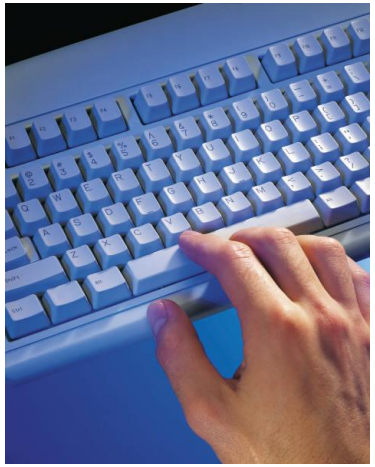
Lenguaje C#

Estructura de un programa en C#

Adriana Natalia Ramírez Salazar

Declaración de entrada/salida

- Las objetos principales de entrada y salida de C#, son:
  - WriteLine
  - ReadLine



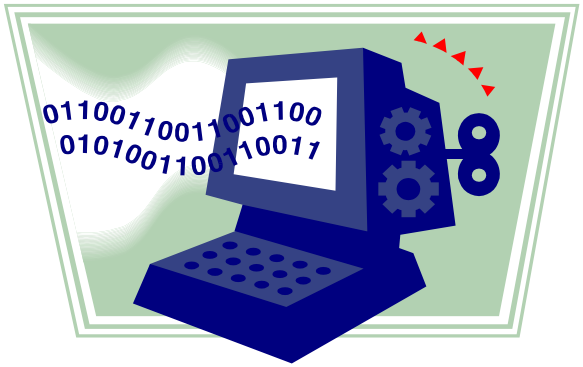
Declaración de entrada/salida

- **Ejemplo:**
  `Console.WriteLine("Hola mundo \n soy Natalia");`

- **Programa:**

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

namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hola Mundo \n soy Natalia");
        }
    }
}
```



Estructura de un programa en C#

- **Tipos de datos en C#:** Los tipos de datos básicos en C# son: **carácter**, **entero**, **real**, **real de doble precisión** y **sin valor**.

Tipo de dato	Declaración	Ejemplo
carácter	char	c, a, R, S, etc.
entero	int	34, 1024, etc.
real	float	5.12, 3.1416, 0.45678, etc.
real de doble precisión	double	20147483647. 8976766
sin valor	void	
cadena	string	Hola
boleano	bool	true, false

Estructura de un programa en C#

- A continuación se muestra la tabla de todos los tipos de datos en C#:

TIPO	ANCHO EN BIT	RANGO EN PC
char	8	-128 a 127
unsigned char	8	0 a 255
signed char	8	-128 a 127
int	16	-32768 a 32767
unsigned int	16	0 a 65535
signed int	16	-32768 a 32767
short int	16	-32768 a 32767
unsigned short int	16	0 a 65535
signed short int	16	-32768 a 32767
long int	32	-2147483648 a 2147483647
signed long int	32	-2147483648 a 2147483647
unsigned long int	32	0 a 4294967295
float	32	3.4E-38 a 3.4E+38
double	64	1.7E-308 a 1.7E+308
long double	64	1.7E-308 a 1.7E+308

Uso de identificadores. Palabras reservadas

- Las **palabras clave** son identificadores predefinidos reservados que tienen un significado especial para el compilador, por lo tanto, no se pueden utilizar para definir nombres de variables.
- **Ejercicio:**
- Crear un programa que muestre las palabras reservadas en C#.

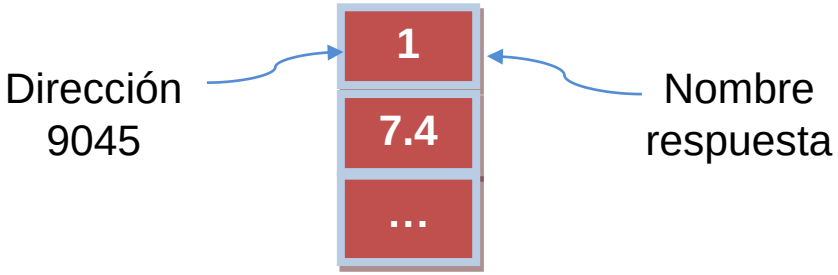
Declaración de entrada/salida

- **Caracteres especiales**

Código	Significado
\b	Espacio atrás
\f	Salto de página
\n	Salto de línea
\r	Retorno de carro
\t	Tabulador
\”	Comillas
\’	Apóstrofo
\0	Nulo
\\	Barra invertida
\v	Tabulador vertical
\a	Alerta (sonido)

Uso de variables y constantes

- **Variable:** Es una posición de memoria con un nombre donde se almacena un valor de un cierto tipo de dato.



- Las variables pueden almacenar todo tipo de datos:
  - cadenas, números, arreglos, estructuras, etc.

Uso de variables y constantes

- Las **constantes** son **variables** cuyo valor no puede ser modificado.

- Una variable tiene:
  - Dirección
  - Nombre (identificador): describe su propósito

Uso de variables y constantes

- **Definición:** Para definir una variable es necesario poner: el tipo de dato, el identificador o nombre de la variable y opcionalmente el valor inicial que tomará.
  - *tipo variable*

- *tipo* es el tipo de dato
- *variable* es un identificador o nombre válido en C#

Uso de variables y constantes

- En base a su ámbito o alcance las variables, pueden ser:
  - Variables locales
  - Variables globales

- **Variables locales:**
  Son aquellas definidas en el interior de una función y son visibles solo en esa función específica.

Uso de variables y constantes

- **Variables globales**

  Son variables que se declaran fuera de la función y son visibles en cualquier función, incluyendo a la función `main()`.

- Las variables locales desaparecen cuando se termina su bloque, en cambio, las variables globales son visibles desde el punto en que se definen hasta el final del programa o archivo.

Uso de variables y constantes

- **Ejemplo :**
- **Entrada del teclado ReadLine**

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

namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            int n1, n2, r; //Variables locales
            Console.WriteLine("Introducir 2 números");
            //Forma 1 ReadLine
            n1 = int.Parse(Console.ReadLine());
            n2 = int.Parse(Console.ReadLine());

            r= n1+n2;
            //Forma 1 WriteLine:
            Console.WriteLine("La suma de " + n1 + " + " + n2 + " es: " + r )
         }
    }
}
```

Uso de variables y constantes

- **Ejemplo:**
- **Entrada del teclado ReadLine**

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

namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            int n1, n2, r;
            Console.WriteLine("Introducir 2 números");
            //Forma 1 ReadLine
            //n1 = int.Parse(Console.ReadLine());
            //n2 = int.Parse(Console.ReadLine());
            //Forma 2 ReadLine
            n1 = Convert.ToInt32(Console.ReadLine());
            n2 = Convert.ToInt32(Console.ReadLine());
            r= n1+n2;

            //Forma 1 WriteLine:
            //Console.WriteLine("La suma de " + n1 + " + " + n2 + " es: " + r );
            //Forma 2 WriteLine:
            Console.WriteLine("La suma de {0} + {1} es: {2}", n1, n2, r);
        }
    }
}
```

Si recibe un null lo convierte a cero

Uso de variables y constantes

- **Ejercicios:**
  - **Programa que convierta una cantidad de dinero en billetes**
  - **Programa que convierta una cantidad de segundos en horas, minutos y segundos**

Uso de variables y constantes

- **Constantes:** Son valores fijos que no pueden ser modificados por el programa.

- **Ejemplo:** `const int months = 12;`
- **Ejercicio:** realizar un programa que diga cuantos TB, GB, MB, KB, bytes tiene una cantidad de bytes, utilizar constantes para el número de TB, GB, MB, KB.
- **Ejemplo:** `const int KB = 1024;`

Declaración de entrada/salida

- **Captar string**
- **Ejemplo:**

```
class Program
    {
        static void Main(string[] args)
        {
            string nombre, apellido;
            float sueldo;
            Console.WriteLine("Introduce tu nombre y tu apellido");
            nombre = Console.ReadLine();
            apellido = Console.ReadLine();
            Console.WriteLine("Introduce tu sueldo: ");
            sueldo = float.Parse(Console.ReadLine());

            Console.WriteLine("Tu nombre y apellido es: " + nombre +" "+apellido+
                              "\nSueldo: " + sueldo);
            //Console.WriteLine("Tu nombre y apellido es: {0} {1} \nSueldo: {2}",
                               nombre,apellido, sueldo);
        }
    }
```

Declaración de entrada/salida (Formatos)

- **Ejemplo:**

```
class Program
    {
        static void Main(string[] args)
        {
            string nombre, apellido;
            float sueldo;
            Console.WriteLine("Introduce tu nombre y tu apellido");
            nombre = Console.ReadLine();
            apellido = Console.ReadLine();
            Console.WriteLine("Introduce tu sueldo: ");
            sueldo = float.Parse(Console.ReadLine());

            //Deja espacio de 15, separador de miles y cero decimales
            Console.WriteLine("Tu nombre y apellido es: {0} {1} \nSueldo:
                {2,15:N0}", nombre, apellido, sueldo);
        }
    }
```

https://msdn.microsoft.com/es-es/library/dwhawy9k(v=vs.110).aspx

Declaración de entrada/salida (Formatos)

- **Ejercicio:**
  - **Programa que pida los datos de una factura: código, descripción, cantidad y precio unitario de 2 productos y genere una factura. Usar formatos numéricos.**