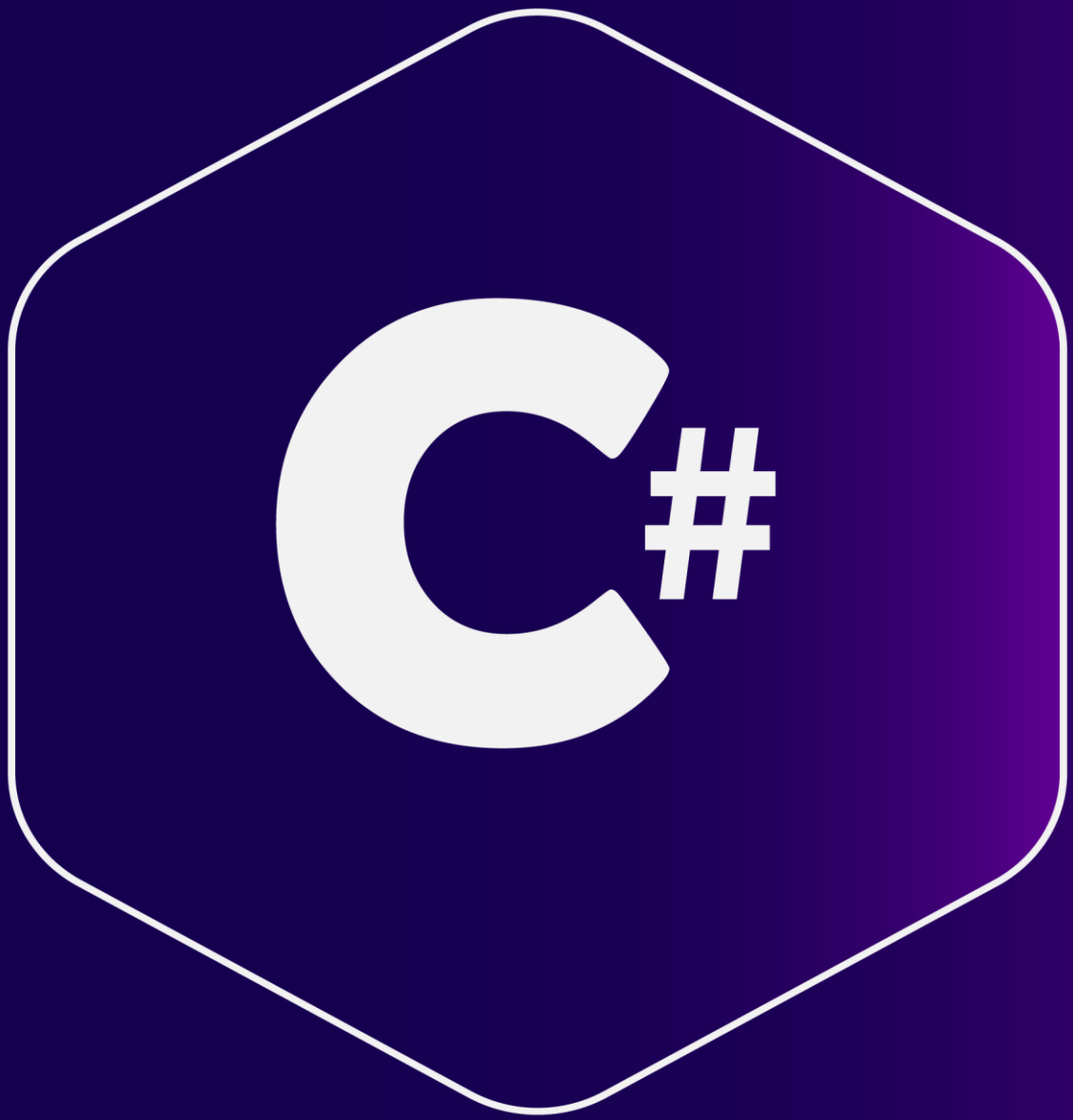
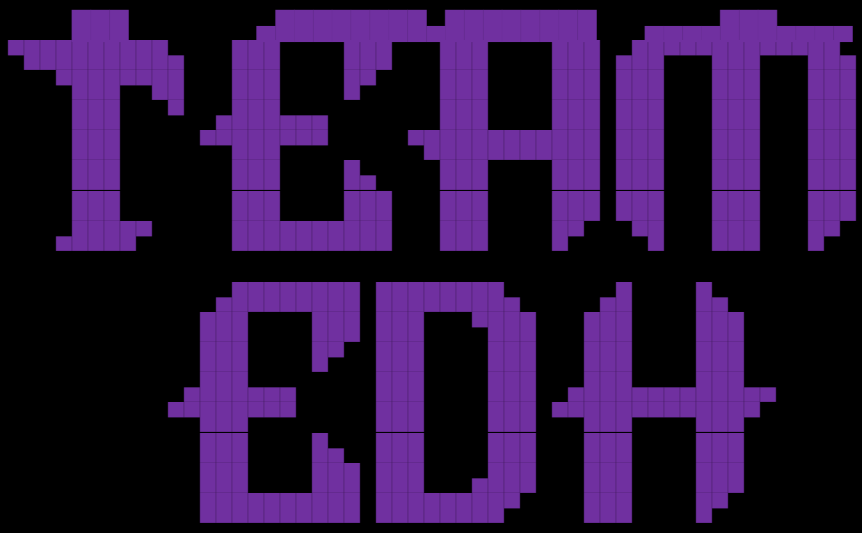
Manual de introducción a la programación



BY: ESKOGUL



Educación Desarrollo Hacking

MANUAL DE INTRODUCCIÓN A LA PROGRAMACIÓN EN C#

Hola, compañero verde, así es, compañero, ya que al igual que tu soy un “amiguito verde”, primero quiero agradecer al admin “Alpra_tdm” por esta oportunidad de hacer un manual y compartirlo con toda la comunidad “EDH”, hemos visto que muchos de la comunidad no saben programar, y los entiendo a veces puede sonar complicado la palabra “Programación”, pero en este manual verán que en menos de lo que se dan cuenta ya estarán programando lo básico.

Este manual es básico, por el momento no se tiene que descargar ningún programa adicional, pero más adelante si será necesario en futuros ejercicios, mucha de la teoría se omite para no hacer tediosa la lectura.

MENSAJE AL LECTOR:

Querido compañero verde, este manual lo realizo para enseñarte lo poco que se de programación a veces puede parecer algo complicado, pero créanme que si le ponen empeño, quien sabe, quizá la próxima vez yo sea el que este leyendo un manual de ustedes. La informática y ciberseguridad acoge a todo aquel que quiera aprender, solo debes empeñarte en ello.

RETOS:

Reto 1: Realiza un programa donde el usuario ingrese un número y se determine si es número es par o impar, y mostrar un mensaje en consola que diga si lo es o no.

pista: se utiliza un if-else tendrás que investigarlo por tu cuenta, parte de la programación es la investigación en diversas fuentes para resolver problemas.

Reto 2: Realizar un programa donde se registre el nombre de un alumno, y sus calificaciones en 5 materias, para después sacar su promedio, y mostrar en consola su nombre y su promedio, donde si su promedio es mayor a 7, salga un mensaje de aprobado, si es menor entonces, que salga un mensaje de que esta reprobado.

pista: se usa un if-else y condicional "7>".

Sin más rodeos explicare lo que haremos el día de hoy, usaremos C# miembro de la familia C, usaremos un compilador online de navegador llamado replit. https://replit.com/languages/csharp

solo entren a este enlace y ya podremos empezar con el manual sin necesidad de instalar nada en nuestras computadoras.

#HAPPYHACKING

TEMAS

Declaración de distintos tipos de variables

Mostrar las variables en consola

Operadores aritméticos como: +,-,*,/,%.

Pedir datos al usuario en la consola

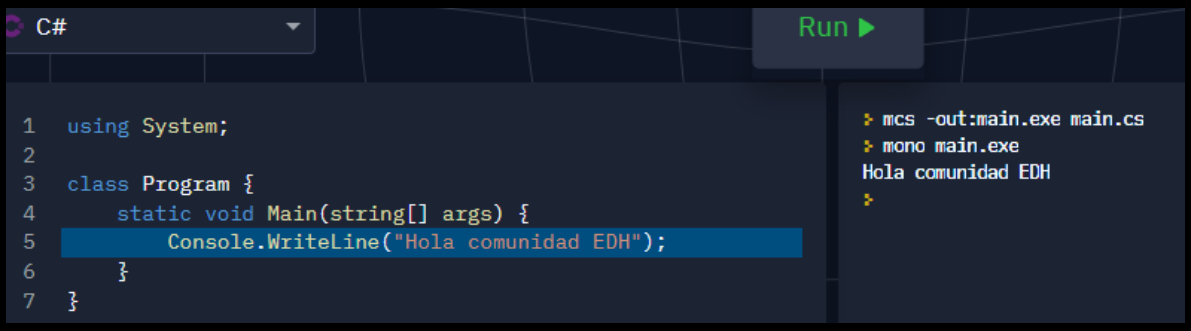
Como verán es un temario sencillo y comenzamos con el código, verán que en menos de lo que le hackean el wifi al vecino ya estarán programando.

Empezamos

```
1  using System;
2
3  class Program {
4      static void Main(string[] args) {
5          Console.WriteLine("Hello, world!");
6      }
7  }
```

```
> mcs -out:main.exe main.cs
> mono main.exe
Hello, world!
>
```

Como verán ya tenemos la interfaz lista y con la práctica típica de programación llamada **hello Word,** y si tú le das en el botón **RUN**, te lo imprimirá en una consola.



Pero nosotros le daremos un toque diferente 😇, borraremos el "**hello, world!**" y escribiremos **"hola comunidad EDH"** (es importante que este entre comillas todo **" "** ya que estas indican que algo es un texto en este lenguaje), y le damos en **RUN**.

Y felicidades así de simple ya programaste y ni siquiera hemos empezado con el temario como tal.

Nota: es importante al final de cada cosa que programemos poner un ; como lo iran viendo mas adelante, esto significa que ahí termina esa sentencia, si no lo ponen el programa marcara muchos errores y no funcionara, por lo que es importante no olvidarse del ;

Tipos de datos

Borramos el **Console.writeLine**, y nos quedamos con la consola como muestro a continuación.

```
using System;

class Program {
    static void Main(string[] args) {

    }
}
```

Y escribiremos los siguientes comandos:

int entero = 7;	int	Numero sin decimales
string texto= "Comunidad EDH";	string	Puede tener cualquier texto siempre va “ ” entre comillas
double numero = 15.5;	double	Es un número que puede poseer decimales, o no, a diferencia del int que solo son número enteros.
char caracter = ‘H’;	char	Este tipo de dato solo recibe un carácter, ni más ni menos.
bool booleano = true;	bool	Este tipo de dato solo tiene 2 estados, verdadero o falso (true o false).

```
using System;

class Program {
    static void Main(string[] args) {

        int entero= 7;
        string texto= "Comunidad EDH";
        double numero= 15.5;
        char caracter = 'H';
        bool booleano = true;

    }
}
```

Nos vendría quedando algo como esto

Las variables se componen de **[tipo de dato] [nombre de la variable] = [el valor de la variable];**

Estas sirven para aguardar datos, y como podemos ver los nombres de nuestras variables van acorde al tipo de dato que tienen asignado, pero le podemos poner cualquier nombre que queramos, siempre y cuando no tenga ningún espacio puede ser nombre_persona, pero no ~~nombre persona.~~

Se preguntarán ¿qué podemos hacer con esto?, vamos para allá, los mostraremos en consola y podremos ver como lo que escribimos aquí, se puede mostrar y pronto interactuaremos con esto.

Imprimir en consola

En este manual imprimiremos todo en consola, quiere decir que todo lo que le digamos a la computadora que haga, esta lo mostrara en la consola, que si están familiarizados con antiguos manuales de **EDH** ya sabrán que es, pero basta de tanta charla y vamos para que vean que fácil es.

Solo basta con escribir el siguiente comando:

Console.WriteLine([aquí va lo que vamos a imprimir]);	Este comando manda a imprimir en la consola, o mostrar en consola.

“los nombres” de nuestras variables antes escritas, quedando algo así:

```
1  using System;
2
3  class Program {
4      static void Main(string[] args) {
5
6          int entero= 7;
7          string texto= "Comunidad EDH";
8          double numero= 15.5;
9          char sexo = 'H';
10         bool booleano = true;
11
12         Console.WriteLine(entero);
13         Console.WriteLine(texto);
14         Console.WriteLine(numero);
15         Console.WriteLine(sexo);
16         Console.WriteLine(booleano);
17     }
18 }
```

Ahora viene la magia, los mostraremos en consola, presionamos el botón **Run**, y en el espacio de la derecha nos aparecerá la consola con nuestro código ya ejecutado.

¿Qué acabamos de hacer?

Primero creamos variables con distintos nombres, las cuales pueden guardar un tipo de dato que le indiquemos, y después con el comando "**Console.WriteLine();"**, nosotros mandamos a mostrar en consola esas variables llamándolas dentro de ese comando. Suena complejo, pero como podrán ver ya como si nada acaban de programar sus primeras instrucciones.

Concatenación

Pero ¿para qué sirve guardar estos datos y mostrarlos sin ningún sentido?, ahora veremos algo sencillo como lo es la concatenación para imprimir datos en consola, no es nada más que juntar todos estos datos de la manera en que queremos mostrarlos en consola.

Nota: las diagonales juntas así: // representan un comentario, ¿qué quiere decir? es texto o código que no se ejecutara ni se mostrara en la compilación, ni afectara al código, sirve para depurar o hacer pruebas, o como en esta práctica solo como anotación para indicar algo.

Primero tenemos que eliminar todo y poner las siguientes variables, las cuales nombramos en esta ocasión de acuerdo con su función en el programa **(consejo:** *siempre pongan nombres que ustedes identifiquen a sus variables, es importante para poder identificarlas luego, esto ya es a gusto del programador*.)

```
using System;

class Program {
    static void Main(string[] args) {

    int edad= 21; //aqui tu edad
    string presentacion= "Soy Esco";//aqui pon tu nombre
    string carrera= "Ing. Tic's"; //aqui lo que estudias
    double promedio= 90.78; //aqui tu promedio
    //aqui tu genero M masculino o F femenino O no binario
    char sexo = 'M';

    }
}
```

importante para poder identificarlas luego, esto ya es a gusto del programador.)

Y ahora armaremos una presentación personal con estos datos, concatenando texto fijo en nuestra función **Console.WriteLine(),** ejemplos:

Console.WriteLine("Hola comunidad EDH " + [variable] + " y tengo " + [variable] + " de edad");

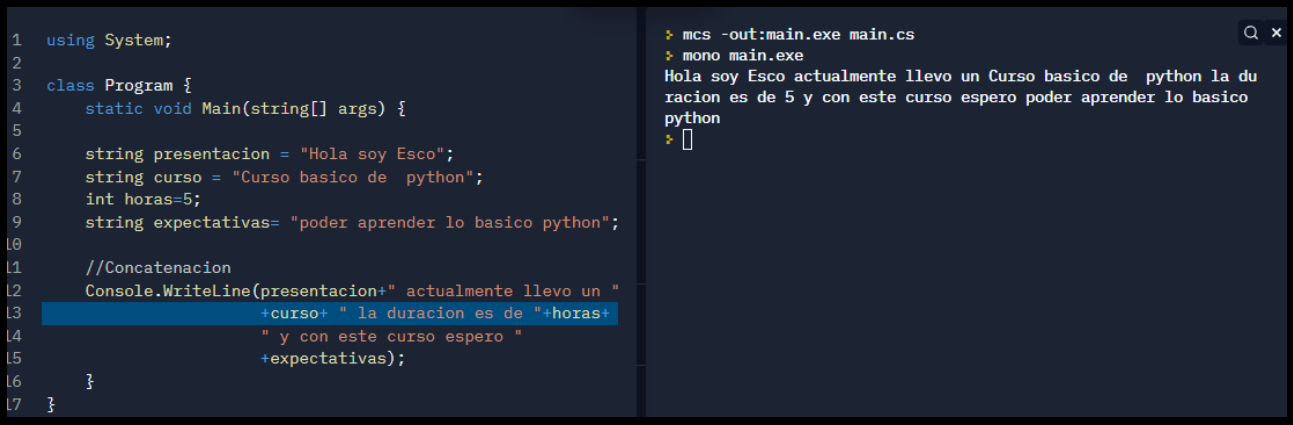
Quedándonos algo más o menos así con todo:

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

    int edad= 21; //aqui tu edad
    string presentacion= "Soy Esco";//aqui pon tu nombre
    string carrera= "Ing. Tic's"; //aqui lo que estudias
    double promedio= 90.78; //aqui tu promedio
    //aqui tu genero M masculino o F femenino O no binario
    char sexo = 'M';

        //Primera concatenacion
    Console.WriteLine("Hola comunidad EDH " + presentacion
                + " y tengo " + edad +  " de edad");

        //Segunda concatenacion
        Console.WriteLine("Estudio la carrera de: "
                          +carrera+ " y mi promedio es de:
                          +promedio);

        //Tercera concatenacion
        Console.WriteLine("soy de genero "+sexo);
    }
}
```

Nota: *las concatenaciones por temas de espacio (que saliera todo en una sola captura), las espacie con* ***enter****, pero ustedes pueden escribir cada una en una sola línea, sin ningún problema.*

Presionamos el botón **Run**, y veremos la magia, de cómo las variables que ya tenemos guardadas se concatenan o se pegan con el texto fijo del comando Console.WriteLine(), es concatenar, ahora como podemos ver, nuestras variables con texto aparentemente sin sentido ahora forman parte de una oración con sentido.

Y listo, ahora te invito a que juegues un poco con esto, puedes cambiar el contenido de las variables poner otro texto completamente diferente, por ejemplo:



Nota: *La programación es mucho de jugar con el código, programarlo, romperlo, buscar la solución, y sentir la satisfacción de ver cómo funciona después de solucionar algún error, no tengas miedo de experimentar y personalizar el programa, haz tuyo el código y no que el código te haga suyo a ti.*

Nota: *Internet, es un gran aliado para solucionar errores o dudas, el estar investigando por tu cuenta y encontrar la solución después de estar googleando durante horas es parte de la programación, también puedes apoyarte de alguien que sepa un poco sobre el tema, lo importante es no desanimarte y persistir.*

Operadores aritméticos

Estos son para realizar diversas operaciones matemáticas simples como: suma, resta, multiplicación, división y el residuo. En un momento mostraremos todos ellos y su significado, y veremos cómo usarlos.

+	Suma	Realiza una suma con valores numéricos.
-	Resta	Realiza una resta con valores numéricos.
*	Multiplicación	Realiza una multiplicación con valores numéricos.
/	División	Realiza una división con valores numéricos.
%	Residuo	Nos da el residuo de una división.

Suma

Empezaremos con la suma, la cual será en este caso de números enteros y mostrare como se puede concatenar y realizar dentro y fuera del método **Console.WriteLine().**

```
1  using System;
2
3  class Program {
4      static void Main(string[] args) {
5
6          //Numeros enteros
7          int num1=5;
8          int num2=3;
9
10 //Suma dentro de la impresion
11 Console.WriteLine(num1 + num2);
12     }
13 }
14
```

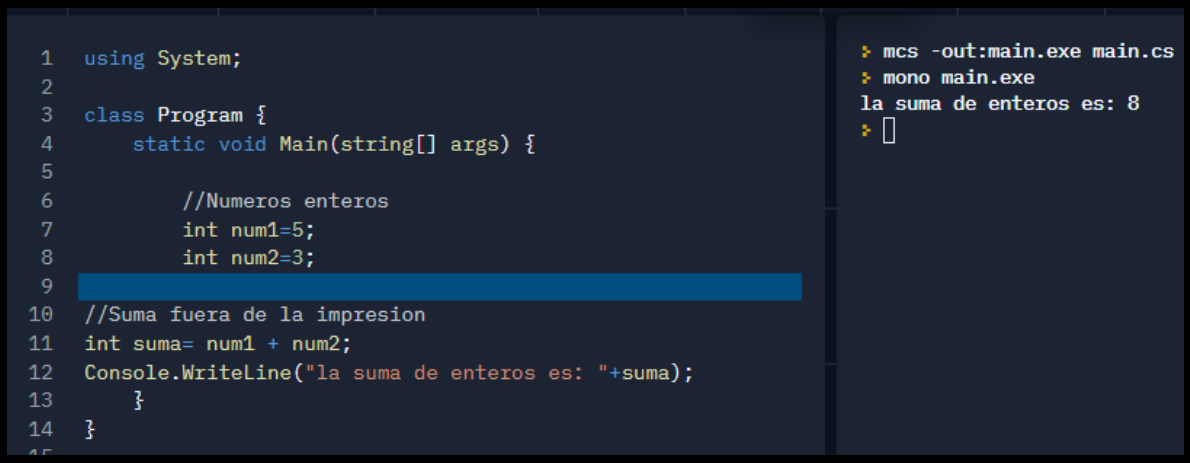
```
mcs -out:main.exe main.cs
mono main.exe
8
```

Suma dentro del la impresión: consiste en realizar la operación dentro del método para imprimir en consola, solo basta con añadir el operador aritmético entre los 2 números. Le damos en **RUN** vemos que el resultado en consola solo es 8

```
1  using System;
2
3  class Program {
4      static void Main(string[] args) {
5
6          //Numeros enteros
7          int num1=5;
8          int num2=3;
9
10 //Suma dentro de la impresion
11 Console.WriteLine(num1 + num2);
12     }
13 }
14
```

```
mcs -out:main.exe main.cs
mono main.exe
8
```

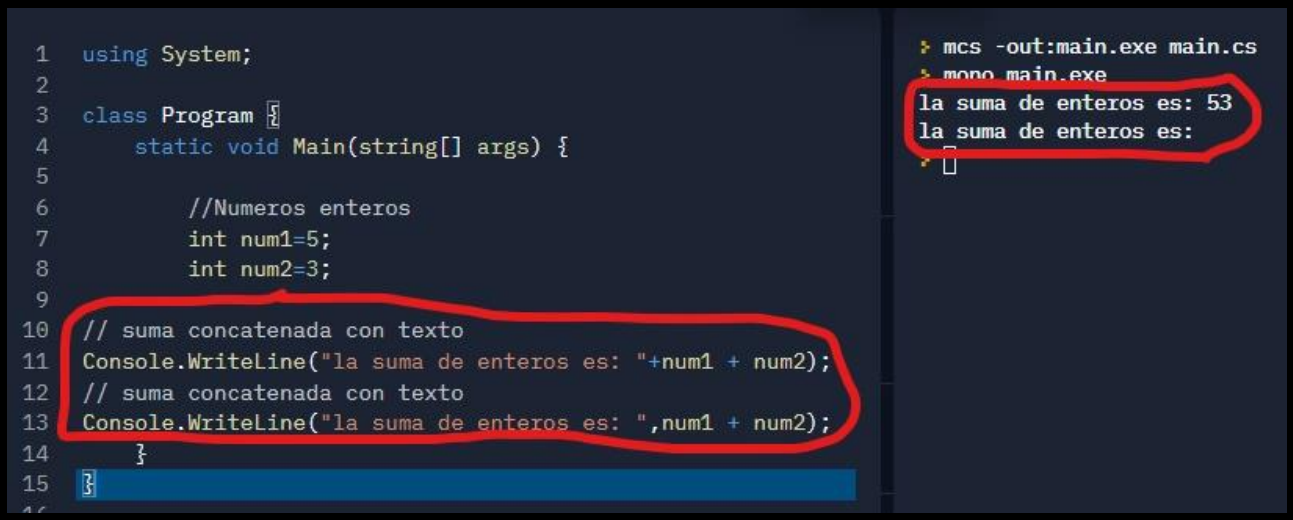
Suma fuera de la impresión: consiste en realizar en una variable aparte la suma, para después solo llamarla a imprimir, esto tiene varias ventajas ya que puede ser editable, cambiar el numero o el mismo operador aritmético.



```
using System;

class Program {
    static void Main(string[] args) {

        //Numeros enteros
        int num1=5;
        int num2=3;

//Suma fuera de la impresion
int suma= num1 + num2;
Console.WriteLine("la suma de enteros es: "+suma);
    }
}
```

```
mcs -out:main.exe main.cs
mono main.exe
la suma de enteros es: 8
```

Suma dentro de impresión y concatenada: consiste en realizar la operación dentro del mismo método **Console.WriteLine()**, pero esta vez concatenada con texto, es un poco mas complejo ya que no se puede concatenar como los demás o dará errores, por ejemplo:



```
using System;

class Program {
    static void Main(string[] args) {

        //Numeros enteros
        int num1=5;
        int num2=3;

// suma concatenada con texto
Console.WriteLine("la suma de enteros es: "+num1 + num2);
// suma concatenada con texto
Console.WriteLine("la suma de enteros es: ",num1 + num2);
    }
}
```

```
mcs -out:main.exe main.cs
mono main.exe
la suma de enteros es: 53
la suma de enteros es:
```

Podemos ver como en uno la suma es errónea y en el segundo ni siquiera la reconoce *(de hecho, esto me paso ahorita que, hacia el manual, pero estuve haciendo pruebas durante 15 minutos, rompiendo el código y experimentando hasta que llegue a la solución)*, la manera correcta es la siguiente:

```
1  using System;
2
3  class Program {
4      static void Main(string[] args) {
5
6          //Numeros enteros
7          int num1=5;
8          int num2=3;
9
10 // suma concatenada con texto
11 Console.WriteLine("la suma de enteros es: "+(num1 + num2));
12     }
13 }
```

```
> mcs -out:main.exe main.cs
> mono main.exe
la suma de enteros es: 8
>
```

Con los paréntesis **( )**, le estamos indicando al programa que primero realice la suma antes de imprimir, ya que, si no los ponemos, va a imprimir y después realizara la suma, dejándola fuera de la impresión en consola, se preguntaran ¿de dónde viene esto?, pues como Inge puedo decirles que de las matemáticas, específicamente de la jerarquía de operaciones, recordemos que todo lo que este dentro de paréntesis, tiene prioridad y se hace primero, ¡así es!, la programación también son matemáticas 🤓.

Nota: *los demás operadores utilizarán los mismos métodos de impresión, y para no alargar mas el manual no se volverán a especificar como la suma, pero es el mismo concepto, solo que, cambiando los operadores aritméticos, ahorita les muestro.*

Resta

La resta se hará con los mismos valores de la suma, y el mismo método de impresión, se mostrará todo junto, con sus diversas maneras de imprimir. Solo es de acomodar los números como quieren que quede la resta y poner el operador arietico -.

```
1  using System;
2
3  class Program {
4      static void Main(string[] args) {
5
6          //Numeros enteros
7          int num1=5;
8          int num2=3;
9
10 //Resta dentro de la impresion
11 Console.WriteLine(num1 - num2);
12  //Resta fuera de la impresion
13 double resta = num1-num2;
14 Console.WriteLine("La resta es: "+ resta);
15 // Resta concatenada con texto
16 Console.WriteLine("la suma de enteros es: "+(num1 - num2));
17     }
18 }
19
20
```

```
> mcs -out:main.exe main.cs
> mono main.exe
2
La resta es: 2
la suma de enteros es: 2
>
```

Multiplicación

La multiplicación será con los mismos valores de la resta, solo es de cambiar el operador aritmético en las operaciones y listo.

```
1  using System;
2
3  class Program {
4      static void Main(string[] args) {
5
6          //Numeros enteros
7          int num1=5;
8          int num2=3;
9
10 //Multiplicacion  dentro de la impresion
11 Console.WriteLine(num1 * num2);
12 //Multiplicacion fuera de la impresion
13 double multipli = num1*num2;
14 Console.WriteLine("La multiplicacion es: "+ multipli);
15 // Multiplicacion concatenada con texto
16 Console.WriteLine("la multi de enteros es: "+(num1 * num2))
17     }
18 }
```

```
> mcs -out:main.exe main.cs
> mono main.exe
15
La multiplicacion es: 15
la multi de enteros es: 15
>
```

División

En esta ocasión se modificará las variables **num1** y **num2** por **doubles** en vez de **int** para poder ver los decimales en la división, se cambiará el operador aritmético y quedaría igual los métodos de impresión. \

```
1  using System;
2
3  class Program {
4      static void Main(string[] args) {
5
6          //Numeros enteros
7          double num1=5;
8          double num2=3;
9
10 //Division  dentro de la impresion
11 Console.WriteLine(num1 / num2);
12 //Division fuera de la impresion
13 double divi = num1/num2;
14 Console.WriteLine("La division es: "+ divi);
15 //Division concatenada con texto
16 Console.WriteLine("la divi de enteros es: "+(num1 / num2));
17     }
18 }
```

```
> mcs -out:main.exe main.cs
> mono main.exe
1.66666666666667
La division es: 1.66666666666667
la divi de enteros es: 1.66666666666667
>
```

Residuo

Para fines de comprobación se usarán los mismos datos de la división solo cambiando el operador aritmético por le dé residuo que es %.

```
using System;

class Program {
    static void Main(string[] args) {

        //Numeros enteros
        double num1=5;
        double num2=3;

//Residuo  dentro de la impresion
Console.WriteLine(num1 % num2);
//Residuo fuera de la impresion
double residuo = num1%num2;
Console.WriteLine("El Residuo es: "+ residuo);
//Residuo concatenada con texto
Console.WriteLine("El resi de enteros es: "+(num1 % num2));
    }
}
```

```
> mcs -out:main.exe main.cs
> mono main.exe
2
El Residuo es: 2
El resi de enteros es: 2
>
```

Y básicamente estos son los operadores aritméticos, y como se emplean, se pueden utilizar enteros o números reales como decimales, ya depende de lo que necesites hacer, de nuevo de exhorto a practicar y jugar con las posibilidades que da la programación.

Variables y constantes

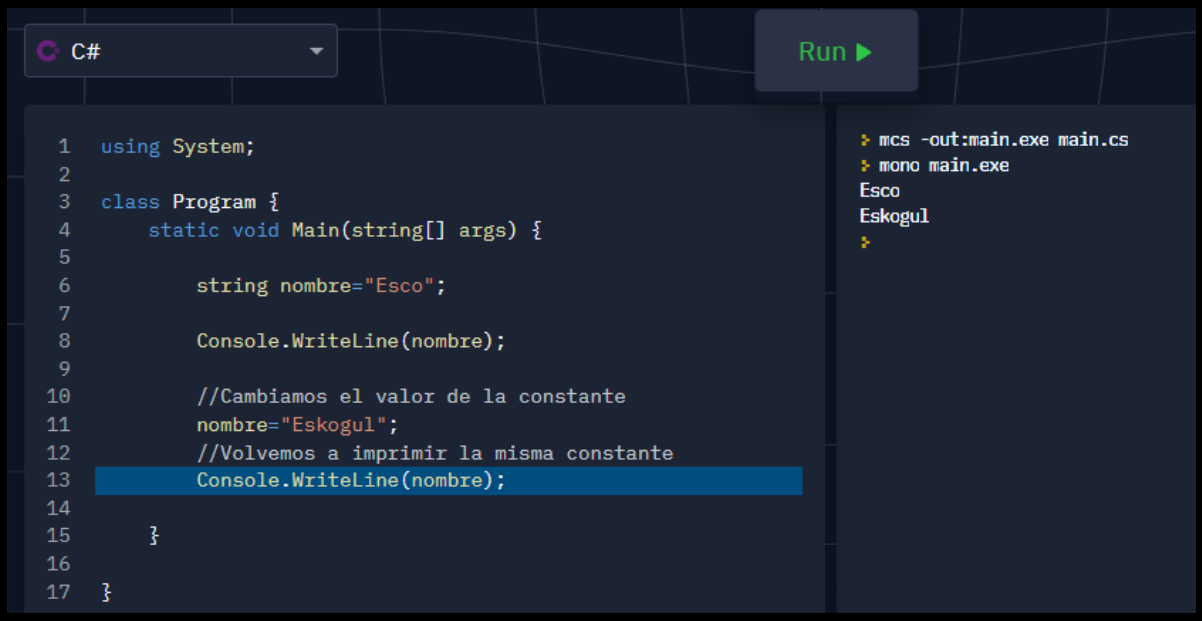
Todo lo que hemos venido haciendo hasta ahora ha sido ingresando los datos nosotros mismos, como programadores que somos, pero un programa tiene que interactuar con un usuario, no solo con el programador, (todos los datos que hemos venido usando ahorita son constantes, pero para no revolvernos antes lo llamaba como variables), ahora explicare como esta todo esto.

Variable	Int edad ; String nombre ;	Como podrán ver una variable, no tiene valor asignado ya que su valor puede “variar”, de ahí su nombre.
Constante	Int edad = 21; String nombre = “Esco” ;	Una constante se le conoce a un dato el cual ya tiene un valor asignado y establecido va después del = (que en este caso es asignación). Las constantes pueden también ser cambiadas y volverse variables, pero por lo general no se hace ya que al inicio del código ya se le asigno un valor y cambiarlo más adelante puede generar conflictos en el programa.

Muestra de una variable y una constante

Como podemos ver las variables están en blanco sin valor asignado, y las constantes si tienen un valor asignado después del "=", y si tratamos de imprimirlas, las constates se muestran en consola sin ningún problema, pero las variables al todavía no tener un valor asignado tiraran un error.

Como es útil esto, pues si quieres dejar ya un dato con un valor establecido al arrancar un programa dejamos una constante, y si queremos un dato que no tenga valor inicial por ejemplo **para que un usuario ingrese datos**, ya se usaría una variable.



Nota: *siempre que se vuelva a iniciar el programa, la constante mantendrá el valor que se le asigno al inicio, así le sea cambiado el valor mas adelante, este siempre mantendrá su valor original al iniciar el programa, por eso es una constante. Deja te muestro:*

Como puedes ver se puede usar una constante, después cambiarle su valor inicial, pero este segundo cambio siempre será “temporal”, ya que si volvemos a ejecutar el programa el valor original siempre será el primero que le asignamos, ya que el programa se ejecuta de arriba hacia abajo. No se recomienda andar cambiando las constantes demasiado, ya que, en programas más complejos, el modificar estas constantes puede afectar otras partes del código donde sea usada esta constante, por eso es importante siempre tener un buen uso de sus variables y constantes.

Pedir datos a un usuario en consola

Ahora si viene lo chido, diría un youtuber famoso por ahí, muchas veces como programadores, tendremos que hacer programas que interactúen con el usuario, y este tendrá que ingresar sus propios datos, dependiendo que le pidamos, y para eso al menos en consola tenemos la función **Console.ReadLine();** dicha función va dentro de una variable asignada.

String nombre= Console.ReadLine();	Leer dato en consola ingresado por teclado.	Este comando ira dentro de una variable que recibirá el dato que estamos pidiendo.

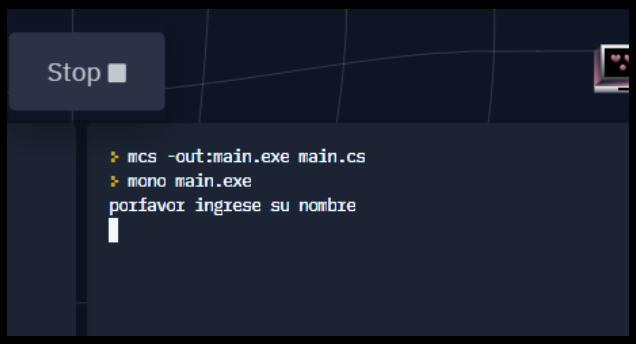
Ingresar un string

Primero hacemos un código, declarando la variable a usar, después con una función que ya conocemos **Console.WriteLine();** mandamos un mensaje para el usuario que diga **“ingrese su nombre”**, para después hacer uso de nuestra variable antes declarada llamada **nombre**, pero esta ves le asignamos un valor que en este caso es la función **Console.ReadLine();** y como toque final usamos ese dato que el usuario ingresara en una concatenación de texto, dejen les muestro:

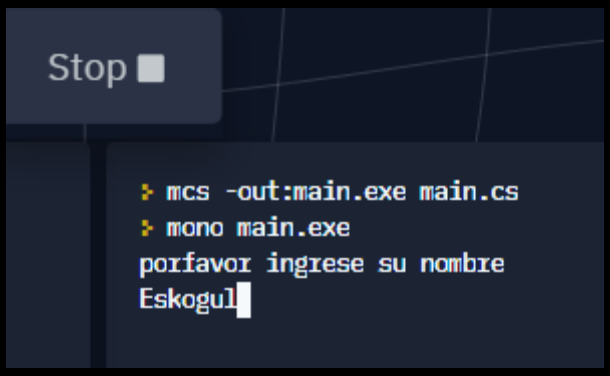
```
1  using System;
2
3  class Program {
4      static void Main(string[] args) {
5
6          //Declaramos las variables
7          string nombre;
8
9  //Mensaje para indicar al usuario que ingrese dato
10 Console.WriteLine("porfavor ingrese su nombre");
11 //ingresamos la varibale con la funcion para recibir datos
12         nombre=Console.ReadLine();
13
14 //hacemos uso del dato ingresado
15         Console.WriteLine("Hola "+nombre+" bienvenido");
16
```

Corremos el programa con el botón **RUN**, y pasara lo siguiente:

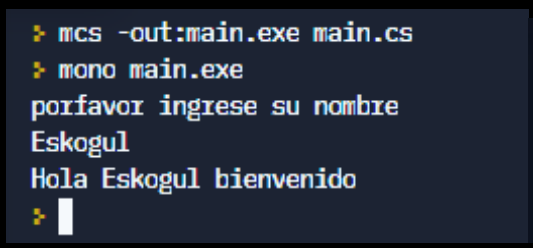
Nos sale el mensaje para ingresar un dato.

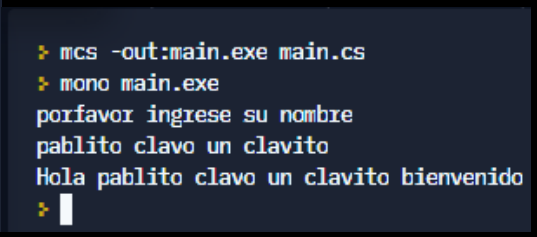


ingresamos el dato y le damos enter en el teclado:



Como podemos ver se ejecuta la concatenación después de presionar **enter**, y el nombre que ingresamos aparece, es lo interesante de esto, ahora el usuario puede poner su nombre y es completamente personalizable el mensaje final:





Pedir un int o double

Ahora entramos a terreno un poco más complicado, ya que la función **Console.ReadLine();** si lee datos, pero solo string /cadena de caracteres, pero no int o enteros, entonces ¿como pedimos un dato de tipo int o double? Es sencillo, pero a la vez tedioso, ya que tenemos que agregar ooooootra función fuera del ReadLine, deja te muestro:

Para enteros o int:

```
6           //Declaramos las variables
7           int promedio;
8           double sueldo;
9
10  //Mensaje para indicar al usuario que ingrese dato
11  Console.WriteLine("porfavor ingrese su promedio escolar");
12  //ingresamos la varibale con la funcion para recibir datos
13          promedio=int.Parse(Console.ReadLine());
14
```

Podemos ver que se agrega la función **int.Parse(Console.Read Line);**

Para dobles o double:

```
Console.WriteLine("porfavor ingrese su sueldo");
        sueldo=double.Parse(Console.ReadLine());
    }
```

Podemos ver que se agrega la función **double.Parse(Console.Read Line);**

```
1   using System;
2
3   class Program {
4       static void Main(string[] args) {
5
6           //Declaramos las variables
7           int promedio;
8           double sueldo;
9
10  //Mensaje para indicar al usuario que ingrese dato
11  Console.WriteLine("porfavor ingrese su promedio escolar");
12  //ingresamos la varibale con la funcion para recibir datos
13          promedio=int.Parse(Console.ReadLine());
14
15  Console.WriteLine("porfavor ingrese su sueldo");
16          sueldo=double.Parse(Console.ReadLine());
17
18  Console.WriteLine("su promedio: "+promedio+" su sueldo: "
19                    +sueldo);
20      }
21
22  }
```

```
> mcs -out:main.exe main.cs
> mono main.exe
porfavor ingrese su promedio escolar
8
porfavor ingrese su sueldo
556.876
su promedio: 8 su sueldo: 556.876
>
```

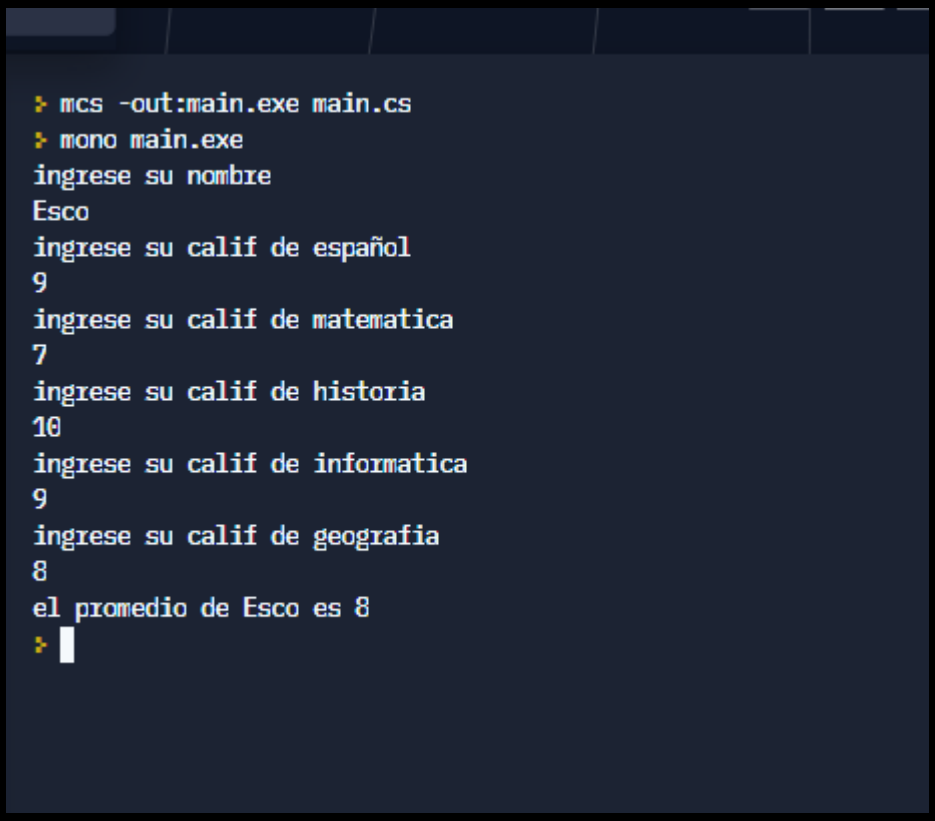
Repasemos lo aprendido recientemente

Haremos un programa que recibirá string, 5 enteros y realizaremos operaciones con dichos enteros y concatenaremos (mucha atención porque será de utilidad en uno de los retos) no explicare demasiado, para que por ustedes mismos razonen el código, vean lo que hice y así puedan replicarlo ustedes.

```
1  using System;
2
3  class Program {
4      static void Main(string[] args) {
5
6  //Declaramos las variables
7  string nombre;
8  int espanol;
9  int matematica;
10 int historia;
11 int informatica;
12 int geografia;
13 double promedio;
14
15 Console.WriteLine("ingrese su nombre");
16     nombre=Console.ReadLine();
17 Console.WriteLine("ingrese su calif de español");
18     espanol=int.Parse(Console.ReadLine());
19 Console.WriteLine("ingrese su calif de matematica");
20     matematica=int.Parse(Console.ReadLine());
21 Console.WriteLine("ingrese su calif de historia");
22     historia=int.Parse(Console.ReadLine());
```

```
23 Console.WriteLine("ingrese su calif de informatica");
24     informatica=int.Parse(Console.ReadLine());
25 Console.WriteLine("ingrese su calif de geografia");
26     geografia=int.Parse(Console.ReadLine());
27
28 promedio= (espanol+matematica+historia+informatica
29     +geografia)/5;
30
31 Console.WriteLine("el promedio de " +nombre+ " es "
32                 +promedio);
33
34
```

Y el resultado de todo este código es lo siguiente:



Despedida

Y bueno compañero verde, esto sería todo por este manual, esta es la primera parte de la lo básico de programación en C#, dicen por ahí que si aprendes a programar en un lenguaje, ya sabes programar en los demás, y en parte es cierto y parte no, ya que si bien agarras la lógica de la programación y ese pensamiento analítico de un programador, aun para aprender un nuevo lenguaje tienes que aprender sus reglas, ya que cada lenguaje las tiene, pero es cierto que si ya sabes programar en un lenguaje ya es mucho más fácil aprender otro.

Les mando un saludo a todos ustedes la comunidad EDH y les deseo un buen aprendizaje, por cierto, varias de las notas son personales, son cosas que me hubiera gustado escuchar cuando estaba aprendiendo, pero no fue así, aun así, quise compartirlas con ustedes para que no cometan los mismos errores que yo en su momento.

TEAM EDH