**Lesson 02- Collections and Data Structure in C#**

Objective:

In this lesson, we will learn the usages of Array, ArrayList and List in C#. At the end of this lesson we will discuss Stack, Queue, Hashtable and Dictionary data structures.

Prerequisite:

Participants must have familiarization with basic C# syntax.

Topics to be discussed:

* Array
* ArrayList
* List
* Differences between Array, ArrayList and List
* Stack
* Queue
* Hashtable
* Dictionary

Pretest& Posttest questions:

1. What is array?
2. What is the benefit of foreach loop instead of for loop
3. What are the differences between Array, ArrayList and List?
4. What is the disadvantage of hashtable over dictionary?

* **Understanding Array in C#**

**Array in C#:**

We use an array to store similar type of data together. An *array* is a data structure that contains a number of variables called the *elements* of the array. An array is fixed in size. In C# arrays can be created and initialized in different ways:

int[] numbers = new int[5];

numbers[0] = 15;

numbers[1] = 6;

numbers[2] = 74;

numbers[3] = 8;

numbers[4] = 29;

First line,int[] numbers = new int[5] is for declaring an array which can keep five integer numbers sequencially. Then next five lines is used to initialize five integers into the five location of numbers arrays using index value. Suppose you need to get the third number of this array. You need to write the following line:

int myNumber = numbers[2];

The value of myNumber will be 74.

If you go out-of-index for initializing or for getting data from an array, you will get an exception, IndexOutOfRangeException in runtime.

numbers[5] = 8736;

int myNumber = numbers[8];

Both lines throw the IndexOutOfRangeException.

Array can be declare and initialize in a single line

int[] numbers = new int[5]{15,6,37,84,9};

or

int[] numbers = {5,28,72,8,36};

Following line is for declaring and initializing an string array, animals:

string[] animals = { "Elephant", "Cat", "Mouse" };

**Walkthrough: Adding Five Numbers using array**

1. Select conlose application from template and update the Main() method with following code:

static void Main(string[] args)

{

double[] numbers = new double[5];

double sum = 0;

Console.WriteLine("This program adds five numbers");

for (int index = 0; index < numbers.Length ; index++)

{

Console.WriteLine("Enter number " +

(index + 1).ToString());

numbers[index] = Convert.ToDouble(Console.ReadLine());

sum += numbers[index];

}

Console.WriteLine("Sum of five numbers is: " + sum);

Console.ReadKey();

}

1. Run the application.

Here, I use Length property of numbers array to get it’s length. All the remaining codes are self-describtive so you can easily understand it.

In .Net class library, there’s a class Array which is used to do some operations for an array of any kind of data type. Sort() method of Array is used to sort an array:

int[] numbers = new int[5] {31, 6, 4, 3, 7};

Array.Sort(numbers);

After running this code snippet you will get the sorted array.

IndexOf() method used to return index of a specific value within array:

int desiredValue = Array.IndexOf(numbers, 100);

**Walkthrough: Sorting an array.**

We will create an array of integer in which we will keep some integer values and by using Sort method of Array class we will sort the array and print the elements of the array after sorting.

**Steps:**

1. Create a console application .
2. We will decalre and intialize an array in following way:

int[] numbers = new int[5] { 3, 6, 4, 3,100 };

1. We will sort an array using sort method and show output as below:

Array.Sort(numbers);

for (int i = 0; i < numbers.Length; i++)

{

Console.WriteLine(numbers[i]);

}

**Foreach loop:**

You are already familiar with for, while, do-while for repeating statements in your code.Here, we will discuss another interesting looping for iteration and it is ***Foreach***. Foreach is used to iterate through an arrays or collection. We can define foreach as below:

int[] numbers = { 5, 28, 72, 8, 36 };

foreach (int myNumber in numbers)

{

Console.WriteLine(myNumber);

}

In this code, we initialize an integer array and iterate through it using *foreach.*Remember, you can’t initialize (directly) any value in any index position of an array using foreach. Even you can’t change iteration variable inside *foreach* loop.

foreach (int myNumber in numbers)

{

myNumber = Convert.ToInt32(Console.ReadLine());

}

This code will show a compile error as we are trying to assign a value to iteration variable, myNumber.

**Walkthrough: We will create an array of string and we will show the contents of string array in output using foreach.**

Steps:

1. Create a console application .
2. We will declare and intialize an array of string in following way:

string [] personNames = { "Rahim", "Shoaib", "Kamrul" };

1. We can see these names of personNames array by following way using forecah loop

foreach (string name in personNames)

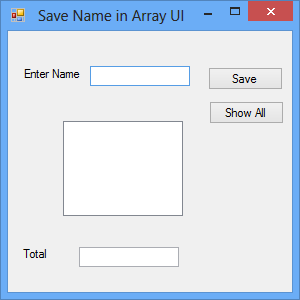
{

Console.WriteLine(name);

}

**Limitation of Array:**Array in C# doesn’t grow dynamically. ☹

**Practice: Save persons’ name in array:**



User will enter name and press save button. When show all will be clicked, all name will be displayed in listbox and total number of name will be shown in textbox below the form. Note that only five names will be saved.

Download the solution from attachment. Practice : Save persons name in array.

* **ArrayList & List:**

Arraylist removes the fixed-size limitation of array.

ArrayList myList = new ArrayList();

myList.Add(100);

myList.Add(300);

foreach (int myNumber in myList)

{

Console.WriteLine(myNumber);

}

In the above lines we have declared an ArrayList, myList and add two integer numbers using it’s Add() method. After that, I iterate through the ArrayList using foreach.

**Practice: Save persons’ name in ArrayList:**

Do the same practice, what you have done for Array.But in this practice you will use ArrayList instead of array. Download the solution from attachment and see the code. In the top of NameUI form you will find the following code:

ArrayList names = new ArrayList();

And it will grow as many names (item) as you want. That means ArrayList grows as per need.

But still ArrayList suffers for generics facility. i.e. you can’t customize the data type for it You can add any kind of data type here. This is a big disadvantage of ArrayList in object-oriented programming.

Instead of using ArrayList we can use List. In List you can define the data type which will be used in the list later. We can define a List following way:

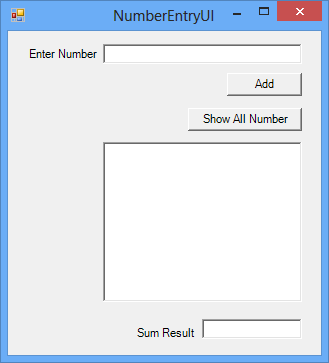
List<int> numbers = new List<int>();

In List we can define the data type which List will hold.As List is also an object as array it has different merhods and properties to working with elements of the List.We can add a element in a List using it Add method as following way:

numbers.Add(4);

We can use Remove method to remove a specific item from List. Sort method to Sort elements fo the List. Max,Min to return maximum or minimum item of the array.Revrse to reverse the array.Count property to count number of Items in the array.

**Walkthrough:Create a List which can take doublenumber and show the contents of List and their sumresult like following UI:**



1. Design the UI as above.
2. Declare a List<double> inside form as follows:

List<double> numbers = new List<double>();

1. Now write the following code inside *add button* code:

double number = Convert.ToDouble(numberTextBox.Text);

numbers.Add(number);

1. And finally add following code inside *show all button* code:

double total = 0;

foreach (double aNumber in numbers)

{

numberListBox.Items.Add(aNumber);

total += aNumber;

}

totalTextBox.Text = total.ToString();

1. Run your project and see how it works. It is suggested to debug your code to understand it clearly.

*Don’t know how to debug? No worries. See how easy it is:* [*Mastering Debugging in Visual Studio*](http://www.codeproject.com/Articles/79508/Mastering-Debugging-in-Visual-Studio-A-Beginn)

In the later chapters, we will use List most instead of ArrayuList or even array, since it grows as needed and has the generics support.☺

* **Stack and Queue**

Queue works on First in first Out (FIFO) basis. And stack works on Last in First Out (LIFO) basis.

**Working with Queue.** It has two public methods:

|  |  |
| --- | --- |
| Enqueue-Queue | Adds an object to the end of the Queue. |
| Dequeue-Queue | Removes and returns the object at the beginning of the Queue. |

**Walkthrough: Keep some integer number to a queue.**

Create a new Console application and write the following code.

Class Program

{

Static void Main(string[] args)

{

Queue<int> aQueue = new Queue<int>();

aQueue.Enqueue(100);

aQueue.Enqueue(200);

aQueue.Enqueue(150);

Console.WriteLine(aQueue.Dequeue());

Console.WriteLine(aQueue.Dequeue());

Console.WriteLine(aQueue.Dequeue());

Console.ReadKey();

}

}

You can use foreach to iterate through a Queue. So update your code as follows:

Static void Main(string[] args)

{

Queue<int> aQueue = new Queue<int>();

aQueue.Enqueue(100);

aQueue.Enqueue(200);

aQueue.Enqueue(150);

foreach (int anItem in aQueue)

{

Console.WriteLine(anItem);

}

Console.ReadKey();

}

**Working with Stack:**

Stack has few public methods as described below:

|  |  |
| --- | --- |
| Count | Gets the number of objects contained in a stack. |
| Push | Inserts an object at the top of the stack. |
| Pop | Removes and returns the object at the top of the stack. |

**Practice: Keep some person’s name to a stack and show them using foreach loop.**

Find the solution in attachment and see the code.

* **Hashtable and Dictionary**

**Working with Hashtable:**

The Hashtable class represents a collection of **key-and-value pairs** that are organized based on the hash code of the key. It uses the key to access the elements in the collection.A hash table is used when you need to access elements by using **key**, and you can identify a useful key value. Each item in the hash table has a key/value pair. The key is used to access the items in the collection.See the following code:

Static void Main(string[] args)

{

Hashtable aHashtable = new Hashtable();

aHashtable.Add("111", "ASP.Net MVC");

aHashtable.Add("222", "Java Basics");

aHashtable.Add("101", "Laravel");

aHashtable.Add("101", "Zend");

aHashtable.Add("987", "SEO");

string courseName = (string) aHashtable["101"];

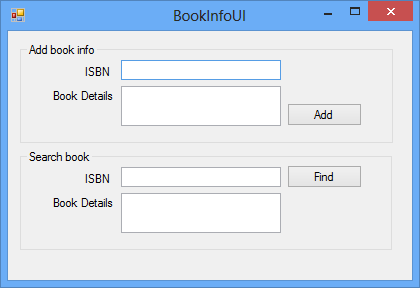
Console.WriteLine(courseName);

Console.ReadKey();

}

Here, we create an object of Hashtable and add some data (course id and name) using key value pair mechanism.

**Practice: Make a desktop application as follows where you will use Hashtable for string book information.**



Download the solution from attachment and see how code works.

**Limitation of Hashtable:** Do you remember why List is good over ArrayList? Yes, List has the generics support.Now, think about Hashtable, it has no generics support ☹. Let’s work with Dictionary.

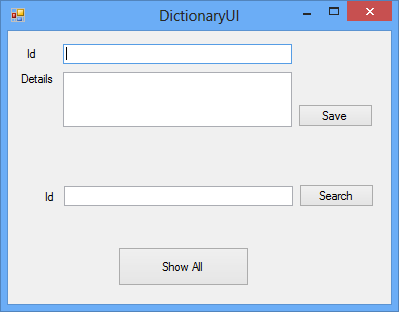
**Working with Dictionary:**

Dictionary is same as Hashtable, i.e. it a key-value style data structure.The only difference is that dictionary has generics support.

See how a dictionary is created:

Dictionary<string, string> studentInfo = new Dictionary<string, string>();

**Practice : Let’s make an application as folloing UI by which user will keep citizen information. Note that here you wil dictionary instead of Hashtable**



Download the project and see how it works.