WAP TO DEMONSTRATE LAMBDA EXPRESSION IN C#:

Introduction:

A lambda expression is a convenient way of defining an anonymous (unnamed) function that can be passed around as a variable or as a parameters to a method call. many LINQ method takes a function (called a delegate) as a parameters. lambda expression is expressed as:

source code:

```
using system;
namespace LambdaExp
{
  class program
  {
    delegate int del(int i);
    static void Main(string[] args)
    {
      del d = y => y*y;
      intj = d(5);
      console.WriteLine(j);
    }
  }
}
```

program output:

```
25
```

WAP TO DEMONSTRATE LINK QUERY EXPRESSION IN C#:

Introduction:

LINQ is the name for a set of technologies based on the Integration of query capabilities directly into the C# longuage, with LINQ, a query is a fixed class longuage construct, just like classes, method, events.

When we write LINQ queries in C# for SQL server database, XML documents, ADO.NET dataset and any collection of object that supports IEnumerable or the generic IEnumerable <T> interface. LINQ support is also provided by third parties for many web service and other database implementation.

source code:

```
using system;
namespace LinkqueryExpression
{
  Internal class program
  {
    static void Main(string [] args)
    {
      int[] scores = new int [] { 97, 92, 81, 69};
      IEnumerable<int> Scoresquery = from score in
                                     scores where score >80
                                     select score;

      Foreach (int i in scoresquery)
      {
        concole.writeLine (i+ " ");
      }
```

```
        }
    }
}
```

program output:

```
97, 92, 81
```

WAP TO DEMONSTRATE TRY STATEMENT AND EXCEPTION IN C#:

Introduction:

Exception handling in c#, supported by the try, catch and finally block is a mechanism to defect and handle run-time errors in code. c# provides three keywords try, catch and finally to implement exception handling.

source code:

```
using system;
namespace ExceptionHandling
{
  internal class program
  {
    static void Main(string[] args)
    {
      try
      {
        int a = 12;
        int b = 0;
        int c = a/b;
        console.WriteLine(c);
      }
      catch(DivideByZeroException de)
      {
        console.WriteLine("Divide By zero");

      }
      catch(Exception)
      {
        console.WriteLine("Exception");
      }
```

```
        Finally
        {  console.WriteLine("The error" + "Hey");
        }
        console.WriteLine("Rest of the code");
     }
  }
}
```

program output:

```
Divide By zero
The error Hey
Rest of the code.
```