ILERNA Online

Videotutoría Extra: aprendiendo a programar

Modulo 03A: Programación

Conversión implícita: La que se hace automáticamente,
sin necesidad de que el programador incluya ningún tipo de parámetro. Tener cuidado con no perder información

```
int a = 10;
double b = 9.2;

b = a;
```

Salida por pantalla: 10

Conversión explicita: Lo que se llama hacer un casting, o castear.

```
double x = 123.1;
int a = (int)x;
```

Salida por pantalla: 123

```
string frase = “Hola esta es la…”;
char Micaracter = frase[0]; //para coger la h
```

Console.writeLine(Micaracter); Mostraría por pantalla “H”

Método parse: para cadena de caracteres (string)

```
Console.WriteLine(“Usuario, introdice dos numeros”);

int numero = int.Parse(Console.Readline());
```

También convert (probar)

Tipos de datos simples

- ***Enum***: se utiliza para declarar una enumeración, un tipo distinto que consiste en un conjunto de constantes con nombre denominado lista de enumeradores.
- Cada tipo de enumeración tiene un tipo subyacente, que puede ser cualquier tipo numérico entero. El tipo char no puede ser un tipo subyacente de una enumeración. El tipo subyacente predeterminado de los elementos de la enumeración es int.

```
class Program
{
    1 referencia
    enum dias { lunes, martes, miércoles, jueves, viernes, sabado, domingo};
    0 referencias
    static void Main(string[] args)
    {

        int x = (int)dias.lunes;
        Console.WriteLine(x);
        Console.ReadKey();
    }
}
```

ILERNA Online

Tipos de datos compuestos

- ***Vectores:*** se utilizan para agrupar distintas variables de un mismo tipo con un nombre único.

```
class Program
{
    0 referencias
    enum dias { lunes, martes, miércoles, jueves, viernes, sabado, domingo};
    0 referencias
    static void Main(string[] args)
    {
        | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
        int[] v = new int[10];{10, 100, 1000, …}; si conozco los datos que quiero guardar
        Console.WriteLine("introduce datos");

        for (int i = 0; i < 10; i++) {

            v[i] = int.Parse(Console.ReadLine());

        }

        Console.ReadKey();
    }
}
```

ILERNA Online

Tipos de datos compuestos

- ***Matrices:*** un *array* bidimensional, por tanto, esos datos nos indican que necesitamos dos indicadores de posición para acceder al elemento. La primera posición de un array es el 0.

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

    int[,] matriz = new int [2,4];
    Console.WriteLine("introduce datos");

    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 3; j++) {

            matriz[i, j] = int.Parse(Console.ReadLine());
            char[  ,  ] M = new char [0, 0]

        }   | a | b | c |
            | d | e | f |
            | g | h | i |  con [0, 0] accederíamos a la “a”.
    }       [3, 3] estaría fuera de rango.

    Console.ReadKey();
}
```

ILERNA Online

Estructuras de programación: IF/WHILE

if (a == 0). Si pongo a = 0 estaría intentando comparar una condición booleana a un número entero. Evitar bucles infinitos.

```
        Console.WriteLine("Si");
else
        Console.WriteLine("No");
else
        Console.WriteLine("Tal vez");
```

```
SI CONDICIÓN ENTONCES:
    INSTRUCCIONES_1;
SI NO:
    INSTRUCCIONES_2;
FIN SI;
```

- **Mientras *(While)*** Tal vez no se ejecute ni una sola vez

Mientras que se cumpla una condición, el código incluido dentro del bucle se repite. La condición se evalúa al principio, por lo que puede que no llegue a ejecutarse nunca.


```
Mientras CONDICIÓN Hacer
    Instrucción_1;
    Instrucción_2;
    ...;
    Instrucción_N;
    ModificarCondición;
FinMientras;
```

- **Hacer... mientras *(Do... while)*** Se ejecuta mínimo una vez

Mientras que se cumpla una condición, el código incluido dentro del bucle se repite. La condición se evalúa al final, por lo que, como mínimo, se va a ejecutar una vez.


```
Hacer
    Instrucción_1;
    Instrucción_2;
    ...
    Instrucción_N;
    ModificarCondición;
Mientras CONDICIÓN;
```

FOR: necesitamos un índice que se declara dentro de for : for (int i = 0; i < 10; i ++) //1ª condición establecer la variable, 2ª establecer condición de salida. También puede ser for (int i = 10; i > 0; i--). El primer caso para recorrer un vector p.e.

ILERNA Online

Estructuras de programación: Switch

Normalmente se utiliza para realizar un menú que se pregunta al usuario.

```
static void Main(string[] args)
{
    int var = 1;
    switch (var)
    {
        case 1:
            Console.WriteLine("Caso 1");
            break;
        case 2:
            Console.WriteLine("Caso 2");
            break;
        default:
            Console.WriteLine("Otro caso");
            break;
    }
    Console.ReadLine();
}
```

Se ejecutarían las instrucciones del caso 1 por ser var = 1

Si var != que 1 y que 2 salta “default”. Si no estuviese el default no saltaría nada en pantalla.

ILERNA Online

Strings

- ***String:*** Representa una secuencia de cero o más caracteres

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

    string cadena = "Esta es mi cadena";

    char caracter = cadena[0];

    Console.WriteLine(caracter);
    Console.ReadKey();
}
```

Break

```
static void Main(string[] args)
{
    for (int i = 1; i <= 100; i++)
    {
        if (i == 5)
        {
            break;
        }
        Console.WriteLine(i);
    }

    // Keep the console open in debug mode.
    Console.WriteLine("Press any key to exit.");
    Console.ReadKey();
}
```

Índice de entrada, de salida, y como vamos a incrementar.

Cuando i = 5 break rompe el bucle de for y nos salimos, mostrando los números del 1 al 4.

```
int var = 0;
int[] array = new int[3] { 1, 2, 3}

for (int i=0; i<=3; i++
{

    array[i] = i;

}


Console.ReadKey();
```

Daría error ya que si i llega a valer 3, hablaríamos de 4 valores, cuando se han declarado 3 y se empieza a contar desde el 0 como primer índice del array.

Daría un error lógico por intentar acceder a más posiciones de las que tiene mi array.

De hacerlo bien estaría asignando un valor a cada elemento del vector.

Continue

La instrucción continue transfiere el control a la siguiente iteración de la instrucción envolvente while, for, do en la que aparece.

Estos bucles suele preguntar en el examen.

```
0 referencias
static void Main(string[] args)
{

    for (int i = 1; i <= 10; i++)
    {
        if (i < 9)
        {
            continue;
        }
        Console.WriteLine(i);
    }

    // Keep the console open in debug mode.
    Console.WriteLine("Press any key to exit.");
    Console.ReadKey();
}
```

En este caso el for. Imprimiría 9 y 10

ILERNA Online

Return

La instrucción return termina la ejecución del método en el que aparece y devuelve el control al método de llamada

```
1 referencia
static double CalculateArea(int r)
{
    double area = r * r * Math.PI;
    return area;
}

0 referencias
static void Main()
{
    int radius = 5;
    double result = CalculateArea(radius);
    Console.WriteLine("The area is {0:0.00}", result);

    // Keep the console open in debug mode.
    Console.WriteLine("Press any key to exit.");
    Console.ReadKey();
}
```

Esta sentencia esta obsoleta, no utilizar. Esta prohibidísimo utilizarlo. Porque en un programa de 10.000 líneas seria un caos. Decirle cada vez que vaya a determinada línea.

GOTO

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

    onsole.WriteLine("Coffee sizes: 1=Small 2=Medium 3=La  ");
    ole.Write("Please enter your selection: ");
    s = Console.ReadLine();
    nt.Parse(s);
    0;
    switch (n)
    {
        case 1:
            cost += 25;
            break;
        case 2:
            cost += 25;
            goto case 1;
        case 3:
            cost += 50
            goto cas  ;
        default:
            Con  .WriteLine("Invalid selec  .");
            b  k;
    }
    if (  t != 0)
    {
        Console.WriteLine($"Please insert {cost} cents.");
    }
    Console.WriteLine("Thank you for your business.");
```

Programación estructurada: ciclo de vida

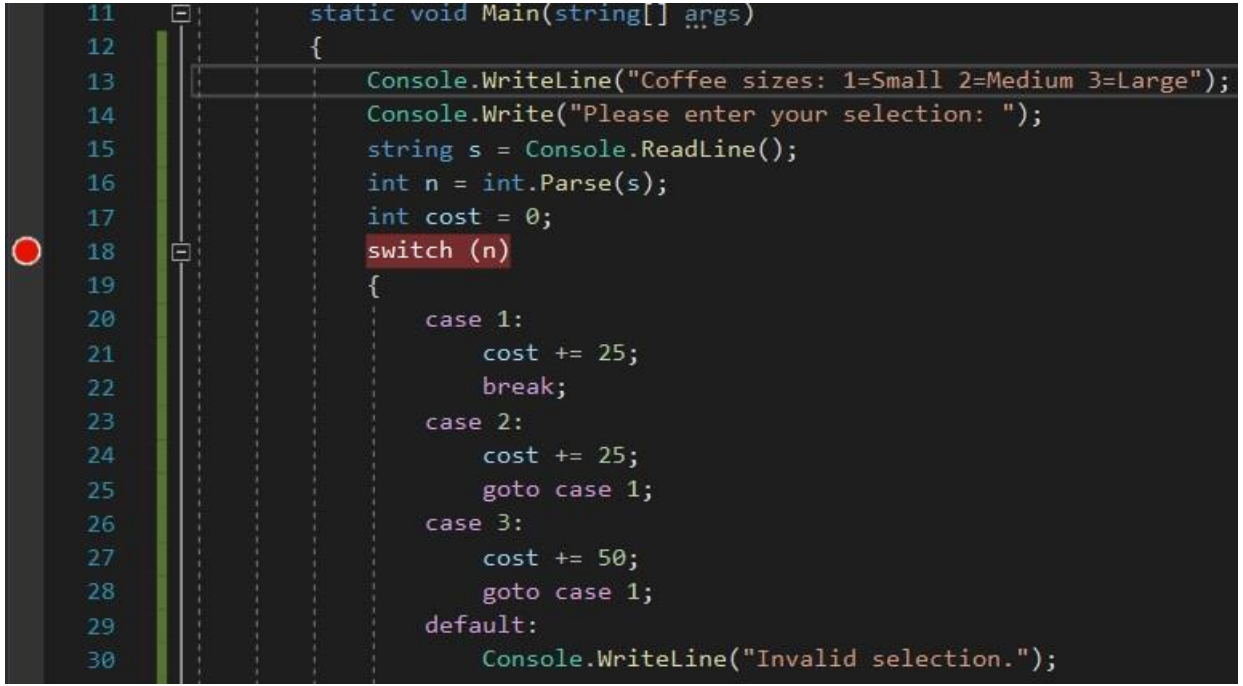
FASES DEL DESARROLLO DEL SOFTWARE

Programación estructurada

Muy importante en el mundo de la programación

- ***Edsger Dijkstra*** comprobó que todo programa se puede escribir utilizando únicamente tres instrucciones de control:
    - Secuencia de instrucciones.
    - Instrucción condicional. (if/If…else)
    - Iteración o bucle de instrucciones. (While/Do…While/for)

Puntos de interrupción: F5

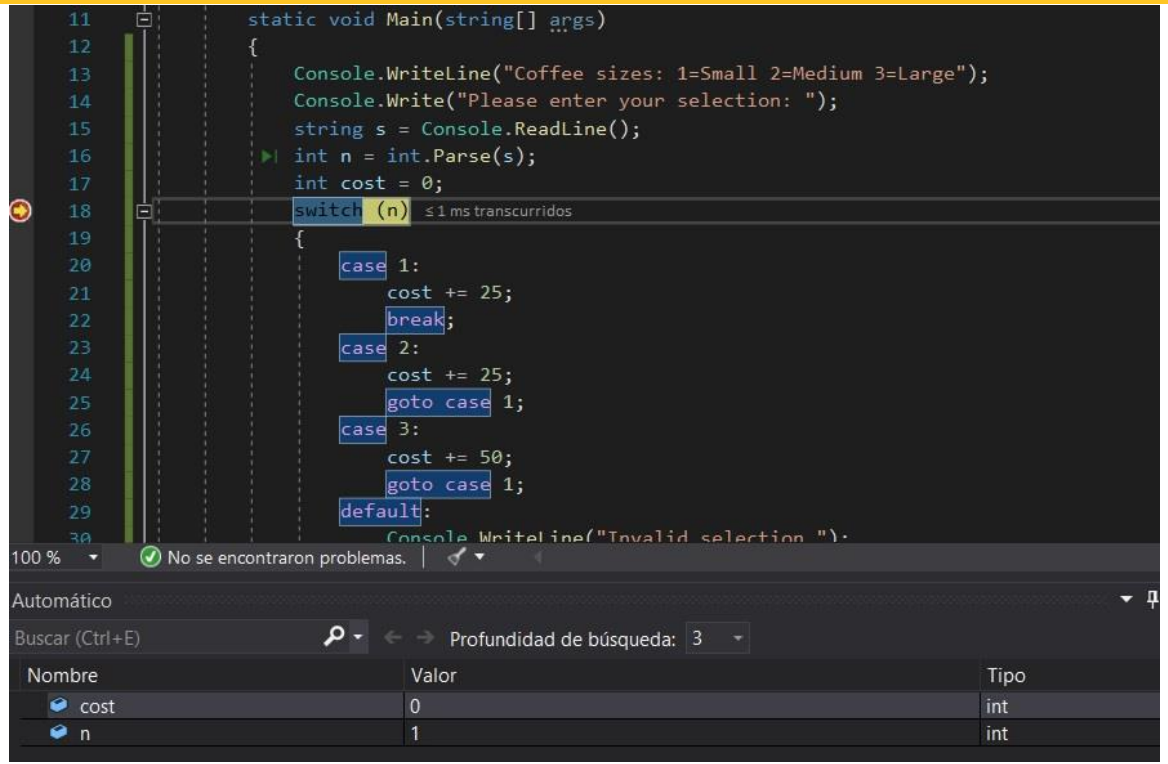


```
11      static void Main(string[] args)
12      {
13          Console.WriteLine("Coffee sizes: 1=Small 2=Medium 3=Large");
14          Console.Write("Please enter your selection: ");
15          string s = Console.ReadLine();
16          int n = int.Parse(s);
17          int cost = 0;
18          switch (n)
19          {
20              case 1:
21                  cost += 25;
22                  break;
23              case 2:
24                  cost += 25;
25                  goto case 1;
26              case 3:
27                  cost += 50;
28                  goto case 1;
29              default:
30                  Console.WriteLine("Invalid selection.");
```

ILERNA Online

Hay que darle a iniciar, se parará donde hayamos puesto el punto de interrupción, e ir dándole para que avance.

Debuguear paso a paso: F10



ERRORES

- ***Errores de compilación***: estos errores impiden la ejecución de un programa. Mediante F5 ejecutamos un programa. Inicialmente se compila el programa, y, si el compilador de Visual Studio encuentra cualquier cosa que no entiende, lanza un error de compilación.

- ***Errores en tiempo de ejecución***: aparecen mientras se ejecuta el programa, normalmente cuando se pretende realizar una operación que no lleva a ninguna solución, como, por ejemplo, cuando se pretende dividir por cero.

- ***Errores lógicos***: impiden que se lleve a cabo lo que se había previsto. El código se puede compilar y ejecutar sin problema, pero, en el caso de que se compile, devuelve algo que no era la solución que se esperaba. El programa no da error cuando se ejecuta en el inicio, por lo que su corrección es más difícil.

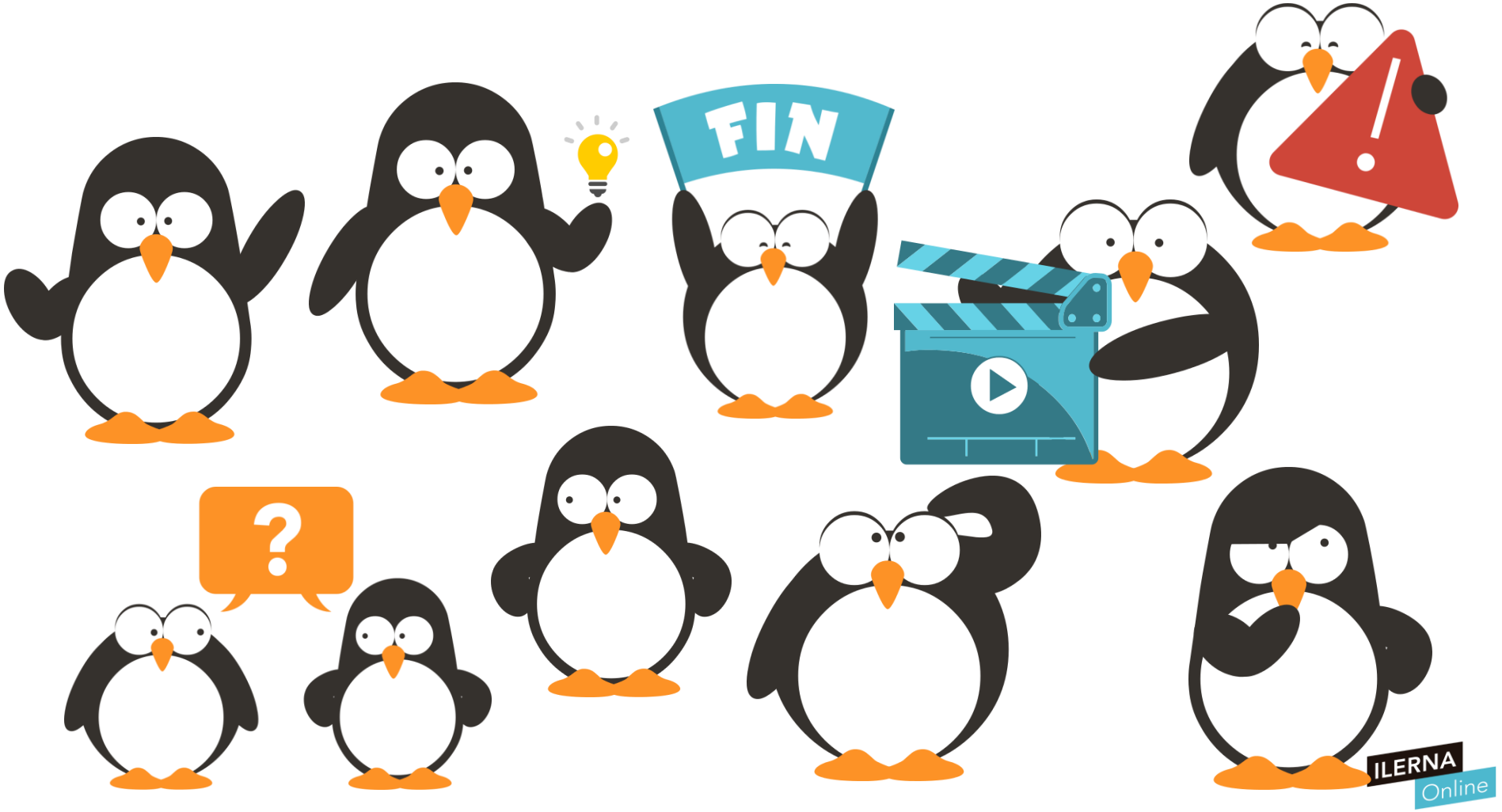
EJERCICIOS

- Realiza un programa que pida un número, y calcule el perímetro de la circunferencia y el área del círculo con ese radio. Ten en cuenta que el área de un círculo es a = PI * r * r, y el perímetro de la circunferencia es p = 2 * PI * r.

- Realiza un programa que calcule el coste del precio de la gasolina en un viaje. El usuario introducirá los kilómetros a recorrer, el precio de la gasolina y el consumo del coche en litros por cada 100 kilómetros.

- Cree un programa que realiza la transformación de una cantidad de segundos en horas, minutos y segundos. La cantidad de segundos se introduce por teclado.

- Realiza un programa que muestra los resultados de una encuesta en porcentajes. En la encuesta se puede contestar SI, NO o NO SABE-NO CONTESTA. Por teclado únicamente se introducirá el número de respuestas de cada categoría.


FIN
?
ILERNA
Online