Day -6 Morning Assignment

By

Vamsi Krishna Mandapati

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

A) The elements of an ArrayList are stored in **a chunk of contiguous memory**. When that memory becomes full, a larger chunk of contiguous memory has to be allocated (usually twice the size) and the existing elements are copied into this new chunk. We call this chunk the capacity of the ArrayList object.

ArrayList internally uses an array to store the elements. Just like arrays, It allows you to retrieve the elements by their index.

ArrayList changes memory allocation as it grows.
When we specify the capacity while initializing the ArrayList, it allocates enough memory to **store objects up to that capacity**. The logical size remains 0. When it is time to expand the capacity, a new, larger array is created, and the values are copied to it

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

A) 1. EveryTime Boxing and Unboxing to get the value is a disadvantage in arraylist

2. if there is chance of assigning wrong dadaype you will get RunTime errors and Runtime errors are more dangerous than Compile Errors is a disadvantage in ArrayList.

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

1. namespace
2. Each element is of what type
3. do you need type casting here
4. Example - ArrayList, List<T>

A)

	Collections	Generics
NameSpace	System.Collections	System.Collections.Generic
Type	Object	Int
Type Casting	Yes	No
Syntax	`ArrayList data = new ArrayList();`	`List<T> data = new List<T>();`

Example	`using System;`
	`using System.Collections;`
	`using System.Collections.Generic;`
	`using System.Linq;`
	`using System.Text;`
	`using System.Threading.Tasks;`
	
	`namespace D6P1ArrayListSum`
	`{`
	`internal class Program`
	`{`
	`static void Main(string[] args)`
	`{`
	
	`ArrayList data = new ArrayList();`
	
	`int sum = 0;`
	`data.Add(5);`
	`data.Add(10);`
	`data.Add(15);`
	`data.Add(25);`
	
	`foreach (var d in data)`
	`{`
	`sum = sum + (int)d;`
	`}`
		`using System;`
		`using System.Collections.Generic;`
		`using System.Linq;`
		`using System.Text;`
		`using System.Threading.Tasks;`
		
		`namespace D6P2GerericsSum`
		`{`
		`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(15);`
		`data.Add(25);`
		
		`foreach (var d in data)`
		`{`
		`sum = sum + d;`
		`}`

	Console.WriteLine(sum); Console.ReadLine(); } } }	Console.WriteLine(sum); Console.ReadLine(); } } }

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

A) In generics, such as List<T> , they're still stored on the heap. The difference is that, internally, **a List<int>** makes a single array of integers, and can store the numbers directly. WIth ArrayList, you end up storing an array of references to boxed integer values.

9). 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
char	Char
string	String
boolean	Boolean

10. Write example programs for implicit and explicit type casting.

A) **Implict TypeCasting :**

Converting or TypeCasting smaller DataType to Larger DateType is known as Implicit Type Casting.

Ex:-

```
using Sysyem;
class Program
public static void Main(string[] args)
{
 short p = 5;
 int q = p;
 Console.WriteLine(q);
}
```

Explicit TypeCasting :

Converting or TypeCasting Larger DataType to Smalller DateType is known as Explicit Type Casting.

Ex:-

```
using Sysyem;
class Program
public static void Main(string[] args)
{
 int p = 10;
 short q = (short)p;
 Console.WriteLine(q);
}
```

==========END==========