

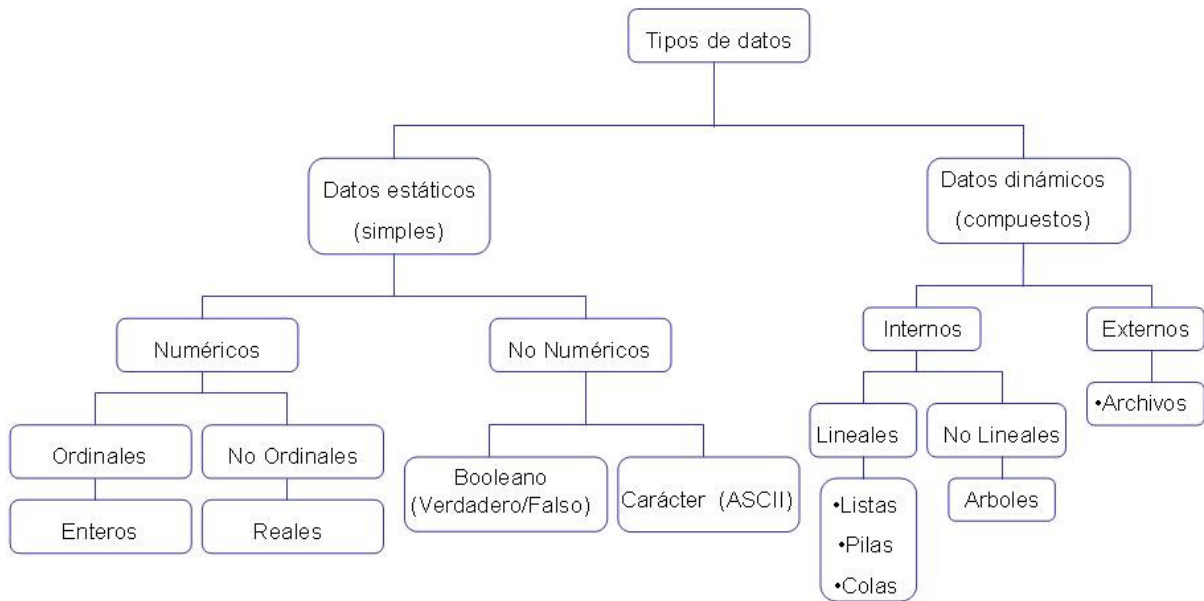


TEMA1. ESTRUCTURAS DE DATOS BÁSICAS

Un **tipo de datos** es una descripción formal del conjunto de valores (o *dominio*) que una variable o expresión de dicho tipo puede tener, junto con el conjunto básico de operaciones que pueden ser aplicadas a estos valores.

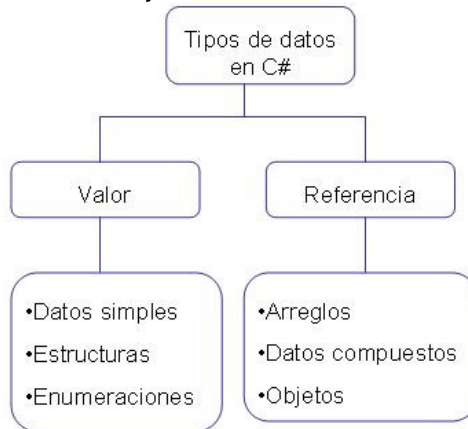
Es importante considerar la naturaleza de la información que se van a manipular dentro de un programa, ya que esta determina el espacio en memoria que se va a utilizar y los operadores válidos para cada tipo de dato.

Básicamente los datos están clasificados de la siguiente manera:



Tipos de datos en C#

Los tipos de datos en C# se clasifican en: **tipos valor** y **tipos referencia**. Una variable de tipo valor almacena directamente un valor (datos en general), mientras que una variable de un tipo referencia permite almacenar una referencia a un objeto.





TEMA1. ESTRUCTURAS DE DATOS BÁSICAS

En c# todos los tipos de datos heredan de la clase System.Object

CTS	Alias C#	Descripcion	Ejemplo
System.Object	Object	Es el tipo base de todos los tipos	object obj=null;
System.String	String	Secuencia de caracteres Unicode	string st="Sierra";
System.Sbyte	Sbyte	entero con signo de 8 bit (1 byte)	sbyte val=12;
System.Int16	Short	entero con signo de 16 bit	short val=12;
System.Int32	Int	entero con signo de 32 bit	int val=12;
System.Int64	Long	entero con signo de 64 bit	long val=12; long val2=34L;
System.Boolean	Bool	tipo booleano (cierto - falso)	bool opc=false; bool opc=true;
System.Char	Char	tipo caracter, se corresponde con un Unicode	char val='h';
System.Byte	Byte	entero sin signo de 8 bit	byte val=12; byte val2=12U;
System.UInt16	Ushort	entero sin signo de 16 bit	ushort val=12; ushort val2=12U;
System.UInt32	UInt	entero sin signo de 32 bit	uint val=12; uint val2=12U;
System.UInt64	Ulong	entero sin signo de 64 bit	ulong val=12; ulongval2=12U; ulong val3=24L ulong val4=34UL
System.Single	float	numero en punto flotante con precisión simple	float val=12.23F float val2=12.23
System.Double	double	numero en punto flotante con precisión doble	double val=12.23 double val2=12.23D
System.Decimal	decimal	numero decimal con 28 dígitos significativos	decimal val=1.23M

Tipos enteros

Las variables de tipos de datos enteros pueden almacenar únicamente **valores numéricos sin decimales**. Si se intenta almacenar un valor con decimales en una variable entera, se producirá un **truncamiento** del valor.

El tipo entero básico es **int**. Existen además las variantes **short** que permite almacenar valores más pequeños (con el consiguiente ahorro de espacio en memoria), y el **long**, que soporta números más grandes (pero con mayor gasto de memoria).

El valor máximo que se puede almacenar en las variables de cada uno de estos tipos depende de la arquitectura del computador y del compilador que se estén empleando.

Constantes de Tipo Entero

Para declarar valores constantes de tipo entero, hay que reconocer si se trata de enteros de tipo int, o long. Por ejemplo, la inicialización de las siguientes dos variables considera un valor escrito



TEMA1. ESTRUCTURAS DE DATOS BÁSICAS

como constante entera, diferenciando que para el `int`, se usa un número simple y para el `long`, se agrega una 'L' al final:

```
int n1 = 10;  
long n2 = 120390L;
```

Tipos reales

Las variables de tipos de datos reales pueden almacenar valores numéricos con decimales.

El tipo real básico es **float**. Existe también la variante **double**, que permite almacenar valores en **doble precisión**, más grandes y con mayor cantidad de decimales. Sin embargo, el uso de este tipo provoca un mayor gasto de memoria que si se empleara el tipo básico **float**. La cantidad de decimales, así como el valor máximo que se puede almacenar en variables de estos tipos depende de la arquitectura del computador y del compilador que se estén empleando.

Constantes de Tipo Real

Al igual que en el caso de los enteros, los valores constantes de tipo real se diferencian en su declaración al identificarse como `float` o `double`, aún cuando ambos usan valores decimales en su declaración. El caso de los `float`s requiere agregar una 'F' al final, tal como se indica en el siguiente ejemplo:

```
float n1 = 10.0F;  
double n2 = 120.390;
```

Caracteres: Tipo de datos *char*

Las variables de tipos de datos **char**, pueden almacenar un carácter del **código ASCII extendido** (256 caracteres). En realidad, lo que se guarda no es el carácter en sí, sino el código correspondiente.

Por lo tanto, puede verse al tipo **char** como un subtipo del tipo **int**, que puede almacenar enteros entre 0 y 255. De hecho, las expresiones de tipo **char** pueden manipularse como enteros.

Las constantes de tipo **char** se representan delimitadas por comillas simples. Así, para escribir el carácter que representa la letra A mayúscula escribimos: **'A'**

El hecho de que las expresiones de tipo **char** se consideren como enteros permite llevar a cabo **conversión de tipos entre expresiones char y expresiones int**. Por ejemplo, es posible llevar a cabo las siguientes asignaciones, las cuales son todas equivalentes y almacenan en la variable correspondiente la letra **A** mayúscula (código ASCII 65).

```
char ch;  
int i;  
ch = 65;  
i = 'A';  
ch = 'A';  
i = 65;
```

Además, puede emplearse el símbolo `\` para denotar caracteres mediante su **código ASCII**. Con esta notación, el carácter **'A'** puede representarse también como **'65'**

Esta notación es muy útil para representar caracteres especiales, que no aparecen en los teclados y no se pueden ver en la pantalla. Este es el caso del **carácter nulo** (ASCII 0): **'\0'**, el cual se empleará más adelante para indicar el final de los **strings**. Así, las siguientes son representaciones equivalentes en C para el carácter constante que representa la letra A mayúscula:

```
'A'  
65  
'65'
```

La preferencia en usar la tercera notación y no la segunda es por simple claridad, para dejar bien claro que se está manipulando un carácter y no un número (aunque para el C es lo mismo). Otros caracteres especiales que se pueden representar mediante esta notación son:

'\n': Carácter de nueva línea (a veces usado en **Console.WriteLine**, más adelante)



TEMA1. ESTRUCTURAS DE DATOS BÁSICAS

`'\t'`: Tabulador (también útil en **Console.WriteLine**)

`'\\'`: Caracter `\` (esta es la única forma de especificarlo)

Más interesante resulta la manera de organizar los caracteres como cadenas de **char**, de manera que puedan manipularse cadenas de texto (**string**). Las cadenas de texto se especifican entre comillas dobles: **"iic1102 sección"**.

Cadenas de Texto (string)

Los strings esencialmente son cadenas de texto, compuestas por uno o más caracteres. Si bien no se incluyen dentro de los tipos básicos de C#, como son el `int`, el `double`, el `char`, efectivamente se reconoció su enorme importancia al haberse incluido una clase *String* y su alias *string* en la base del lenguaje C#.

Internamente, un string, al ser una lista de caracteres, se puede equiparar con un arreglo de char, sin embargo, dada su propia necesidad funcional, se han agregado diversos atributos y métodos a la clase

`String`, para simplificar y potenciar su utilización, hasta cierto punto ocultando su real implementación de arreglo interna.

Operadores básicos sobre un string

Adicionalmente, algunos de los operadores más recurridos dentro de las expresiones del lenguaje C# también son válidos para trabajar con strings, tal es el caso de la asignación (`=`), la igualdad (`==`), y la suma (`+`).

Sufijos para variables

Sufijo	Tipo	Ejemplo
	char	a'
U	uint	35U
L	long	35L
D	double	3.1416D
F	float	3.1416F
M	decimal	35M

Códigos de escape

Carácter	Código de escape especial
Comilla simple	<code>'\'</code>
Comilla doble	<code>'\"'</code>
Carácter nulo	<code>'\0'</code>
Alarma	<code>'\a'</code>
Retroceso	<code>'\b'</code>
Salto de página	<code>'\f'</code>
Nueva línea	<code>'\n'</code>
Retorno de carro	<code>'\r'</code>
Tabulación horizontal	<code>'\t'</code>
Tabulación vertical	<code>'\v'</code>
Barra invertida	<code>'\\'</code>



TEMA1. ESTRUCTURAS DE DATOS BÁSICAS

Operadores aritméticos:

Operadores aritméticos	
Suma	+
Resta	-
Producto	*
División	/
Módulo	%

Operadores Lógicos:

Operadores lógicos	
AND	&&
OR	
NOT	!
XOR	^

Operadores relacionales:

Operadores Relacionales	
Igual	==
Diferente	!=
Mayor que	>
Menor que	<
Mayor o igual que	>=
Menor o igual que	<=

Entrada y salida de datos en C#

Para cazar un dato desde teclado se utiliza el método Read o ReadLine de la clase Console:

```
Variable = Console.ReadLine(); //Mientras no se oprima "enter"  
Variable = Console.Read(); //Un caracter
```

Para imprimir información en pantalla se utiliza el Write y WriteLine:

```
Console.Write("Hola"); // Imprime Hola sin salto de línea  
Console.WriteLine("Hola"); // Imprime Hola con salto de línea  
Console.WriteLine(Variable); // Imprime el valor de Variable  
Console.WriteLine("Texto {0}, {1}={2}", var1, var2, var3); //Imprime 3 variables  
Console.WriteLine("Info = {0} * {1} * {2} * {3}", var1,var2,var3,var4); //Imprime 4 variables
```

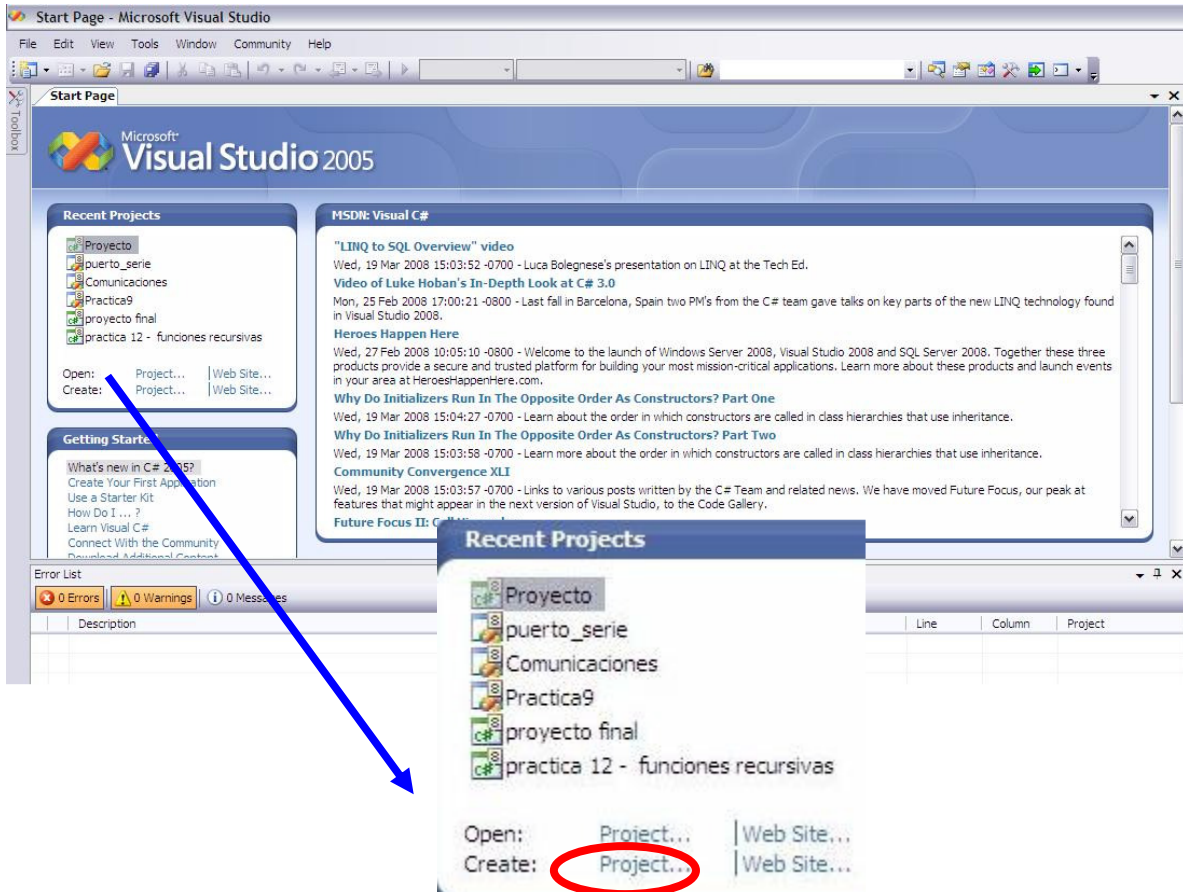
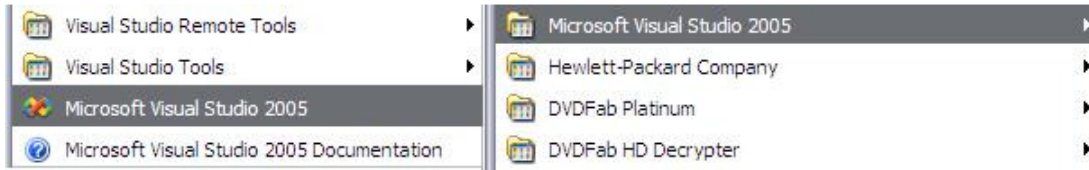


Ejercicios de la práctica:

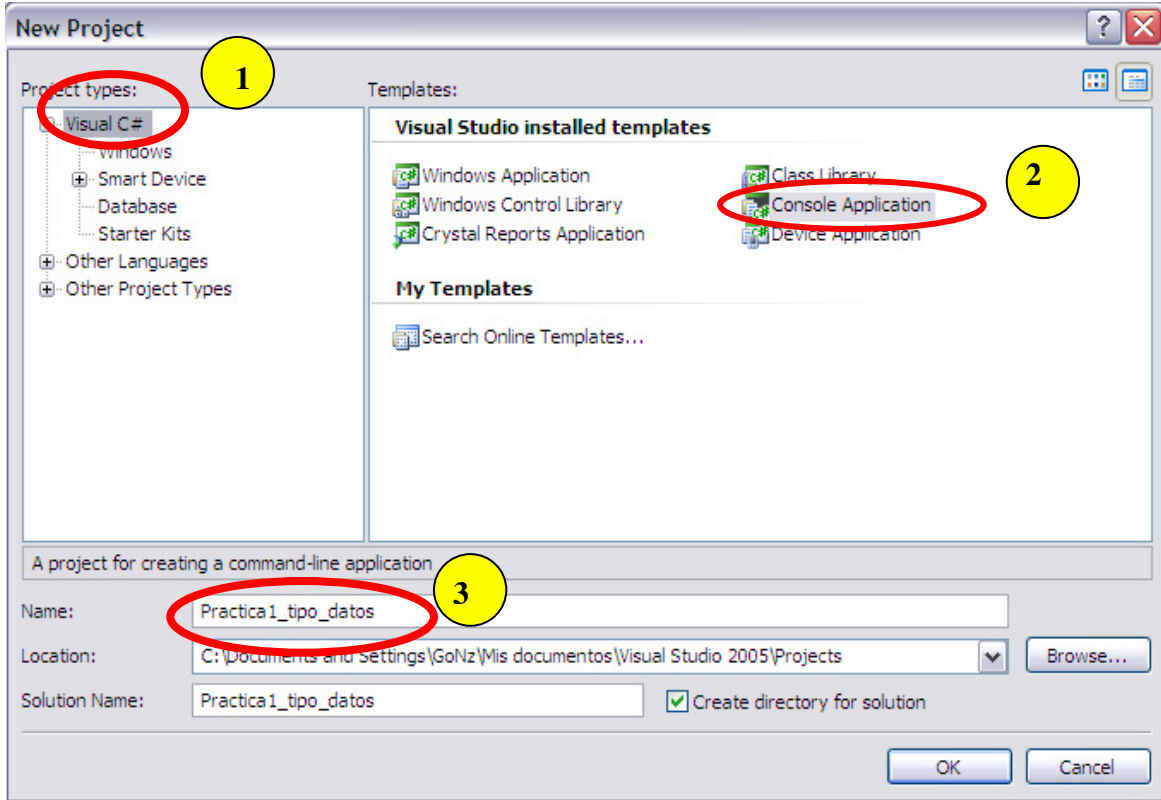
Manejo de Tipos de Datos. Con estos ejemplos podrás comenzar a trabajar con los tipos de datos existentes en C#

Ejercicio 1

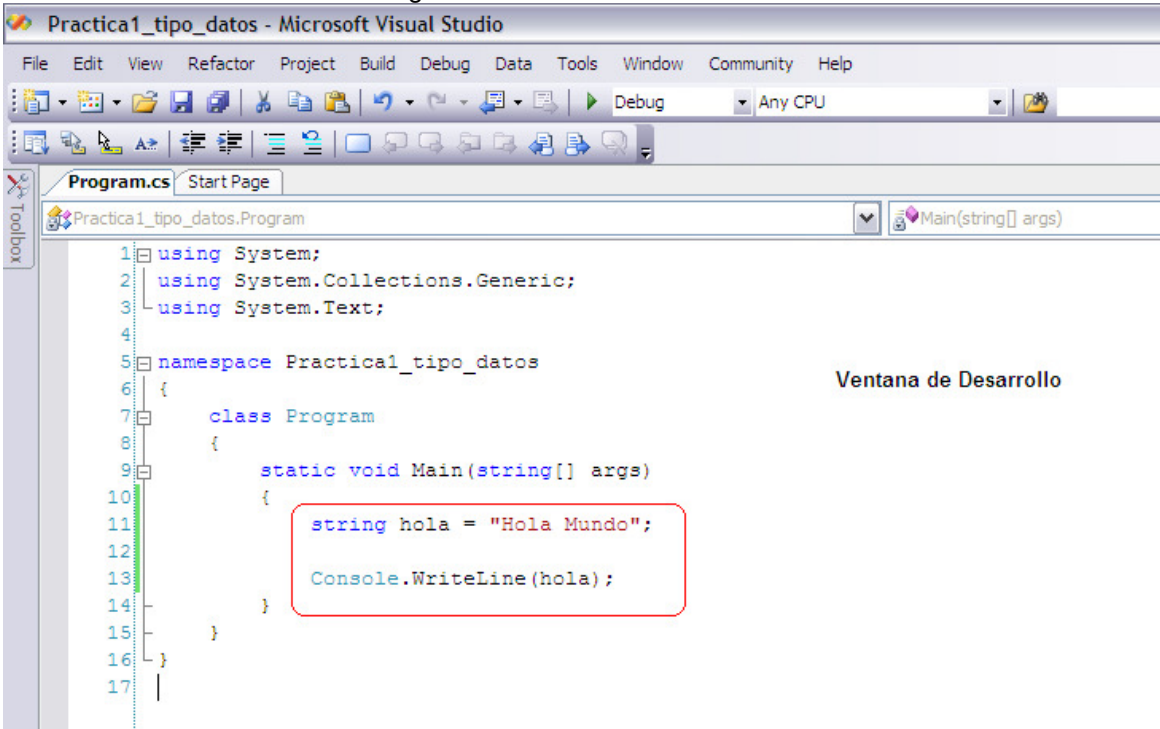
1. Abre el Visual Studio



2. Crea un nuevo proyecto de **Aplicación de Consola (Console Application)**, llámalo Practica2_tipo_datos

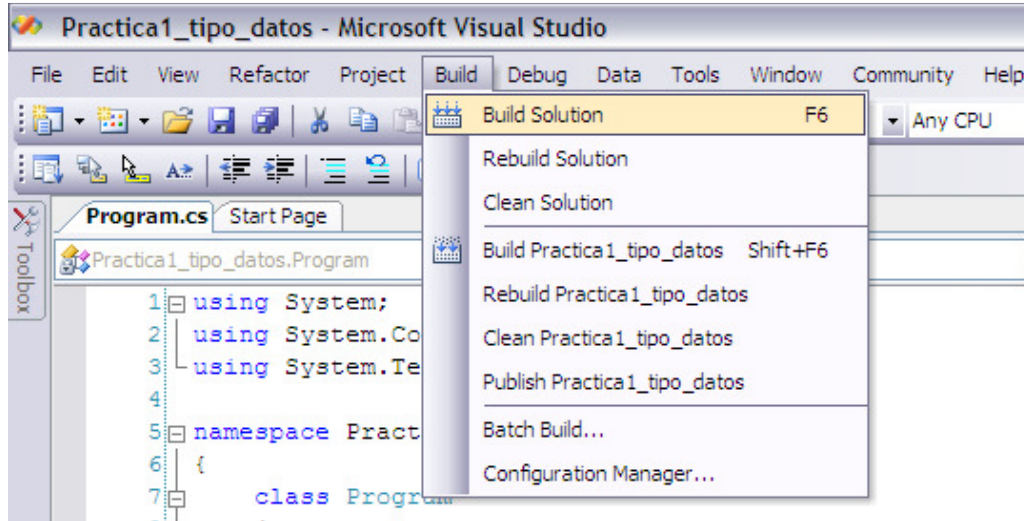


3. Escribe las siguientes líneas

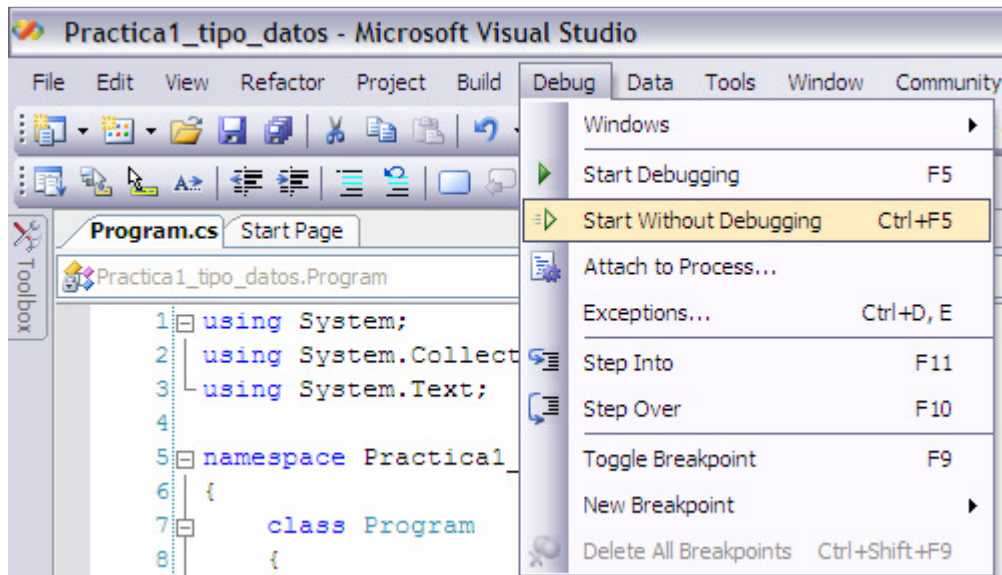




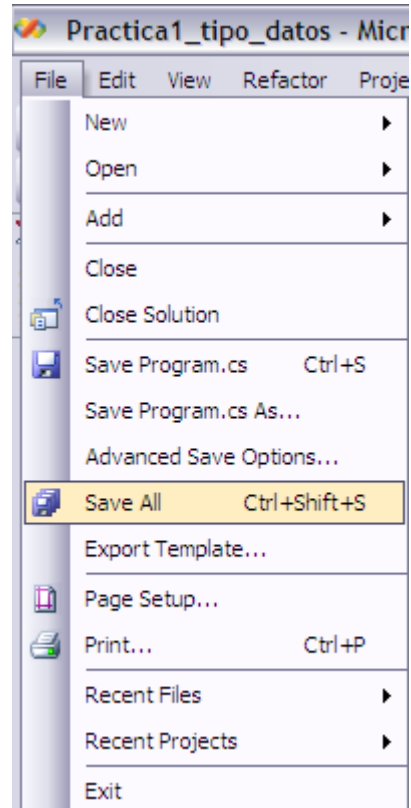
4. Compile el programa seleccionando el menú principal > Build > Build Solution



5. Ejecuta el programa seleccionando Menú > Debug > Start Without Debugging



6. Guarda el proyecto Seleccionando el menú principal > Archivo > Guardar Todo (File > Save All)

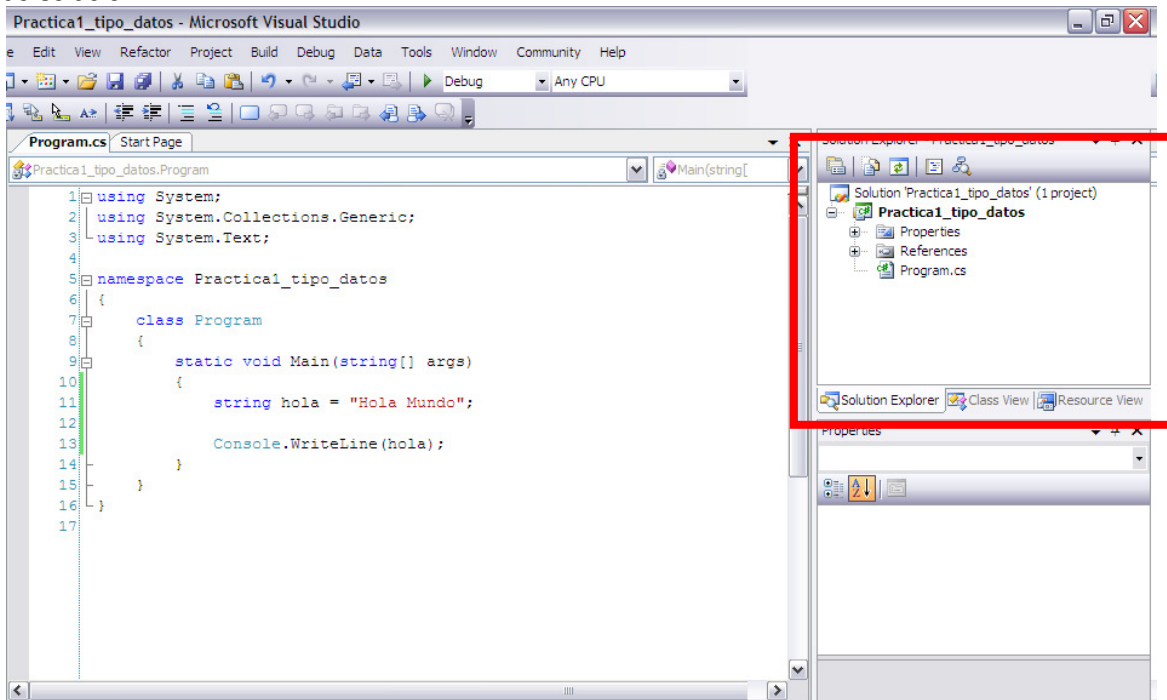


7. Anota tus observaciones.

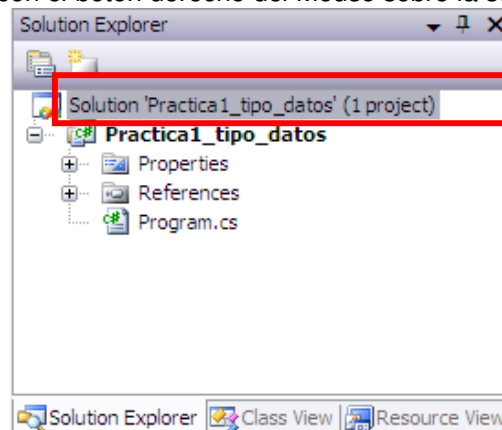


TEMA1. ESTRUCTURAS DE DATOS BÁSICAS

Ejercicio 2. Vamos a crear un nuevo proyecto dentro de la solución, para esto localiza la ventana de solución



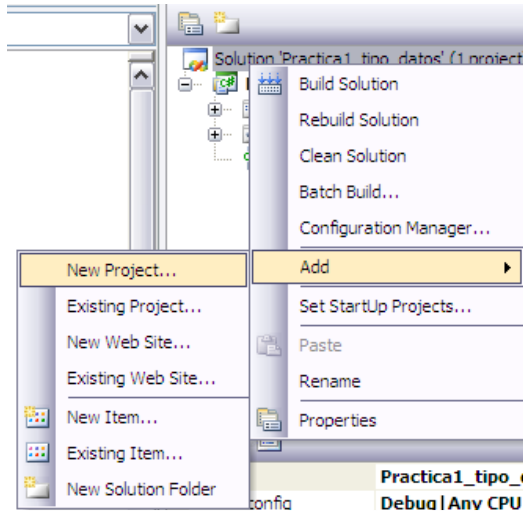
1. En la ventana da clic con el botón derecho del Mouse sobre la solución.



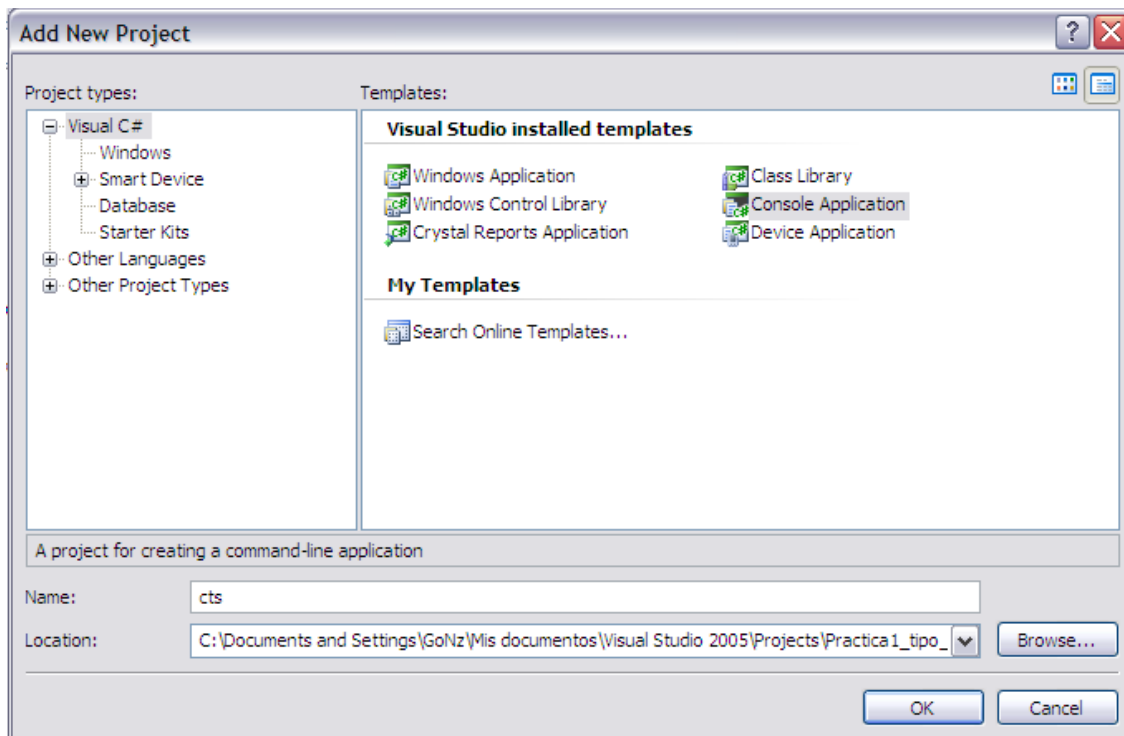
2. En el menú flotante selecciona Nuevo > Proyecto



TEMA1. ESTRUCTURAS DE DATOS BÁSICAS



3. Al nuevo proyecto llámalo tipos_system.





TEMA1. ESTRUCTURAS DE DATOS BÁSICAS

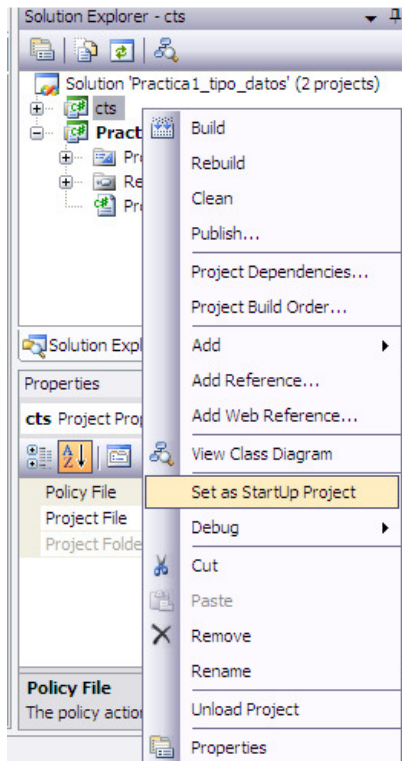
4. En el nuevo proyecto escribe el siguiente código:

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

namespace tipos_system
{
    class Program
    {
        static void Main(string[] args)
        {
            int Entero = 50;
            char Caracter = 'a';
            float Flotante = 1.05F;
            string Cadena = "Tecnicas";
            double Doble = 10.9999999D;

            Console.WriteLine(Entero.GetType());
            Console.WriteLine(Caracter.GetType());
            Console.WriteLine(Flotante.GetType());
            Console.WriteLine(Cadena.GetType());
            Console.WriteLine(Doble.GetType());
        }
    }
}
```

5. Para que ahora se ejecute este proyecto ve a la ventana de soluciones, colócate sobre el proyecto tipos_system y con el botón derecho del Mouse selecciona la opción **Set as StartUp Project**





6. Compíllalo, ejecútalo y anota tus observaciones.

Ejercicio 3. Para cada uno de los siguientes ejercicios realiza un proyecto diferente. En los ejercicios que requieran operaciones considere variables inicializadas con algún valor.

1. Hallar la superficie de un triángulo conociendo la base y la altura.
2. Dado el radio de una esfera calcular el volumen.
3. Escribir un programa que contenga instrucciones que muestren por pantalla el valor de tres variables de tipo entero, real y carácter que hayan sido definidas pero no inicializadas. Compilar y ejecutar el programa. Analiza la salida que produce y adiciona tus conclusiones al manual.
4. Escribir un programa que declare e inicialice dos variables de tipo cadena cad1 y cad2 y dos variables de tipo carácter c1 y c2, posteriormente imprimir el resultado de sumar cad1 + cad2 y c1 + c2. Analizar la salida que produce y adicionar sus conclusiones al manual.
5. Declara variables de tipo numérico, inicialízalas y posteriormente realiza las siguientes operaciones (Imprime el resultado):

$$b^2 - 4ac$$

$$x(y+z)$$

$$\frac{1}{x^2+x+3}$$

$$\frac{a+b}{c+d}$$

6. Calcular el sueldo de un operario conociendo la cantidad de horas que trabajó en el mes y el valor de la hora.
7. 1 milla tienen 1.609344 kilómetros. Un maratón tiene 26 millas y 385 yardas. Una milla tiene 1760 yardas. Calcula la distancia del maratón en kilómetros.