Facultad: Ingeniería
Escuela: Computación
Asignatura: Programación II

GUIA 1:
Entorno de Visual Studio. Lenguajes de Programación.

Objetivos

- Conocer los lenguajes de programación contenidos en el paquete de Visual Studio
- Evaluar qué lenguaje es apropiado de acuerdo al software a desarrollar

Introducción

Esta guía de laboratorio, ofrece una breve introducción al IDE (Entorno de Desarrollo Integrado) de Visual Studio 2012, utilizado para sistemas operativos Windows. Este soporta múltiples lenguajes de programación tales como C++, C#, Visual Basic .NET, F#, Javascript, al igual que entornos de desarrollo web como ASP.NET MVC (Modelo Vista Controlador) además de capacidades online bajo Windows Azure.

Visual Studio permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET. Así pueden crearse aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, móviles, consolas, entre otros. .NET Frameworks que soporta 2.0, 3.0, 3.5, 4.0, 4.5

Adicionalmente, Microsoft ha puesto a disposición una versión reducida de MS SQL Server llamada SQL Server Express Edition cuyas únicas limitaciones son que no soporta bases de datos superiores a 4GB de tamaño, se ejecutan en un procesador, y no cuentan con el agente de SQL Server.

En el caso de las prácticas se utilizará C#. Como podrá comprobar, la sintaxis del lenguaje (C#) en sí mismo es muy fácil de aprender. Sin embargo lo que requiere más tiempo es aprender a programar en .NET Framework (marco de trabajo .NET de Microsoft) usando el lenguaje C#.

C# irrumpe en el mercado como un lenguaje bien diseñado y con muchas virtudes en una industria plagada de soluciones y herramientas de programación para todos los gustos. ¿Cuáles son entonces, los motivos por los cuales deberíamos optar por C#?

- C# es un lenguaje moderno y orientado a objetos, con una sintaxis muy similar a la de **C++** y **Java**. Combina la alta productividad de **Visual Basic** con el poder y la flexibilidad de **Java**.

- La misma aplicación que se ejecuta bajo **Windows** podría funcionar en un dispositivo móvil de tipo **PDA**.
- Se puede crear una gran variedad de aplicaciones en C#. aplicaciones de consola, aplicaciones para Windows con ventanas y controles, aplicaciones para la Web, etc.
- C# gestiona automáticamente la memoria, y de este modo evita los problemas de programación tan típicos en lenguajes como C o C++.
- Mediante la plataforma .NET desde la cual se ejecuta es posible interactuar con otros componentes realizados en otros lenguajes .NET de manera muy sencilla.
- También es posible interactuar con componentes no gestionados fuera de la plataforma .NET. Por ello, puede ser integrado con facilidad en sistemas ya creados.
- Desde C# podremos acceder a una librería de clases muy completa y muy bien diseñada, que nos permitirá disminuir en gran medida los tiempos de desarrollo.

En C# y .NET todo, absolutamente todo es una clase que, en última instancia deriva de la clase base ***object***, esto responde a una decisión de diseño muy importante ya que, entre otras cosas, permite que se considere cualquier elemento (como por ejemplo un entero) como un objeto.

Es importante distinguir el programa en el cual se está trabajando por medio de su extensión, Visual Basic ocupa la extensión .vb, C# se maneja con la extensión .cs y C++ emplea .cpp

Materiales y Equipo

Nº	*Cantidad*	*Descripción*
1	1	Guía de Laboratorio #1 de Programación II
2	1	Computadora con programa: ➢ Microsoft Visual C#
3	1	Dispositivo de memoria externo

Procedimiento

G1_Ejemplo_01:

Nuestra primera aplicación con Visual Studio 2012 / 2013

Para crear nuestra primera aplicación debemos tomar en cuenta que C# pertenece a la suite de Visual Studio, es por ello que comparte ciertas características de entorno con otros lenguajes de la misma suite; esta ventaja permite que sea sumamente configurable y que la organización de los paneles, así como la pantalla de inicio, podrá variar en función de cómo la configuremos.

Teniendo el entorno ya abierto, lo primero que debemos de hacer será “Crear un nuevo proyecto”. Para ello tiene que seguir los siguientes pasos:

1. Abra **Visual Studio 2012/2013**, que se encuentra en el menú ***Inicio*, *Todos los programas, Microsoft Visual Studio 2012 / 2013***.

2. Deberá cargársele una página de inicio (como la de la figura 1), si ese es el caso seleccione la opción **New Project (Nuevo Proyecto)**, como muestra la imagen.

   Si en su computadora no se carga una página de inicio entonces diríjase al menú **Archivo (File)** y seleccione la opción **Nuevo ->Proyecto (New ->Project)** como muestra la figura 2:

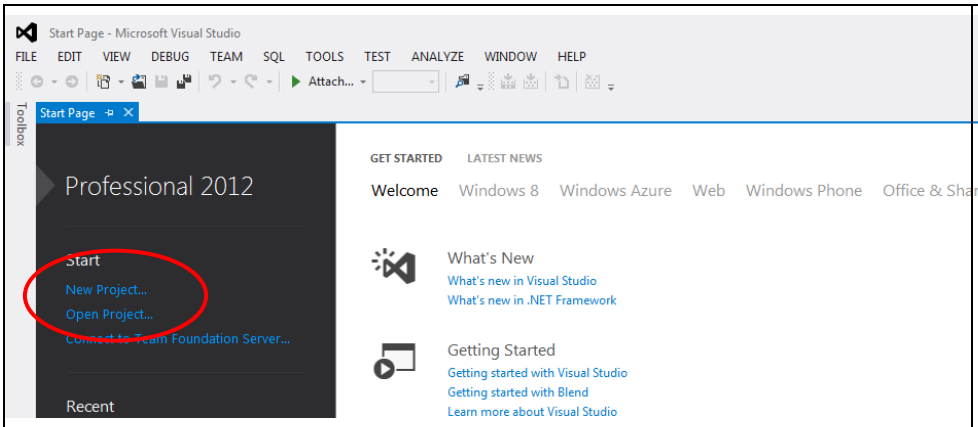


Figura 1. *Creación de nuevo proyecto desde la página de inicio*

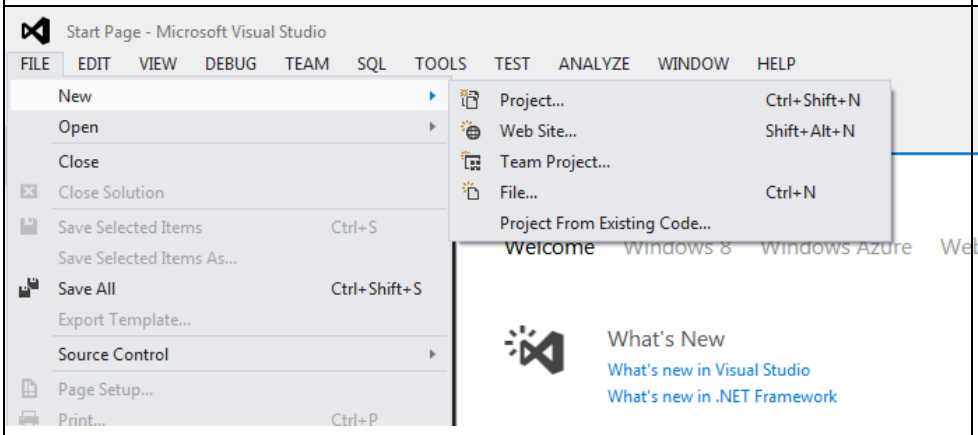


Figura 2. *Creación de nuevo proyecto desde el menú de Visual Studio*

3. Aparecerá una ventana emergente como se muestra en la figura 3, de todas las opciones posibles seleccionamos en la parte izquierda (1) **Visual C#,** al centro **Console Application**(2), colocamos un nombre a nuestro proyecto (3**) (HolaMundo)** y para finalizar damos click en el botón **OK** (4)

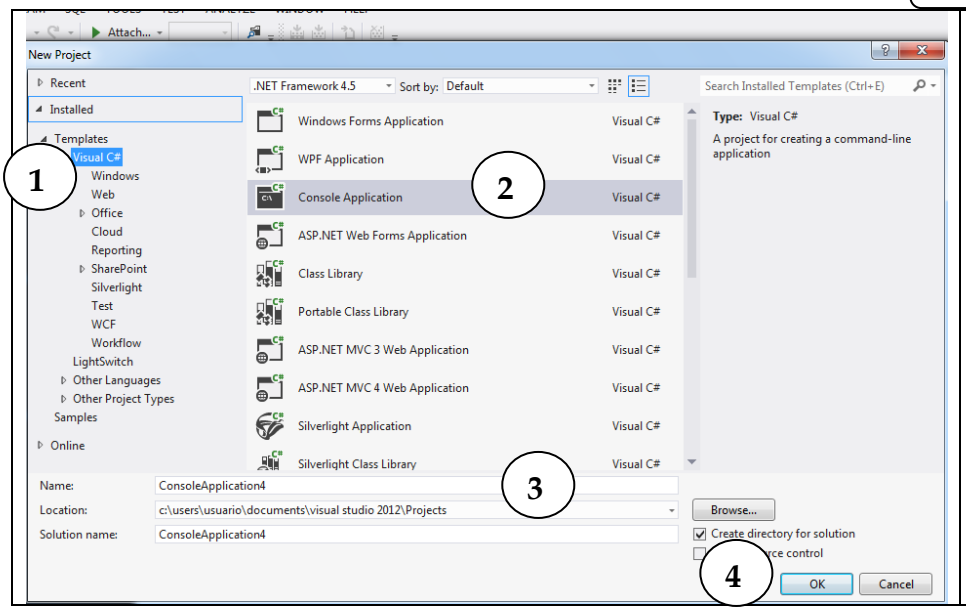


***Figura 3**. Creación de proyecto de C# en modo consola.*

4. Con la ejecución exitosa de los pasos anteriores, usted ha creado un proyecto en Visual C# llamado **HolaMundo** que contiene una clase llamada **Program.cs** Esta clase es el punto de partida de la aplicación para proyectos que se ejecutan en *Modo de Consola*; y es dentro de esta clase donde iremos escribiendo el código de nuestra aplicación.

5. El código preescrito en nuestra clase **Program.cs** es el siguiente:

```
using System;
using System.Collections.Generic;
using System.Text;

namespace HolaMundo
{
   class Program
   {
      static void Main(string[] args)
      {
      }
   }
}
```

6. Analicemos el código. Siempre que comience una aplicación de Consola en C#, Visual Studio .NET añadirá este código. La directiva using System nos permitirá usar miembros de este espacio de nombres sin poner la palabra System delante. Luego hay definido un espacio de nombres (namespace HolaMundo) para la aplicación, que se llama igual que la misma (HolaMundo).

7. Un namespace (espacio de nombres) constituye una forma lógica de organizar y estructura nuestro código de forma que podamos organizar la funcionalidad con un determinado sentido semántico.

   Ejemplos de namespace presentes en el framework son por ejemplo el (namespace System) que contiene la funcionalidad básica del framework, el (namespace

System.Collections) que contiene los tipos de datos de colecciones (ArrayList, Listas Enlazadas) o el (namespace System.Collections.Generic) que contiene los tipos de datos de colecciones que usan el nuevo sistema de genéricos de .NET 4.0 o .NET 4.5.1 (dependiendo del framework presente en su computadora).

La sentencia Using, es de gran utilidad para ahorrarnos trabajo, al tener que anteponer nombres de *namespace* a cada una de las clases que utilicemos. Ejemplo using System;

8. Punto de entrada: En cualquier caso, puesto que todo programa debe empezar por algún punto en todo assembly ejecutable debe existir un punto por el que comienza la ejecución. En .NET a nivel de código intermedio dicha entrada se marca mediante el comando .entrypoint (de hecho la única diferencia entre un dll de .NET y un exe de .NET es la presencia o ausencia de dicha entrada). En C# el inicio de la ejecución se declara mediante un metodo estático llamado Main cuya estructura debe ser

```
static void Main(string[] args)
```

Nuestro programa siempre comenzará por ese punto

9. Ahora escriba las siguientes dos líneas de código dentro del método Main:

```
Console.Write("Recordando como programar en C#");
Console.Read();
```

Console es una clase que no existe en el espacio de nombres global, sino en el espacio de nombres System. Gracias a que le hemos indicado al compilador que estamos utilizando dicho espacio de nombres, él sabrá que las clases que utilicemos también podrán estar ahí.

Write: es un método de la clase Consola, que sirve para imprimir en la pantalla el mensaje que deseemos, en este caso “Recordando como programar en C#”.

10. Por lo tanto el código completo de nuestra aplicación es:

```
using System;
using System.Collections.Generic;
using System.Text;

namespace HolaMundo
{
   class Program
   {
      static void Main(string[] args)
      {
         Console.Write("Recordando como programar con C#"); // Escribe en consola el mensaje
         Console.Read();          // Hace una pausa
      }
   }
}
```

11. Para ejecutar la aplicación, dar clic en el botón verde que está en la barra de herramientas estándar, como se muestra a continuación (figura 4):

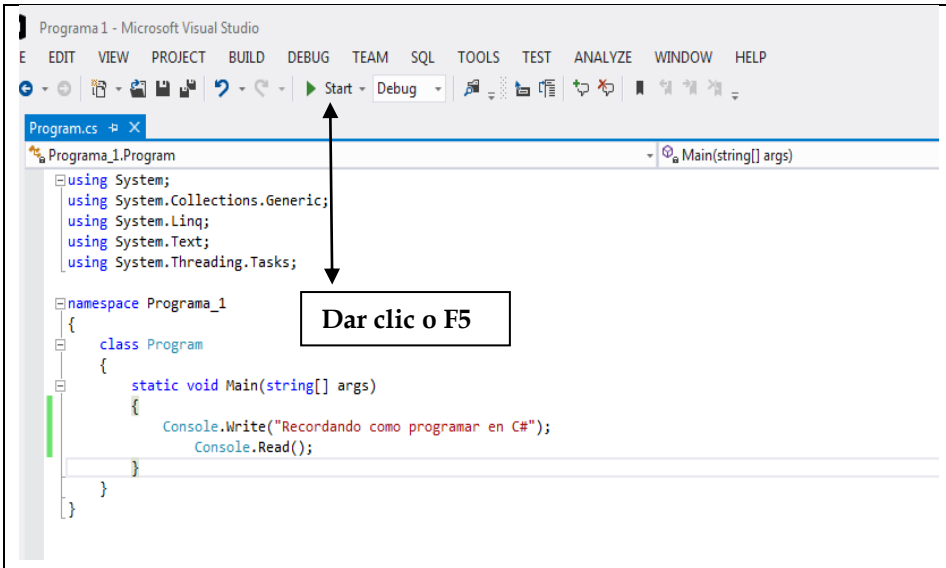
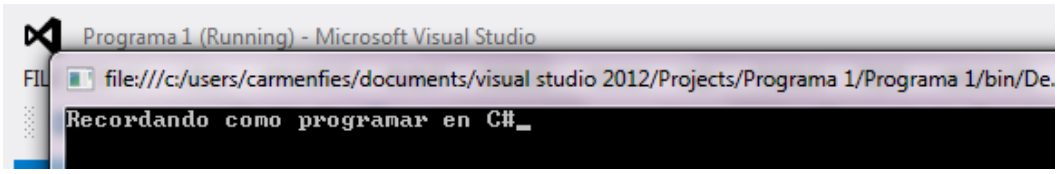


Figura 4. *Ejecución del código escrito*

12. La ejecución exitosa del paso anterior, tuvo que haber lanzado una ventana de consola, mostrando el mensaje **Recordando como programar en C#**, como se muestra en la siguiente figura:



G1_Ejemplo_02:

Lo que haremos en este ejemplo es crear un nuevo proyecto, donde se hará un programa que reciba dos números enteros por parte del usuario y a continuación los sume.

1. Repita los pasos del ejercicio anterior, hasta llegar al nombre de su aplicación, que llamará "**SumaEnteros".** Se cargará el área de codificación

2. Escriba el siguiente código dentro del método Main:

```
int a;
int b;
int suma;

Console.WriteLine("Ingrese el primer numero a sumar");
a = int.Parse(Console.ReadLine());

Console.WriteLine("Ingrese el segundo numero a sumar");
```

```
        b = int.Parse(Console.ReadLine());

        suma = a + b;

        Console.WriteLine("La suma es: " + suma);
        Console.ReadLine();  //Pausa para ver el resultado
```

G1_Ejemplo_03:

Uso del IF, FOR y WHILE

1. Cree un nuevo proyecto y digite el siguiente código dentro del método Main

```
int a;
int b;
int i;

//uso del IF
Console.WriteLine("**********Ejemplo de IF**********");
Console.WriteLine("-Pide dos numeros enteros y los compara");
//ingreso de datos
Console.WriteLine("Ingrese el primer numero entero");
a = int.Parse(Console.ReadLine());
Console.WriteLine("Ingrese el segundo numero entero");
b = int.Parse(Console.ReadLine());
//estructura if
if (a >= b)
{
    Console.WriteLine("A,{0} es mayor que B,{1}", a, b);
}
else
{
    Console.WriteLine("B,{1} es mayor que A,{0}", a, b);
}
Console.Read();
//uso del FOR
Console.WriteLine("**********Ejemplo de FOR**********");
Console.WriteLine("Imprime numeros de 1 hasta 10");
for (i = 1; i <= 11; i++)
    Console.WriteLine(i);

Console.Read();
//uso del WHILE
Console.WriteLine("**********Ejemplo de WHILE**********");
Console.WriteLine("Imprime numeros de 1 hasta 10");
i=1;
while(i <11)
{
    Console.WriteLine(i);
    i++;
}
Console.ReadLine();
Console.WriteLine("**********Ejemplo de WHILE**********");
```

2. ¿Cómo mejoraría el ejercicio anterior? ¿Qué instrucciones agregaría y por qué? Realice las modificaciones propuestas y discútalas con su instructor

Análisis de Resultados

1) Desarrolle un programa que pida 3 numeros al usuario y determine cual es el menor y el mayor.

2) Crear un programa que imprima los números del 1 al 50, excepto el número 25.

3) Hacer un programa que solicite el total N de empleados de la empresa de Ropa “El buen vestir”. Luego solicite el sueldo base de c/empleado para así calcular y mostrar los descuentos de ley (Renta: 10%, AFP Confía: 5% y Seguro Social ISSS: 4%) y su sueldo neto (sueldo base-descuentos). Al finalizar muestre un reporte contable que muestre el total a pagar en conceptos de sueldos netos y los totales retenidos en concepto de Renta, AFP y del seguro social.

4) Escriba un programa que genere la serie de números primos con tantos elementos como diga el usuario. Es decir si el usuario dice 10, el programa deberá generar los números: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29.

Investigación Complementaria

1) Escribir un programa en C# que al ingresar por medio del teclado la cantidad de agua que cae día a día en un mes (en milímetros), determine el día de mayor lluvia, el de menor y el promedio del mes.

2) Crear un programa que al ingresar una frase (no más de 20 caracteres) muestre cuántas vocales tiene.

3) Un estudiante de Programación III está organizando una fiesta en la cual una computadora controla el acceso mediante 5 claves. Si se ingresa al menos una clave incorrecta esta imprimirá “TE EQUIVOCASTE DE FIESTA” y no permitirá el ingreso. Si las 5 claves son correctas imprimirá “BIENVENIDO A LA FIESTA”

   Las claves son:
   i) “Tienes”
   ii) “ que ser”
   iii) “invitado”
   iv) “para”
   v) “ingresar”

Bibliografía

- Dale, Nell /Weems, Chip. Programación y Resolución de Problemas con C#. Editorial McGraw Hill, México 2007.

- Sharp, John; Jagger, Jon. Microsoft Visual C# .NET Aprenda YA. McGrawHill , 1ª. Edición, 2002

Sitios de Consulta

- http://blogs.msdn.com/b/jasonz/archive/2012/08/15/visual-studio-2012-and-net-framework-4-5-released-to-the-web.aspx. Blog sobre tecnología. Consultado Enero 2016.

- https://msdn.microsoft.com/es-es/library/bb386063(v=vs.110).aspx Sitio de Microsoft Visual Studio. Consultado Diciembre 2015.

Hoja de cotejo: 01

Guía 1: Entorno de Visual Studio. Lenguajes de Programación

Alumno: Máquina No:

Docente: GL: Fecha:

EVALUACIÓN					
	%	1-4	5-7	8-10	Nota
CONOCIMIENTO	40				
APLICACIÓN DEL CONOCIMIENTO	40				
ACTITUD	20				
TOTAL	100%				