```

Output



The screenshot shows a Windows command prompt window with the title bar 'C:\WINDOWS\system32\cmd.exe'. The window contains the following text: 'The Sum of Array List items is : 150' and 'Press any key to continue . . .'. The text is displayed in a monospaced font on a black background.

Assignment 2

Research & find, how the values of an Array List are stored in the Memory ?

Answer

A: A System.Collections.ArrayList object is a sophisticated version of an array. The ArrayList class provide some features that are offered in most System.Collections classes but that are not in the Array class. For example:

The capacity of an Array is fixed, whereas the capacity of an ArrayList is automatically expanded as required. If the capacity of an ArrayList is changed, the memory reallocation and copying of elements are automatically done.

ArrayList provide methods that add, insert, or remove a range of elements. In Array, you can get or set the value of only one element at a time.

ArrayList is a Reference Type, but not Typesafe and less efficient

ArrayList is array-based structures. Therefore they will have a size something like: Capacity*sizeof(T). Since ArrayList stores things internally in an object[] it will have an additional pointer reference for each item.

Assignment 3

What are the dis-advantages of Array List (Collections Array List)?

Answer

A: The Non-generic collection classes such as ArrayList operate on **object** data type. As they operator on object data type, hence they are loosely typed (Which means we can store any type of value into the collection.).

We Have two Disadvantages of ArrayList in C#:

1. Because of this Loosely typed nature, If we have a chance of assigning Wrong Data Type. We have a chance of following into Runtime errors. These runtime errors are difficult to find and solve by the developers.
2. Each and Every Time we have to Unbox the items in an ArrayList.

Example Program :

```
namespace CollectionDemo
{
    class Program
    {
        static void Main(string[] args)
        {
            ArrayList Numbers = new ArrayList(2);
            // Boxing happens - Converting Value type (100, 200) to reference type
            // Means Integer to object type
            Numbers.Add(100);
            Numbers.Add(200);
            // Unboxing happens - 100 and 200 stored as object type
            // now conveted to Integer type
            foreach (int Number in Numbers)
            {
```

```

        Console.WriteLine(Number + " ");
    }
    Console.ReadKey();
}
}
}
}

```

Assignment 4

Write a C# Code, Declare a List<int> with 5 values and Find Sum

Code

```

using System;
using System.Collections.Generic;

//Author : Manoj.Karnatapu
//Purpose : Declare a List<int> with 5 values & find Sum

namespace Day6Project2
{
    internal class Program
    {
        static void Main(string[] args)
        {
            List<int> data = new List<int>();

            int sum = 0;

            data.Add(5);
            data.Add(10);
            data.Add(20);
            data.Add(30);
            data.Add(50);

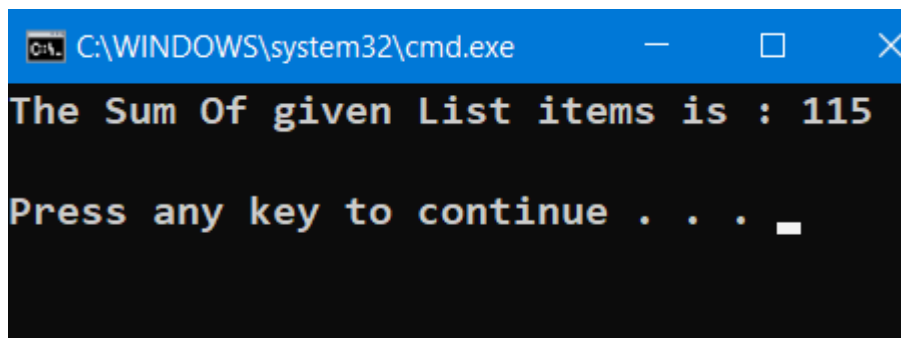
            foreach(int i in data)
            {
                sum = sum + i;
            }

            Console.WriteLine("The Sum Of given List items is : {0}", sum);

            Console.ReadLine();
        }
    }
}

```

Output



The screenshot shows a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The output of the program is displayed in white text on a black background. The first line shows "The Sum Of given List items is : 115". The second line shows "Press any key to continue . . . " followed by a cursor.

Assignment 5

Write the Differences between Collections and Generics.

Answer

A:

Collections	Generic
1. To overcome The Draw Backs of Arrays, we use Collections.	1) To Overcome the DrawBacks of Collections, we use Generics.
2. namespace: System.Collections	2) namespace: System.Collections.Generic
3. Each element is of Object data type in collections.	3) Each element is of Specified Data type during initialization.
4. There is a Need of Type Casting in Collections.	4) No need of any Type Casting in Generics.
5. Syntax: ArrayList data = new ArrayList();	5) Syntax: List<type> data = new List<type>(); Here <type>, indicates any DataType.
6. Example: <pre>using System; using System.Collections; namespace Day6Project2 { internal class Program { static void Main(string[] args) { ArrayList Numbers = new ArrayList(2); // Boxing happens - Converting Value type (100, 200) to reference type // Means Integer to object type Numbers.Add(100); Numbers.Add(200); // Unboxing happens - 100 and 200 stored as object type // now conveted to Integer type foreach (int Number in Numbers) { Console.Write(Number+" "); } Console.ReadKey(); } } }</pre>	6) Example: <pre>namespace Day6Project2 { internal class Program { static void Main(string[] args) { List<int> data = new List<int>(); int sum = 0; data.Add(5); data.Add(10); data.Add(20); data.Add(30); data.Add(50); foreach(int i in data) { sum = sum + i; } Console.WriteLine("The Sum Of given List items is : {0}", sum); Console.ReadLine(); } } }</pre>

Assignment 6

Research & find, how the values of List<T> are stored in the Memory ?

Answer

A: A System.Collections.Generic.List<T> object is a sophisticated version of an array. The List<T> generic class provide some features that are offered in most System.Collections classes but that are not in the Array class. For example:

The capacity of an Array is fixed, whereas the capacity of a List<T> is automatically expanded as required. If the capacity of a List<T> is changed, the memory reallocation and copying of elements are automatically done.

List<T> provide methods that add, insert, or remove a range of elements. In Array, you can get or set the value of only one element at a time.

List<T> or Generic list is a Reference Type, but is Type Safe and efficient

List<T> is array-based structures. Therefore they will have a size something like: Capacity*sizeof(T). Since ArrayList stores things internally in an object[] it will have an additional pointer reference for each item (if the T is a value type).

Assignment 7

Write a C# Code, declare a List<string> with 5 values from user and print using for, foreach & Lambda

Code

```
using System;
using System.Collections.Generic;

// Author : Manoj.Karnatapu
//Purpose : Declare a List<string> with 5 values & print values using for_loop, foreach &
Lambda Expression

namespace Day6Project3
{
    internal class Program
    {
        static void Main(string[] args)
        {
            List<string> data = new List<string>();
            data.Add("Manoj");
            data.Add("Sai");
            data.Add("Vihar");
            data.Add("Pavan");
            data.Add("Sharath");

            // Printing Values Using For Loop
            Console.WriteLine("\nPrinting The Items of the List, Using For Loop : ");
            for (int i=0; i < data.Count; i++)
            {
                Console.WriteLine("\t// {0} //", data[i]);
            }
            Console.WriteLine();

            // Printing Values Using For Each Loop
            Console.WriteLine("\nPrinting The Items of the List, Using For Each Loop : ");
            foreach (var d in data)
```

```

        {
            Console.WriteLine("\t// {0} //",d);
        }
        Console.WriteLine();

        // Printing Values Using Lambda Expressions
        Console.WriteLine("\nPrinting The Items of the List, Using Lambda Expression :
");
        data.ForEach(d => Console.WriteLine("\t// {0} //",d));

        Console.WriteLine();

        Console.ReadLine();
    }
}
}

```

Output

```

C:\WINDOWS\system32\cmd.exe

Printing The Items of the List, Using For Loop :
    // Manoj //
    // Sai //
    // Vihar //
    // Pavan //
    // Sharath //

Printing The Items of the List, Using For Each Loop :
    // Manoj //
    // Sai //
    // Vihar //
    // Pavan //
    // Sharath //

Printing The Items of the List, Using Lambda Expression :
    // Manoj //
    // Sai //
    // Vihar //
    // Pavan //
    // Sharath //

Press any key to continue . . .

```

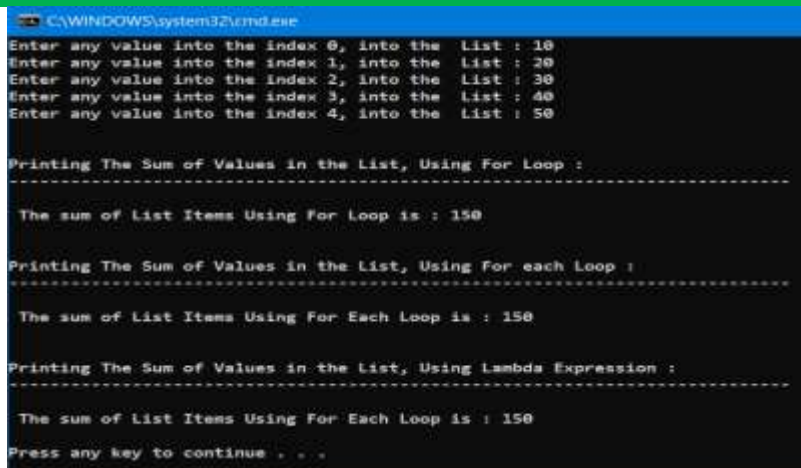
Assignment 8

Write a C# Code, declare a List<string>, read 5 values from user and find sum using for loop, foreach & Lambda

Code

```
using System;
using System.Collections.Generic;
// Author : Manoj.Karnatapu
//Purpose : Declare a List<string> with 5 values from the User & print the sum of it using
for_loop, foreach & Lambda Expression
namespace Day6Project4
{
    internal class Program
    {
        static void Main(string[] args)
        {
            List<int> data = new List<int>();
            int temp;
            int sum1 = 0, sum2 = 0, sum3 = 0;
            // Assigning Values to List
            for (int i = 0; i <= 4; i++)
            {
                Console.WriteLine("Enter any value into the index {0}, into the List : ", i);
                temp = Convert.ToInt32(Console.ReadLine());
                data.Add(temp);
            }
            // Printing Values Using For Loop
            Console.WriteLine("\nPrinting The Sum of Values in the List, Using For Loop:");
            for (int i = 0; i < data.Count; i++)
            {
                sum1 = sum1 + data[i];
            }
            Console.WriteLine("\nThe sum of List Items Using For Loop is: {0}", sum1);
            // Printing Values Using For Each Loop
            Console.WriteLine("Printing The Sum of Values in the List, Using Foreach Loop:");
            foreach (var d in data)
            {
                sum2 += d;
            }
            Console.WriteLine("\nThe sum of List Items Using For Each Loop is: {0}", sum2);
            // Printing Values Using Lambda Expressions
            Console.WriteLine("Printing The Sum of Values in the List, Using Lambda : ");
            data.ForEach(d => sum3 = sum3 + d);
            Console.WriteLine("\nThe sum of List Items Using For Each Loop is: {0}", sum3);
            Console.ReadLine();
        }
    }
}
```

Output



```
C:\WINDOWS\system32\cmd.exe
Enter any value into the index 0, into the List : 10
Enter any value into the index 1, into the List : 20
Enter any value into the index 2, into the List : 30
Enter any value into the index 3, into the List : 40
Enter any value into the index 4, into the List : 50

Printing The Sum of Values in the List, Using For Loop :
-----
The sum of List Items Using For Loop is : 150

Printing The Sum of Values in the List, Using For each Loop :
-----
The sum of List Items Using For Each Loop is : 150

Printing The Sum of Values in the List, Using Lambda Expression :
-----
The sum of List Items Using For Each Loop is : 150
Press any key to continue . . .
```

Assignment 9

Write all data Types in C# and write the respective alias names ?

Answer

A:

Data Types	Alias Names	Class Name
byte	Byte	System.Byte
ushort	UInt16	System.UInt16
uint	UInt32	System.UInt32
ulong	UInt64	System.UInt64
sbyte	SByte	System.Sbyte
short	Int16	System.Int16
int	Int32	System.Int32
long	Int64	System.Int64
float	Single	System.Single
double	Double	System.Double
decimal	Decimal	System.Decimal
bool	Boolean	System.Boolean
char	Char	System.Char
string	String	System.String

Assignment 10

Write example programs for Implicit and Explicit type Casting?

Answer

```
A: using System;
namespace Day6Project5
{
    internal class Program
    {
        static void Main(string[] args)
        {
            // implicit conversion
            // Type Casting short to int
            short myShort = 10;
            int myInt = myShort;

            Console.WriteLine("Implicit Conversion of short to int is: {0}", myInt);

            // Explicit Conversion
            // Casting double to int

            double myDouble = 13.37;
            int myNewInt = (int)myDouble;
            Console.WriteLine("Explicit Conversion of double to int is:{0}", myNewInt);
            Console.ReadLine();
        }
    }
}
```