C# Notes

- **Escape Sequences in C#**
  http://msdn.microsoft.com/en-us/library/h21280bw.aspx

  Verbatim Literal is a string with an @ symbol prefix, as in @“Hello".

  Verbatim literals make escape sequences translate as normal printable characters to enhance readability.

  Practical Example:
  Without Verbatim Literal : “C:\\Pragim\\DotNet\\Training\\Csharp” – Less Readable
  With Verbatim Literal : @“C:\Pragim\DotNet\Training\Csharp” – Better Readable

- **In C# types are divided into 2 broad categories.**
  Value Types - int, float, double, structs, enums etc
  Reference Types – Interface, Class, delegates, arrays etc

  By default value types are non nullable. To make them nullable use ?
  int i = 0 (i is non nullable, so "i" cannot be set to null, i = null will generate compiler error)
  int? j = 0 (j is nullable int, so j=null is legal)

  Nullable types bridge the differences between C# types and Database types

  Program without using NULL coalescing operator
  ```
  using System;
  class Program
  {
      static void Main()
      {
          int AvailableTickets;
          int? TicketsOnSale = null;

          if (TicketsOnSale == null)
          {
              AvailableTickets = 0;
          }
          else
          {
              AvailableTickets = (int)TicketsOnSale;
          }

          Console.WriteLine("Available Tickets={0}", AvailableTickets);
      }
  }
  ```

  The above program is re-written using NULL coalescing operator

```
using System;
class Program
{
   static void Main()
   {
      int AvailableTickets;
      int? TicketsOnSale = null;

      //Using null coalesce operator ??
      AvailableTickets = TicketsOnSale ?? 0;

      Console.WriteLine("Available Tickets={0}", AvailableTickets);
   }
}
```