



UNIVERSIDAD PRIVADA DE TACNA

Tipos de Datos C#

Curso: Programación II

Ing. Enrique Lanchipa Valencia



Resumen



Diferencia entre Visual Basic – C#



Tipos de datos



Operadores



Declaración de sentencias



Bucles



Errores



Variables

En C#, se necesita una variable para cada dato que se quiera guardar en memoria.

Representan un dato de un tipo determinado, declarado dentro de un método y alojado en la memoria

- Contiene un valor que puede cambiar
- Se representa por un nombre
- Se destruye al finalizar el método

Directrices

- Deben de comenzar por una letra.
- No deben incluir espacios en blanco
- Distingue mayúsculas y minúsculas
- No se puede utilizar palabras reservadas

Declaraciones e inicialización de Variables

Tipo de dato

`int a;`

`a = 5;`

`int b;`

`b = 3;`

`Console.WriteLine("a vale: ");`

`Console.WriteLine(a);`

Nombre de Variable

Tipo de dato

`int a=5, b=3;`


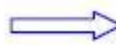


`Console.WriteLine("a vale: ");`

`Console.WriteLine(a);`

Nombre de Variables
e inicialización

Ámbito de una variable



```
public class UnaClase {  
    private int numero1;  Variable miembro o atributo de la clase  
    public void calcular() {  
        int a = 1;  Variable local del método  
        {  
            Console.WriteLine(a + ", " + numero1);  
            int b = 2;  Variable de bloque  
            Console.WriteLine(a + ", " + b);  
            {  
                int c = 3;  Variable de bloque  
                Console.WriteLine(a + ", " + b + ", " + c);  
            } // Fin del ámbito de c. Final del bloque donde se ha declarado  
            Console.WriteLine(a + ", " + b + ", " + c); // esta línea provoca un  
                                                    // error de compilación.  
                                                    // c esta fuera de su ámbito  
                                                    // y por tanto, no declarada  
        } //Fin del ámbito de b. Final del bloque donde se ha declarado  
    } // Fin del ámbito de a. Final del método calcular  
} //Fin del ámbito de número1. Final de la clase
```

Constantes

Una constante es una variable cuyo contenido no puede ser modificado una vez inicializado su valor.

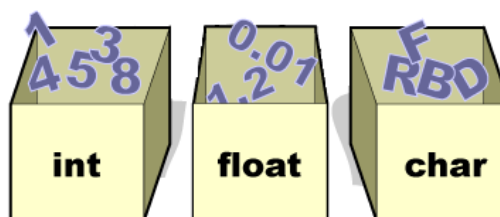
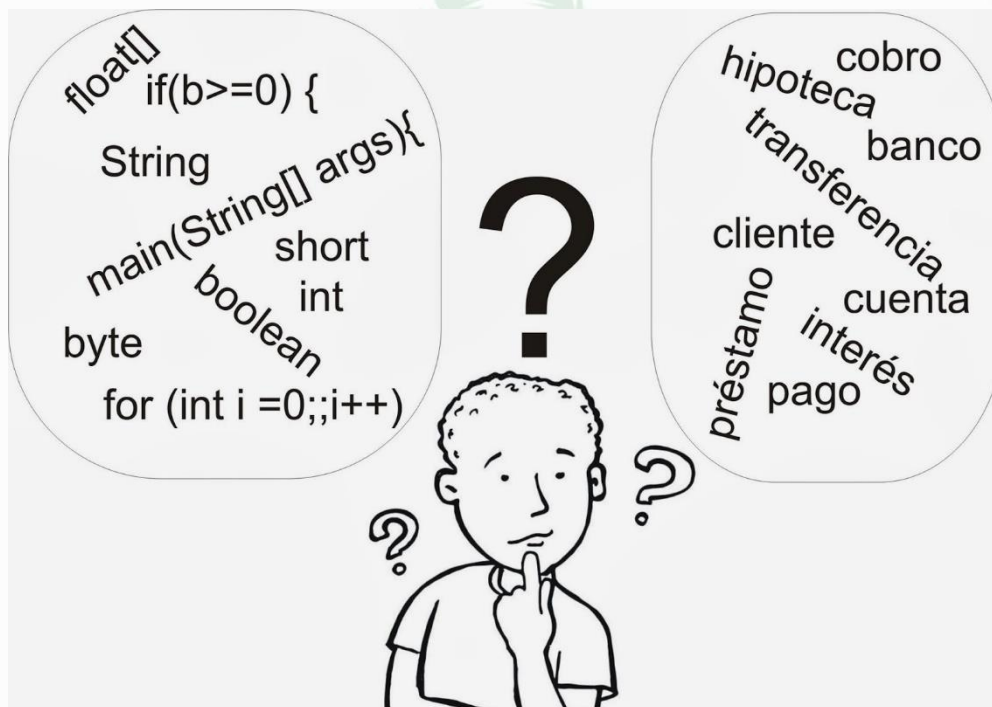
En Java se les llama variables finales

- declaración: atributo o variable con descriptor **const**
- **const** indica que el dato ya no se puede cambiar de valor
- constantes con valor "en blanco": se les puede asignar el valor una vez
- no es necesario definirlas como **private**, ya que nadie puede "estropearlas o modificarlas"

Ejemplos

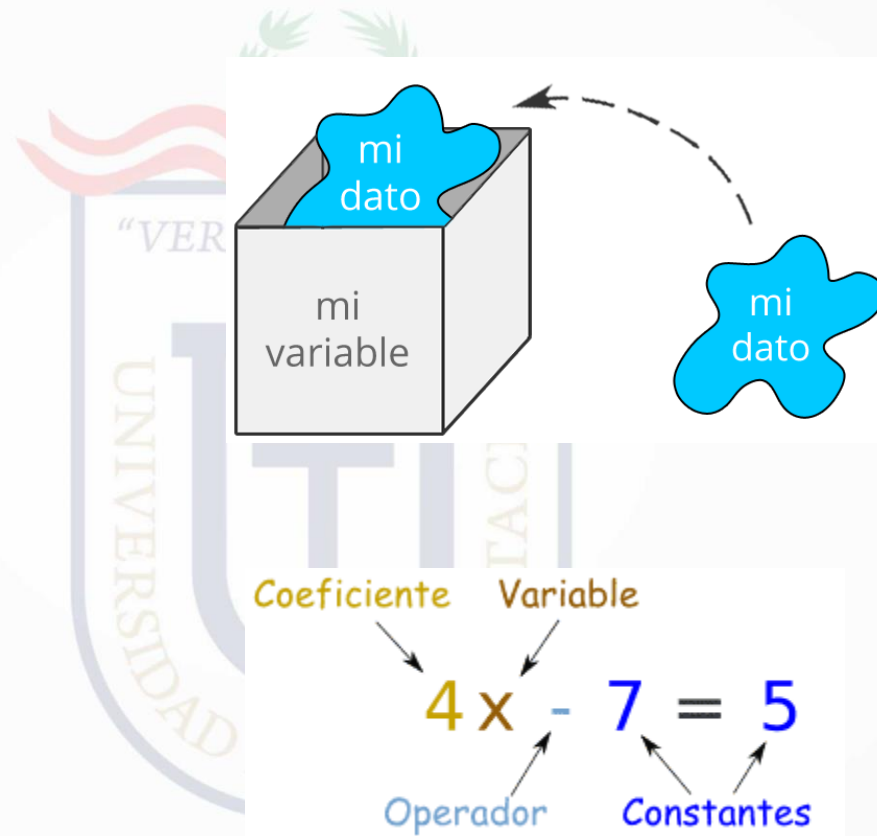
```
const double pi = 3.1416;  
const int maxNum = 50;  
const double factorEscala;
```

Tipo de dato



Tipo de dato

Determinan el valor que debe contener una variable o una constante y que operaciones se pueden desarrollar



Tipos de Datos

Entero (4 bytes)

Entero corto (1 byte)

Entero corto (2 bytes)

Entero largo (8 bytes)

Flotante (4 bytes)

Flotante (8 bytes)

Decimal

Fecha

Carácter Unicode

Cadena Unicode

Booleano

VB

Integer, UInteger

SByte, Byte

Short, Ushort

Long, ULong

Single

Double

Decimal

Date

Char

String

Boolean

C#

int, uint

sbyte, byte

short, ushort

long, ulong

float

double

decimal

DateTime

char

String

bool

Operadores



Asignación
Suma
Resta
Multiplicación
División
División entera
Residuo / Módulo
Potencia
Concatenación
Comparación
Mayor que
Menor que
Desigual
AND
OR
NOT

VB

=
+
-
*
/
\
Mod
^
&
=, Is (objetos)
>
<
<>, Is Not (objetos)
And
Or
Not

C#

=
+
-
*
/
/ (según tipo de variable)
%
No aplica
+
==
>
<
!=
&, &&
|, ||
!

Operadores



Arreglos
Dirección de memoria
Conversiones (Type cast)
Verdadero y Falso
Operador condicional

VB

()
Address Of
CInt, CDbl, CStr, CType
True, False
Iif(cond, true, false)

C#

[]
&
(type)
true, false
(cond) ? true : false

Declaración de variables y funciones

Visual Basic

Variables

```
Dim nombre as String  
Dim suma as Integer = 0
```

Funciones

```
Sub Guardar(x as Integer, y as Integer)  
    . . .  
End Sub  
  
Function Suma(x as Double, y as Double) as  
Double  
    . . .  
End Function
```

C#

Variables

```
String nombre;  
int suma = 0;
```

Funciones

```
void Guardar(int x, int y) {  
    . . .  
}  
  
double Suma(double x, double y) {  
    . . .  
}
```

String nombre = "Juan Carlos"; // variable de tipo String

Tipo de Dato

Nombre de la
Variable

Valor Inicial

Comentario

Declaración de sentencias condicionales

Visual Basic

If

```
If cond Then
    . . .
Else
    . . .
End If
```

Select case

```
Select case var
Case 0
    . . .
Case 1
    . . .
Case Else
    . . .
End Select
```

C#

If

```
If (cond) {
    . . .
} else {
    . . .
}
```

Switch

```
switch (var) {
case 0:
    . . .; break;
case 1:
    . . .; break;
default:
    . . .
}
```

Declaración de sentencias repetitivas

Visual Basic

For

```
For n = 1 To 10
```

```
    . . .
```

```
Next
```

```
For Each prop In obj
```

```
    . . .
```

```
Next prop
```

While

```
While n < 100
```

```
    . . .
```

```
End While
```

```
}
```

C#

For

```
for (int i = 1; i <= 10; i++) {
```

```
    . . .
```

```
}
```

```
foreach (int i in testArray) {
```

```
    . . .
```

```
}
```

While

```
while (int n < 100) {
```

```
    . . .
```

```
}
```

Declaración de Excepciones

Visual Basic

Error Handling

```
Try  
    . . .  
Catch err As System.Exception  
    . . .  
Finally  
    . . .  
End Try
```

C#

Error Handling

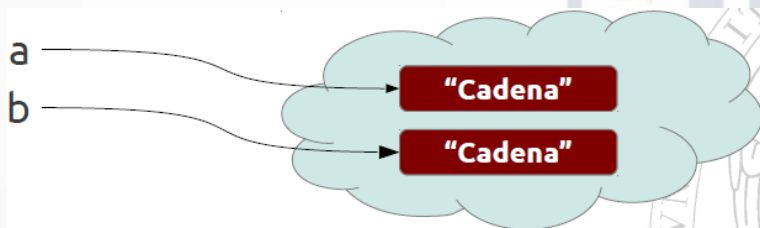
```
try {  
    . . .  
} catch (System.Exception err) {  
    . . .  
} finally {  
    . . .  
}
```

Igualdad con cadenas

- ¡Ojo con la igualdad entre cadenas!

```
String a = "Cadena";  
String b = "Cadena";  
if (a == b)  
System.out.println("Las cadenas son iguales");
```

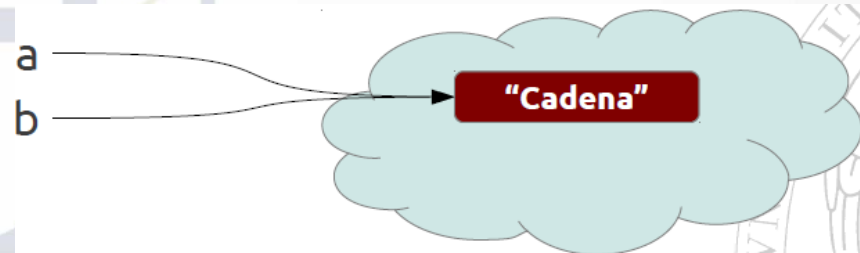
False



- ¡Ojo con la igualdad entre cadenas!

```
String a = "Cadena";  
String b = a;  
if (a == b)  
System.out.println("Las cadenas son iguales");
```

False



Igualdad con cadenas

¡Ojo con la
igualdad entre
cadenas!

```
String a = "Cadena";  
String b = "Cadena";
```

```
if (a.equals(b))
```

True

```
System.out.println("Las cadenas son iguales");
```

a

b

"Cadena"

"Cadena"

Separadores

() – paréntesis: Para contener listas de parámetros en la definición y llamada a métodos. También se utiliza para definir precedencia en expresiones, contener expresiones para control de flujo y rodear las conversiones de tipo.

{ } – llaves: Inicializar valores de matrices inicializadas automáticamente. También se utiliza para definir un bloque de código, para clases, métodos y ámbitos locales.

[] – corchetes: Declarar tipos matriz. También se utiliza cuando se referencian valores de matriz.

;- punto y coma: Separa sentencias.

, - coma: Separa identificadores consecutivos en una declaración de variables.

. – punto: Separa nombres de paquete de subpaquetes y clases.

También se utiliza para separar el receptor del selector del mensaje.

Estructura de un programa

- **public** : el método se puede usar desde fuera.
- **static** : el método pertenece a la clase (no a los objetos de la clase).
- **void** : no retorna nada
- **String[] args** : es el argumento, datos que se pasan a la operación
- **Console**: es una clase predefinida que representa al computador
- **WriteLine** : método para poner un texto en la pantalla

Nombre de la clase

De un nombre descriptivo a la clase principal.

Cuerpo del método

Incluya una secuencia de instrucciones.

```
public class [ ]
{
    public static void main( String[] args )
    {
        [ ]
    } //fin del método main
} //fin de la clase
```

Comentarios

Los comentarios en C# pueden hacerse de dos maneras:

1. Comentarios de una sola línea

`//Esto es un comentario`

2. Comentarios de varias líneas

`/* Esto es un
comentario
de varias líneas*/`

Caracteres especiales

`'\n'` Nueva línea

`'\t'` tabulación

`'\b'` retorno

`'\r'` salto de carro

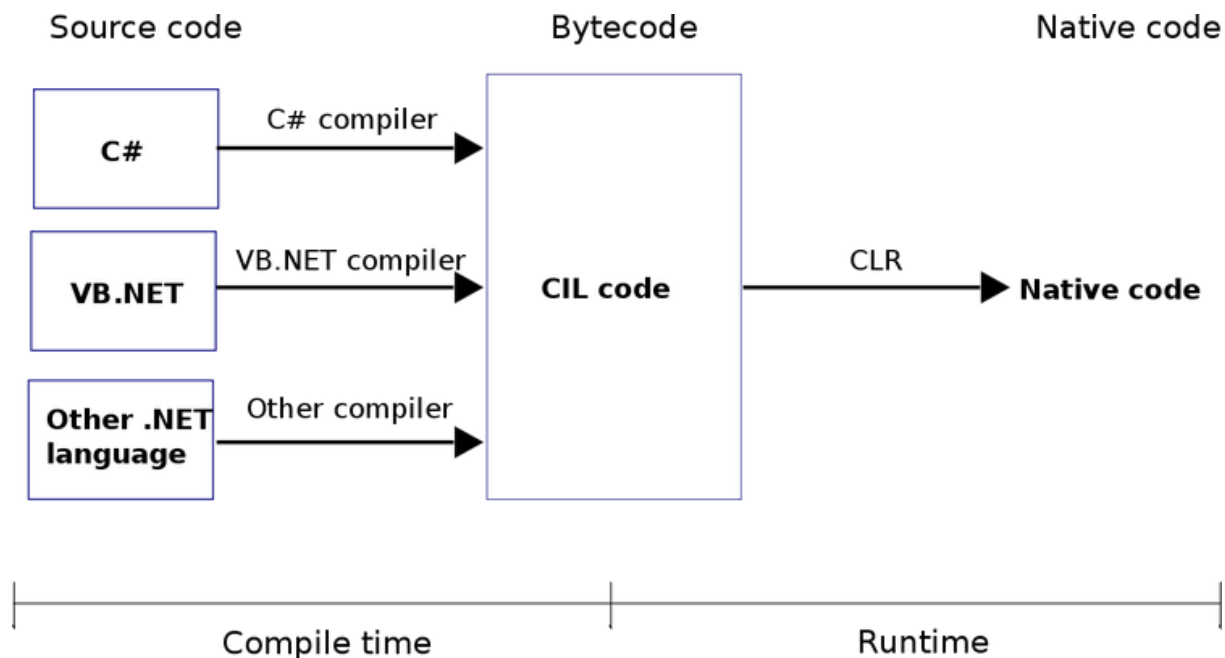
`'\f'` salto de hoja

`'\"'` comilla simple

`'\"'` doble comilla

`'\\'` barra invertida

Proceso de compilación



<https://sharplab.io/#v2:CYLg1APgAgzABFATHAwnA3gWAFBz3HfBeAYwHsA7AZwBcEBGABjgoEMBbAUzgF44AiAFIBXVhX4BuQvml5YeAJYU6nYK2C84MAKxTcM/XPhR6ANgbMA4pxoA5DpwAUASiyG5AdjgASfgAkyABtWDDYuAF9JAYJwvFkCd3kTcxMrG3suAEkKQKVuHgA+H38gkPQwzki9OOxwoA===>

Gracias

