# DAY 6 MORNING ASSIGNMENT

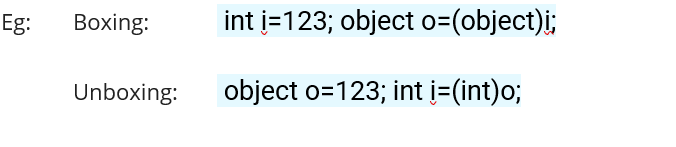
# (BY G V S S SRI LASYA)

1. Research and find how the values of ArrayList are stored in the memory.

* **ArrayList is just a Heterogeneous and Self Re-Dimensioning Array.**
* Elements of various types can be added to the ArrayList. Its size can be changed dynamically.
* The ArrayList uses a “**System. Array”** of **type “Object”**. By default, the **size of this array is 16**, although it can be defined in a constructor or by setting the capacity property. An object array can hold elements of any type.
* Elements can be added to ArrayList using the **Add() Method**. The Add() method first compares the number of elements in array with its capacity. If adding the new element causes the count to exceed the capacity, the array is redimensioned and the **capacity is automatically doubled**.
* The size is not increased one element at a time because this will become very costly since redimensioning is required each time
* Also, it is not increased exponentially because that will waste memory if no value is stored.
* So, it is decided to just double the size of the array when it becomes exhausted.
* This is the precise approach that ArrayList takes and its all done automatically for us.

2. What are the dis-advantages of ArrayList (Collections ArrayList).

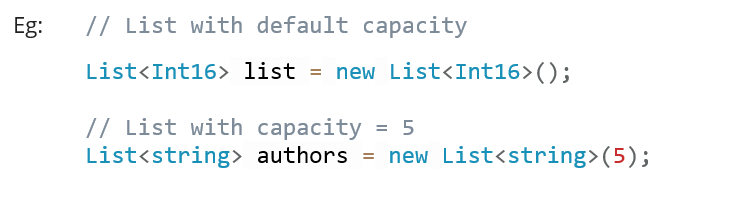
* ArrayList doesn’t support Type Safety. Arraylist can contain any data type of data, we don't declare the data type at the time of an ArrayList
* As the ArrayList's internal array is of object type, every value type is boxed and stored on the heap and each ArrayList element is a reference to a boxed value type. When you access a value type element it is unboxed before you can use it. The boxing and unboxing, can hamper the performance of your application when using large ArrayLists with many reads and writes.



* Any mismatch of type while undergoing say arithmetic operations will lead to run time errors which are much more difficult to resolve
* Eg: trying to add an originally string data typed value stored in ArrayList to an integer.

3. Research and find how the values of List<T> are stored in the memory.

* C# List class represents a collection of strongly typed objects that can be accessed by index.
* IT lets us create a list of objects, find list items, sort list, search list, and manipulate list items. In List<T>, T is the type of objects.
* It is a generic class, defined in the System.Collections.Generic namespace
* List<T> class constructor is used to create a List object of type T. It can either be empty or take an Integer value as an argument that defines the initial size of the list, also known as capacity.
* If there is no integer passed in the constructor, the size of the list is dynamic and grows every time an item is added to the array. You can also pass an initial collection of elements when initialize an object.



**4. In a tabular format write the differences between Collections and generics.**

|  |  |  |
| --- | --- | --- |
|  | **Collections** | **Generic** |
| **namespace** | **System.Collections** | **System.Collections.Generic** |
| **Need for type casting** | **Need typecasting as we don’t specify data type during initialisation and by default all the values are stored as Object type** | **No need of typecasting as data type of each value is specified during initialisation** |
| **Examples** | **1)ArrayList**  **2)Stack**  **3)Queue** | **1)List<T>**  **2)Stack<T>**  **3)Queue<T>** |

5. In a tabular format write all data types in C# and write the respective alias name

|  |  |
| --- | --- |
| Data type | Alias name |
| byte | Byte |
| ushort | UInt16 |
| uint | UInt32 |
| ulong | UInt64 |
| sbyte | sByte |
| short | Int16 |
| int | Int32 |
| long | Int64 |
| float | Single |
| double | Double |
| decimal | Decimal |
| bool | Boolean |
| char | Char |
| string | String |

|  |
| --- |
| WACP to declare List<String> and add 5 values and print the values using for loop ,foreach ,Lambda Expression |
| Code |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day6Project1  {  internal class Program  {  static void Main(string[] args)  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Author : G V S S SRI LASYA  Purpose : declare a List<string> with 5 user inputs and print  using for loop,foreach and Lambda expression  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  List<string> colors = new List<string>(5);  //taking user inputs using for loop  for(int i = 1; i <= 5; i++)  {  Console.Write("\nEnter color{0} : ",i);  colors.Add(Console.ReadLine());  }  //printing values using for loop  Console.Write("\n\n\nColors(using for loop) :");  for (int j = 0; j < colors.Count; j++ )  Console.Write("\t{0}",colors[j]);  //printing values using foreach  Console.Write("\n\n\nColors(using foreach) :");  foreach(var color in colors)  Console.Write("\t{0}",color);  //printing values using Lambda expression  Console.Write("\n\n\nColors(using Lambda expression) :");  colors.ForEach(color => Console.Write("\t" + color));  Console.ReadLine();  }  }  } |
| Output |
|  |

|  |
| --- |
| WACP to declare List<int> and add 5 values and print the sum of values using for loop ,foreach ,Lambda Expression |
| Code |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day6Project2  {  internal class Program  {  static void Main(string[] args)  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Author: G V S S SRI LASYA  Purpose: Declare List<int> and read 5 values from user  and find sum using for loop,foreach and Lambda expression  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  List<int> numbers = new List<int>(5);  int intermediate,sum1 = 0,sum2 = 0,sum3 = 0;  //taking number inputs from user  for(int i = 0 ; i < 5 ; i++)  {  Console.Write("\nEnter number{0} : ",i+1);  intermediate = Convert.ToInt32(Console.ReadLine());  numbers.Add(intermediate);  }  //printing sum using for loop  for(int i = 0; i < numbers.Count ; i++)  sum1 += numbers[i];  Console.Write("\n\nSum of numbers(using for loop) :{0}", sum1);  //printing sum using foreach  foreach (var number in numbers)  sum2 += number;  Console.Write("\n\nSum of numbers(using foreach) :{0}", sum2);  //printing sum using Lambda expression  numbers.ForEach(number => sum3 += number);  Console.Write("\n\nSum of numbers(using Lambda expression) :{0}", sum3);  Console.ReadLine();  }  }  } |
| Output |
|  |

|  |
| --- |
| Declare ArrayList and assign some values and find sum. |
| Code |
| using System;  using System.Collections;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day6Project3  {  internal class Program  {  static void Main(string[] args)  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Author : G V S S SRI LASYA  Purpose : declare ArrayList and assign some values  and find sum.  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  int sum = 0;  ArrayList numbers = new ArrayList();  //assigning values  numbers.Add(10);  numbers.Add(20);  numbers.Add(30);  numbers.Add(40);  numbers.Add(50);  //calculating sum using and printing it  foreach (var number in numbers)  sum += (int)number;  Console.Write("\nSum of ArrayList values is : {0}",sum);  Console.ReadLine();  }  }  } |
| Output |
|  |

|  |
| --- |
| Declare List<int> and assign some values and find sum |
| Code |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day6Project4  {  internal class Program  {  static void Main(string[] args)  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Author : G V S S SRI LASYA  Purpose : declare List<int> and assign some values  and find sum  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  int sum = 0;  List<int> numbers = new List<int>(5);  //assigning values  numbers.Add(10);  numbers.Add(20);  numbers.Add(30);  numbers.Add(40);  numbers.Add(50);  //calculating sum and printing it  foreach (int number in numbers)  sum += number;  Console.WriteLine("\nSum of values of the list is : {0}",sum);  Console.ReadLine();  }  }  } |
| Output |
|  |

|  |
| --- |
| Implicit and explicit type casting. |
| Code |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day6Project5  {  internal class Program  {  static void Main(string[] args)  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Author : G V S S SRI LASYA  Purpose : Implicit and explicit type casting.  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/    short value1 = 10;  int value2 = 40;  int value3 = 13;  double value4 = 100d;  //implicit type conversion  Console.Write("\nValue before implicit type conversion : {0}", value2);  value2 = value1;  Console.Write("\nValue after implicit type conversion : {0}", value2);  //explicit type conversion  Console.Write("\n\n\nValue before explicit type conversion : {0}", value3);  value3 = (int)value4;  Console.Write("\nValue after explicit type conversion : {0}", value3);  Console.ReadLine();  }  }  } |
| Output |
|  |