Aprende programación fácilmente con ejercicios resueltos.

Maties Salom Avellà

Presentación:

Este libro pretende ser un complemento para otros libros de programación, más completos y con más fundamentos teóricos, pero que a veces necesitan ejercicios actualizados para asimilar completamente la teoría.

Es un libro de ejercicios indicado para una primera aproximación a la programación, pudiendo formar parte de la asignatura de “Fundamentos de la programación” o como apoyo para una formación autodidacta.

Está en preparación otro libro como una continuación de estos ejercicios, completando la temática de la programación Visual y la programación Web.

Para cualquier consulta con el autor, pueden hacérsela llegar mediante la web de www.ibserveis.com, en el apartado de *Contacto.*

Capítulo 0

Iniciación a la programación

Capítulo 0

Iniciación a la resolución de problemas estructuradamente con PSEUDOCODIGO

Es de gran importancia, antes de empezar a teclear con el ordenador, plantear correctamente la solución de nuestro proyecto.
Existen infinitas soluciones para cada uno de los problemas que pueden plantearse, pero siempre podemos encontrar la solución más adecuada, por ser la más práctica y fácil de implementar.

La filosofía de ***"la solución más simple, es probablemente la más acertada"*** (Navaja de Occam, wiki: Occam) y su actualización para proyectos informáticos: la filosofía KISS (Keep it Simple) , serán las filosofías que se considerarán prioritariamente para resolver estos ejercicios.

Como ejemplo anecdótico (totalmente falso , pero ilustrativo):
Se comenta que la NASA invirtió muchos millones en un bolígrafo capaz de escribir boca arriba y en condiciones extremas de temperaturas, la URSS hizo servir lápices .
(La historia real en google: sondas espaciales boli)

Un buen artículo respecto a mantener la simplicidad en el software (google: soitu keep simple) http://www.soitu.es/soitu/2008/03/14/pieldigital/1205521516_335354.html

Y si alguien aún duda que la simplicidad es bella, eficiente e indicada para la realización de proyectos, puede aplicar que el "tiempo es oro" si se tiene que desarrollar un proyecto para una empresa, con un presupuesto y tiempo limitado.

Estos ejercicios pueden ser ejecutados en un entorno de programación actual (2009), i gratuito: Microsoft Visual C# Express edition.

Por otra parte, para todos aquellos que no tienen intención de seguir el ritmo de las novedades informáticas, puede visitar esta web:
http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=devesb

google: adictos al trabajo tutoriales c#

y , para terminar, un comic con una situación muy real:
http://www.adictosaltrabajo.com/detalle-noticia.php?noticia=49

INTENTAR SOLUCIONAR LOS EJERCICIOS ANTES DE VER LA SOLUCIÓN PROPUESTA.

Breve explicación de los enunciados del Capítulo 0:

Lo más importante de estos primeros ejercicios es la "ordenación de instrucciones". Existen muchas soluciones para cada enunciado. Se propone una, lo más ordenada y estructurada posible para que los programas realizados más adelante también sean estructurados.

0.1) Pasos a seguir para conseguir cambiar una cuerda de guitarra

Para cambiar una cuerda de guitarra se hacen una serie de movimientos en el clavijero para conseguir la tensión necesaria en cada cuerda, estos pasos son los que describe la solución.

0.2) Resolución del algoritmo de compra de una camisa

En el día a día realizamos muchas acciones que podrían formar parte de un programa, es como si tuviéramos que dar las instrucciones, justas y necesarias, a un robot para escoger ropa.

Pensar en que instrucciones, paso a paso, seguimos para escoger ropa en una tienda.

0.3) PSEUDOCODIGO que calcula la suma de los números enteros entre 1 y 10

(1+2+3+4....)

Este es el primer ejercicio "matemático", las instrucciones para resolver el problema son mas abstractas que los ejercicios anteriores, pero siquen teniendo una dificultat inicial baja.

¿Qué instrucciones daríamos a un niño de primaria para que sume los 10 primeros números?

0.4) Pseudocódigo para calcular el producto de números enteros del 10 al 20

*(10*12*14....)*

El mismo tipo de ejercicio anterior, pero con otra operación matemática.

0.5) Código donde se introduce una serie de números y el ordenador cuenta cuantos son positivos.

La novedad de este ejercicio consiste en la interacción del usuario con el ordenador, el número que introduce el usuario forma parte de las instrucciones del algoritmo. Simplemente una variable asignada desde el teclado.

0.6) Algoritmo con introducción de 2 números y el ordenador indica cual es el mayor

Ejercicio con una instrucción condicional doble. El ordenador sigue una alternativa u otra.

0.7) Algoritmo con introducción de 2 números y el ordenador indica cual es mayor o si son iguales.

Ejercicio con instrucciones condicionales anidadas, ejemplo de programa estructurado.

0.8) Escribir la nota relacionada con la nota numérica: 5, suficiente, 6 bien, 7 notable...

Un ejercicio muy parecido al ejercicio 1.7, pero con la facilidad de una temática conocida por los estudiantes. Ver los resultados de sus esfuerzos reflejados en una nota trimestral.

Importante, como en todos los ejercicios, la estructuración y la simplicidad de la solución.

0.9) PSEUDOCODIGO para contar los alumnos que no llegan al 1.20 m , los que estan entre 1.20 y 1.50, entre 1.50 y 1.60 y los que tienen más de 1.60m de altura.

Un ejercicio bastante completo y estructurado, donde podemos trabajar en un algoritmo más serio y cercano a un programa informático. La solución es estructurada: existen otras soluciones, pero implican más "trabajo" (de comparación) para el ordenador cuando se realice el programa.

0.10) Reparación de una fuente de agua que no funciona.

En todas las reparaciones, sean de ordenadores o de otras máquinas se sigue un algoritmo para conseguir que la reparación sea lo más rápida y eficiente posible. En el ejercicio es una fuente, pero podría ser perfectamente cualquier máquina, motor, o aparato electrónico.

0.11) Pseudocódigo que calcula nota media de 3 asignaturas de cada uno de los 5 alumnos de clase.

Un ejercicio donde la calve está en un buen planteamiento para un sólo alumno, después ampliar la solución per los 5 alumnos que explica el enunciado

0.12) Algoritmo simplificado (20 líneas de código) de un Cajero automático

Este ejercicio podría ser muy largo o complicarse más, pero lo que importa en una estructuración inicial simple para poder ampliarlo, si se desea, más adelante.

Se pueden obtener tantas soluciones como personas resuelvan este ejercicio.

0.1) Pasos a seguir para conseguir cambiar una cuerda de guitarra

```
Inicio
    Preparar material necesario
    Extraer cuerda rota
    Introducir cuerda nueva
    Mientras no tenga  tensión
        Girar clavijero
    volver

    ¿Esta afinada?
     NO :  Mientras no esté  afinada
            ¿Está en un tono más alto?
                Sí : Girar clavijero a la izquierda
                No: Girar clavijero a la derecha
           volver
     Sí : esta afinada
Fin
```

0.2) Resolución del algoritmo de compra de una camisa

```
Inicio
    Entrar a la Tienda
    Mientras no vea nada que me gusta
        Buscar
    volver

    Mirar precio
    ¿Es un precio interesante?
      SÍ:  ¿Es de mi  talla?
            SÍ:  Voy al probador
                 ¿Me gusta cómo me queda?
                    SÍ: Compro la ropa
                    NO: volver a buscar

            NO: volver a buscar

     NO: volver a buscar
Fin
```

0.3) PSEUDOCODIGO que calcula la suma de los números enteros entre 1 y 10 (1+2+3+4....)

```
inicio
        suma ← 0: conta ← 0

        Mientras conta menor que 10
                conta ← conta + 1
                suma ← suma + conta
        Fin Mientras

        Escribir "La suma es"; suma
fin
```

0.4) Pseudocodigo para calcular el producto de números enteros del 10 al 20 10*12*14...

```
inicio
        suma ← 0: conta ← 10: producto ← 1

        Mientras conta menor o igual que 20
                producto ← producto * conta
                conta ← conta + 2
        Fin Mientras

        Escribir "El producto es"; producto
fin
```

0.5) Código donde se introduce una serie de nº y el ordenador cuenta cuantos son positivos.

```
inicio
        positivos ←0: conta ←0
        Introducir "Introduce número";numero

        Mientras numero distinto 999
                conta ← conta + 1
                SI numero >0
                        positivos ← positivos +1

                Introducir "Introducir número";numero
        Fin Mientras

        Escribir "Has introducido un total de "; conta
        Escribir "I son positivos "; positivos
fin
```

Capítulo 1

Programas ejecutables en Basic y C#

Capítulo 1: Programas ejecutables en Basic y C#

Breve explicación de los enunciados del Capítulo 1:

Hemos aprendido como plantear un algoritmo para la resolución de un problema, ahora necesitamos aprender el "idioma" del ordenador para que ejecute la solución diseñada.

Para que el ordenador entienda nuestras instrucciones, necesitaremos un entorno de trabajo (IDE) que nos permita escribir y corregir código, así como también ejecutarlo.

Por su proximidad al pseudocódigo y su facilidad de aprendizaje, se han resuelto unos cuantos ejercicios en lenguaje Basic . El entorno de trabajo escogido es el del **SmallBasic**, inicialmente destinado a máquinas portátiles y móviles, tiene versiones para muchos sistemas operativos: smallbasic (de fácil búsqueda en Google)

- BASIC: http://smallbasic.sourceforge.net/ , versión: smallbasic 0_9_7_ftlk

No es el objetivo de este libro enseñar a manejar entornos de programación. Existen tutoriales muy buenos en internet para empezar a programar en Microsoft Visual C#:

Búsquedas en Google: ***Tutorial c#***

www.devjoker.com : Tutorial C#

http://functionx.com/csharp/Lesson01.htm : FunctionX.com

Libro gratuito recomendado: www.josanguapo.com

1.1) 1.2) Comprobar con la calculadora del S.O. 4 dígitos binarios son 1 digito hexadecimal.
La relación decimal-hexadecimal-binario puede haber perdido importancia en lenguajes de alto nivel, pero ha sido la base matemática de muchas operaciones en lenguaje C.

1.3) Programa que calcula la suma de los números enteros entre 1 y 10 *(1+2+3+4....)*
1.4) Programa producto números enteros del 10 al 20 *(10*12*14...)*
Los ejercicios 1.4 - 1.5 tienen fácil traslación al lenguaje de programación BASIC.

1.5 - 1.6) Programa: usuario introduce números y el ordenador cuenta cuantos son positivos.
Ya podemos comparar ambos lenguajes BASIC- C# , el mismo pseudocódigo escrito en cada uno de estos lenguajes**.** Nuestra "Piedra de Rosetta" particular *(google: piedra rosetta)*

1.7) Contabiliza personas de más de 180, entre 180 y 170, 170 y 160, y menores de 160cm.
El mismo pseudocódigo que el ejercicio 1.7 para comprobar que si hemos pensado correctamente la solución, es fácil implementarla con un lenguaje de programación

1.8) Programa para poner notas: suspendido, aprobado, bien... con la nota numérica.
Solución del ejercicio 0.8 con lenguaje C#

1.9) Introducir números y el ordenador enseña el máximo y cuantas veces se ha repetido.
Ejercicio para insistir y asimilar los conceptos de los ejercicios 2.3 y 2.4

1.10) Calcula nota media de 3 asignaturas de cada uno de los 5 alumnos de clase.
Solución en BASIC y C# del ejercicio 1.8

1.11) Programa para calcular salario semanal si el precio/hora es 10 , hora extra a 15
Un inicio de aplicación práctica, y muy típica, de la programación: cálculo de nominas.

1.12) Programa en el cual se introducen 3 números, el ordenador señala cual es el central.
Es importante pensar en una solución de este ejercicio antes de ver la solución propuesta.

1.13) Realizar programa en el que el ordenador "piensa" en un número al azar entre 1 y 50.
El usuario ha de adivinarlo en 5 oportunidades. El ordenador señalará si es mayor o menor.

1.14) Contabiliza personas de más de 180, entre 180 y 170, 170 y 160, y menores de 160cm. Solución con C# del ya resuelto ejercicio 1.7

1.15) Introducir nº (finaliza con el 999) . Muestra máximo y cuantas veces se ha repetido.

1.16) Cajero automático: Solución del ejercicio 1.10

1.1) Ejecutar el siguiente programa en Basic

```
REM Programa números decimal, binario , hexadecimal
10 x1=&h0001: x2=1 : x3=&b0001
20 print x1,x2,x3
30 y1=&h000A: y2=10: y3=&b00001010
40 print y1,y2,y3
50 z1=&h001B: z2=27: z3=&b00011011
60 print z1,z2,z3
70 end
```

1.2) Comprobar con la calculadora del S.O. (ver científica) que a cada 4 digitos binarios corresponden a 1 digito hexadecimal. Ejemplos: 1001_2 -> 9_{16} , 1100_2 ->C_{16} , $1000\ 1001_2$ ->89_{16}

1.3) Programa que calcula la suma de los números enteros entre 1 y 10 (1+2+3+4....)

```
REM Programa suma números enteros

suma =0: conta =0
WHILE conta <10
 conta = conta + 1
 suma = suma + conta
WEND

PRINT “La suma es”; suma
END
```

Llenguaje C:

```
#include <iostream>
using namespace std;

void main()
{
    int suma=0; int conta =0;

    while (conta<10)
    {
        conta = conta +1;
        suma = suma +conta;
    }
    printf("\n\n La suma es %d ",suma);
    printf("\n");
}
```

1.4) Programa producto números enteros del 10 al 20 10*12*14...

REM Programa **producto** números enteros

```
suma =0: conta = 10: producto =1
 WHILE conta <= 20
  producto = producto * conta
  conta = conta + 2
WEND

PRINT “El producto es”; producto
END
```

1.5) Programa : El usuario introduce números y el ordenador cuenta cuantos son positivos.

REM Programa conta número positivos

```
positius =0: conta =0
INPUT "Introduce número";numero

WHILE numero <>999
  conta = conta + 1
  if numero >0 then positivos = positivos +1

INPUT "Introduce número";numero
WEND

PRINT "Has introducido un total de "; conta
PRINT "y son positivos "; positivos
END
```

TUTORIAL INICIO C# : http://functionx.com/csharp/Lesson01.htm

```
namespace Proyecto1
{
    class Holamundo
    {
        public static void Main(String[] args)
        {
            Console.WriteLine("¡Hola {0}!", args[0]);
        }
    }
}
```

Compilar: csc Holamundo.cs
Ejecutar: Holamundo Maria

1.6) Programa : El usuario introduce números y el ordenador cuenta cuantos son positivos.

```
class Program
 {
      static void Main(string[] args)
      {
          int positivos = 0; int conta = 0; int numero;

          Console.WriteLine("Introduce número ");
          numero = Int32.Parse(Console.ReadLine());

          while (numero != 999)
          {
              conta = conta + 1;
              if (numero > 0) positivos = positivos + 1;

              Console.WriteLine("Introduce número ");
              numero = Int32.Parse(Console.ReadLine());
          }

          Console.WriteLine("Introducidos un total de {0}", conta);
          Console.WriteLine("y son positivos {0}", positivos);

      } //fin Main
 }
```

1.7) Contabiliza personas de más de 180, entre 180 y 170, 170 y 160, y menores 160cm.

```
conta1 = 0:conta2=0:conta3 =0:conta4 =0
INPUT "Introduce altura en cm"; altura
WHILE altura <> 999
  if altura>=160 then
    if altura>=170 then
      if altura >=180 then
        conta1 = conta1 + 1
      else
        conta2 = conta2 + 1
      fi
    else
      conta3= conta3 + 1
    fi
  else
    conta4=conta4 + 1
  fi
INPUT "Introduce altura en cm"; altura
WEND

print "Más grandes de 180 " ;conta1
print "Entre 170 y 180 ";conta2
print "Entre 160 y 170 ";conta3: print "Más bajas que 160 ";conta4
END
```

Capítulo 2

Bucles y Funciones

2.1) Programa en el que el ordenador lanza 50 veces un dado y cuenta las veces sale el nº 1.

Ejemplo para el uso de 'bucles' *for* para repetir un número determinado de veces unas instrucciones. También muestra cómo conseguir números al azar.

2.2) Programa que muestra 15 líneas como estas: 1 12 123 1234
Ejemplo del uso de dos bucles anidados de tipo *for*

2.3) Programa que dibuja un Triangulo isósceles
Una aplicación más compleja i completa del uso de bucles anidados.

2.4) Programa que señala si es múltiplo del número 5
La primera función que aplicamos retorna un valor boolean (verdadero/falso) si el número que enviamos per analizar es o no múltiplo de 5.

2.5) Programa que muestra el día que será mañana. Ex: 31/12/08 -> 01/01/09
Clásico ejercicio donde se ponen en práctica los conocimientos de programación estructurada. Se deja como ejercicio pendiente la versión en C# (por otro lado, nada complicada de realizar)

2.6) Programa para calcular Potencias, Tensiones e Intensidades. P = V * I
Más prácticas con funciones simples, comparando diferentes lenguajes de programación.

2.7) Área Triangulo – Variables locales/"globales"
¿Qué pasa si necesitamos una variable "global" a todas las funciones del proyecto?.

2.8) Cálculo raíz cuadrada.
Podemos crear nuestras propias 'instrucciones de sistema operativo', compilando este programa y ejecutando, per ejemplo, desde MS-DOS.

2.9) Calculadora
Ejemplo completo para trabajar con varias funciones en un programa .

2.10) Una empresa nos ha encargado un programa para calcular las nóminas de los trabajadores.

El sueldo base semanal sale aplicando la siguiente fórmula: horastrabajo * preciohora + horesextra*preciohoraextra

El preciohora es una constante=6. El preciohoraextra depende de las h.extra hechas: si son menos de 10h extras semanales, el precio es un 50% mayor que el preciohora (* 1,5). Si se hacen entre 10 y 20h extra, el precio es un 40% mayor. Si se hacen más de 20h, el precio es un 20% mayor.

Si el trabajador es de categoría 3, el preciohora es el constante.
Si es de categoría 2; el preciohora es un 25% mayor y si es de categoría 1 es un 45% más....

2.1) Programa en el que el ordenador lanza 50 veces un dado y cuenta las veces sale el nº 1.

```
contador =0
randomize timer
cls

for t = 1 to 50
  dado = int(rnd*9)+1
  print dado;
  if dado = 1 then contador = contador + 1
next t

print "Han salido ";contador ;" números  1 a la lista"
end
```

VERSIÓN C#

```
static void Main(string[] args)
    {
          int contador =0; int dado=0;
          Random numero = new Random();
          Console.Clear();

          for (int t = 0; t<=50; t=t + 1)
          {
              dado = numero.Next(1, 7);
              if (dado == 1)
              {
                  contador = contador + 1;
                  Console.ForegroundColor = ConsoleColor.Red;
              }
              else Console.ForegroundColor = ConsoleColor.Gray;

              Console.Write(" {0}", dado);
          }

          Console.WriteLine("-");
          Console.WriteLine("Repetido el número1 {0} veces",contador);
    }
```

Google: ***Consola C#*** , console C#
http://www.c-sharpcorner.com/UploadFile/scottlysle/ColorConsoleCS06082008055747AM/ColorConsoleCS.aspx

2.2) Programa que muestra 15 líneas como estas: 1 12 123 1234

1
12
123 ....

```
static void Main(string[] args)
 {
            int i, j;
            for (i = 1; i <= 15; i++) // 15 lineas
            {
                for (j = 1; j <= i; j++)  // números a cada línea
                Console.Write(" - {0}", j);
                Console.WriteLine("  ") ;
            }
  }
```

2.3) Programa que dibuja un Triangulo isósceles

```
static void Main(string[] args)
 {
        Console.Clear();

    // dibujo de cada linea (bucle externo)
    for (int fila=1; fila <= 7; fila ++)
    {
        //dibuja espacios en blanco (1er bucle interno)
        for (int espacios = 7 - fila; espacios > 0; espacios--)
            Console.Write(" ");// espai en blanc

        // dibuja estrellas (2° bucle interno)
        for (int conta = 1; conta < (2 * fila); conta++)
            Console.Write("*");

        Console.WriteLine(" ");
    }
  }
```

2.4) Programa que señala si un número es múltiplo del número 5

```
class Exercici4
 {
       static void Main(string[] args)
       {
           int num = 1;
           bool respuesta;

           while (num <= 50)
           {
               Console.Write(" - {0}", num);
               respuesta = multiplo5(num);

               if (respuesta) Console.WriteLine(" Es múltiplo de 5");
               else Console.WriteLine(" No es múltiplo de 5");

               num++;
           }
       }


       public static bool multiplo5 (int n)
       {
           if((n % 5) !=0) return false;
           else return true;
       }

  }
```

2.5) Programa que muestra el día que será mañana. Ex: 31/12/08 -> 01/01/09

```
#include <iostream>
using namespace std;
int funcion_divisor(int numero, int divisor);

void main()
{
	int d,max,m,a,resto;

	printf("Introduce el dia: ");
	scanf("%d",&d);
	printf("\nIntroduce el mes: ");
	scanf("%d",&m);
	printf("\Introduce el anyo: ");
	scanf("%d",&a);
	printf("\nHoy es dia %d de %d del %d",d,m,a);

	if (m==4 || m==6 || m==9 || m==11) max=30;
	if (m==1 || m==3 || m==5 || m==7 || m==8 || m==10||m==12) max=31;

	if (m==2)
	{
		resto = funcion_divisor(a,4);
		if (resto==0) max=29;
		else max=28;
	}

	d++;

	if (d>max)
	{
		d=1;
		m++;
		if (m>12) { m=1; a++; }
	}

	printf("\n\ny mañana será %d de %d del %d",d,m,a);
	printf("\n\n");
}



int funcion_divisor(int numero, int divisor)
{
	int resto = numero % divisor;
	return (resto);
}
```

Capítulo 3

Clases

3.1) Creación objeto y asignación de variables propias del objeto.

Podemos ver como se crea un objeto en base a una clase, en lenguaje C++, después en C# 2.0 (Visual Studio 2005) y finalmente con C# 3.0 propio del Visual Studio 2008.

3.2) Creación objetos y asignación de variables propias de los objetos.

Asignación de valores a variables privadas de la clase a través de funciones públicas (accesibles desde cualquier parte del código)

3.3) Asignación propiedades del objeto.

Variables privadas de la clase utilizadas a través de funciones get/set.
OJO! con las mayúsculas

3.4) – 3.14) EN VERSIÓN COMPLETA DEL LIBRO

3.1) Creación objeto y asignación de variables propias del objeto.

Versión C++ , en dos archivos:

cabecera.h

```
#include <iostream>
#include <string>
using namespace std;

class FichaDatos

{
      string nombre;
      int edad;

 public:
 FichaDatos(string dato1, int dato2)
 {
      nombre=dato1;
      edad=dato2;
 }

 void Cargar()
 {
      nombre = "Pedro";
      edad = 29;
 }
 void Mostrar();

};
```

principal.cpp

```
#include <iostream>
#include <string>
using namespace std;
#include "cabecera1.h"

void FichaDatos::Mostrar()
{
      cout<<"El nombre es "<<nombre<<endl;
      cout<<"edad "<<edad;
}

void main (void)
{
      FichaDatos Paciente("Ana",29);

      Paciente.Mostrar();
      Paciente.Cargar();
      cout<<endl<<endl;
      Paciente.Mostrar();

}
```

Versión C# 2.0 : (Visual Studio 2005)

```
using System.Text;

namespace Ejercicio_3_1
{
    public class Propiedades
    {
        public long identificador;
        public string propietario;
        public double precio;
    }
    public class Ejercicio1
    {
        static void Main()
        {
            Propiedades Casa = new Propiedades();
            Casa.identificador = 1001;
            Casa.propietario = "Joan Pera";
            Casa.precio = 999999;

            Console.WriteLine("Lista Propiedades"); ;
            Console.Write("Propiedad {0}:  ", Casa.identificador);
            Console.WriteLine("Propietario {0}: ", Casa.propietario);
            Console.Write(" precio:  ");Console.Write(Casa.precio);
            Console.WriteLine(); Console.WriteLine();
        }
    }
}
```

Versión C# 3.0 : (Visual Studio 2008)

```
namespace Ejercicio_3_1
{
    public class Propiedades
    {
        public long identificador;
        public string propietario;
        public double precio;
    }
    public class Exercici1
    {
        static void Main()
        {
           var Casa = new Propiedades();

            Casa.identificador = 1001;
            Casa.propietario = "Joan Mas";
            Casa.precio = 999999;
            Console.WriteLine("Lista Propiedades"); ;
            Console.Write("Propiedad {0}:  ", Casa.identificador);
            Console.WriteLine("Propietario {0}: ", Casa.propietario);
            Console.Write(" precio:  ");Console.Write(Casa.precio);
        }
    }
}
```

3.2) Creación objetos y asignación de variables propias de los objetos.

```
namespace Ejercicios_Cap3
{
    public class Propiedades
    {
        public long Identificador;
        private string propietario;
        public int habitaciones;
        double precio;

        public Propiedades (string nom, double precio)
        {
                propietario = nom;
                this.precio = precio;
        }

        public string Mostra_Propietario ()
        {
            return (this.propietario);
        }

        public double Mostra_precio()
        {
            return (this.precio);
        }
    }

    public class Exercicio_3_2
    {
        static void Main()
        {
            Propiedades Casa = new Propiedades("desconocido",999999);
            Casa.Identificador = 1001;
            Casa.habitaciones = 4;

            Propiedades Casa2 = new Propiedades("Maria Puyol", 30000);
            Casa2.Identificador = 1002;
            Casa2.habitaciones = 3;

            Console.WriteLine("Llistat Propiedades"); ;
            Console.Write("Propiedad {0}:  ", Casa.Identificador);
            Console.Write("Propietario:  ");
            string veure = Casa.Mostra_Propietario();
            Console.Write(veure);
            Console.Write("\n habitaciones:       ");
            Console.Write(Casa.habitaciones);
            Console.Write(" precio:  ");
            double veure_precio = Casa.Mostra_precio();
            Console.WriteLine(veure_precio.ToString());

            Console.Write("\n\n");
            Console.Write("Propiedad {0}:  ", Casa2.Identificador);
            Console.Write("Propietario:  ");
            veure = Casa2.Mostra_Propietario();
            Console.Write(veure);
```

```
            Console.Write("\n habitaciones:       ");
            Console.Write(Casa2.habitaciones);
            Console.Write(" precio:  ");
            veure_precio = Casa2.Mostra_precio();
            Console.WriteLine(veure_precio.ToString());
        }
    }
}
```

3.3) Asignación propiedades del objeto.

```
namespace Ejercicio_3_3
{
    public class Vehicles
    {
        private string marca;private double precio;
        private double potencia;

        public string Marca
        {
            get { return marca; }
            set { marca = value; }
        }

        public double FuncionPrecio
        {
            get
            {
                if (precio <= 0) return 0;
                else return precio;
            }
            set { precio = value; }
        }

        public double Potencia
        {
            get
            {
                if (potencia > 140)
                {
                    Console.WriteLine("Impuesto por alta poténcia");
                }
                return potencia;
            }
            set { potencia = value; }
        }
    }


class Program
    {
        static void Main(string[] args)
        {
            Vehicles Familiar1 = new Vehicles();
            Familiar1.Marca = "Seat";
            Familiar1.Potencia = 150;
            Familiar1.FuncionPrecio = 999999;

            Console.WriteLine("Vehiculos en la tienda");
            Console.WriteLine("Marca: {0} ", Familiar1.Marca);
            Console.WriteLine("Potencia: {0} ", Familiar1.Potencia);
           Console.WriteLine("Precio: {0} ", Familiar1.FuncionPrecio);
        }
    }
}
```

Capítulo 4

Arrays

4.1) Declaración e inicialización Arrays unidimensionales (C # 2008 /2005)

```
namespace Arrays1
{
 public class Ejercicio1
  {
   static void Main()
  {
  var Temperatura = new double[] { 16.55, 15.28, 16.28,16.72, 15.54 };

       // C# 2005
        //double [] Temperatura = {16.55, 15.28, 16.28, 16.72, 15.54 };

        try
        {
          for (int i = 0; i < 10; i++)
          Console.WriteLine("Valor {0} : {1}", i, Temperatura[i]);
        }
        catch (IndexOutOfRangeException)
        {
          Console.WriteLine("Límite array excedido ");
        }

      } // fin Main

    }
}
```

4.2) C# 2008 / 2005 : MULTIDIMENSIONALES

```
public static class Ejercicio2
    {
      static void Main(string[] args)
       {
            // C# 2005
            // double[,] Temp =

         var Temp = new double [3, 7]
           {
             {16.55, 15.28, 16.28, 16.72, 15.54, 15.64, 16.08 },
             {16.85, 16.98, 17.18, 16.72, 17.24, 17.34, 17.38 },
             {18.55, 17.88, 18.98, 16.72, 16.54, 18.01, 18.48 }
           };

          for (int dia = 0; dia < 7; dia++)
          {
           for (int i = 0; i < 3; i++)
           Console.WriteLine("Valores dia {0}:{1}", dia, Temp[i, dia]);
           Console.WriteLine();
           }
         }
    }
```

4.3) Asignación de valores al array 'Jugadores' (C# 2008 / 2005)

```
namespace Arrays3
{
    class Program
    {
        static void Main()
        {
            string[] jugadores = new string[100];

           // C# 2008
            //  var jugadores = new string[100];


            asignar(jugadores);

            for (int t = 0; t <= 5; t++)
                Console.WriteLine(jugadores[t]);
        }


        public static void asignar(params string[] fichas)
        {
            fichas[0] = "Martina";
            fichas[1] = "Joan";
            fichas[2] = "Ana";

            for (int t=3; t<=5; t++)
            {
              Console.WriteLine("Entra nombre jugador n° {0}", t);
              string teclado = Console.ReadLine();

              fichas[t] = teclado;
            }
        }
    }

}
```

ÍNDEX

PARA MÁS INFORMACIÓN DE LIBRO COMPLETO:

www.ibserveis.com/libro-programacion.aspx

www.ibserveis.com

y en www.bubok.es