TEMA 2

IDENTIFICACIÓN DE LOS ELEMENTOS DE UN PROGRAMA INFORMÁTICO

"Lo mejor de los booleanos es que si te equivocas,

estás a un sólo bit de la solución correcta"

ESTRUCTURA DE UN PROGRAMA

Los programas Java se componen de unos ficheros .java (contienen el código fuente). Dichos ficheros se compilan y se construyen los ficheros equivalentes .class (bytecode). Estos ficheros no tienen por qué estar en una carpeta concreta.

La aplicación se ejecuta desde el método principal main() situado en una clase. Dicha clase es la clase principal (por ella se empieza a ejecutar el programa). No tiene por qué llamarse main.

```
public class Holamundo{

 public static void main(String [] args){

……

}

}
```

Las primeras líneas que forman parte de un programa suelen estar compuestas de comentarios acerca del nombre del programa, el programador, la fecha de creación, observaciones... Aparecen con // si lo que viene a continuación es texto en una sola línea o con /* ..... */ cuando se trata de más de una línea. Hay en Java otro tipo de comentario

```
/**  comentario de documentación, de una o más líneas
*/
```

TIPOS DE DATOS SIMPLES

Los tipos de variables de Java ocupan siempre la misma memoria y tienen el mismo rango de valores en cualquier tipo de ordenador.

Enteros:

byte	8 bits	[-128,127]
short	16 bits	[-32768,32767]
int	32 bits	[-2147483648,2147483647]
long	64 bits	aprox 10 exp 18

Reales en coma flotante:

float	32 bits	aprox 10 exp 38
double	64 bits	aprox 10 exp 308

Por ejemplo: 3.14 2e12 3.1E12

Booleanos: boolean

Toman dos valores: true (verdadero) y false (falso).

```
boolean esMenorDeEdad, esPensionista;
int edad;
```

Ocupan un bit.

Caracteres:char

Los mismos que en C, incluidas las secuencias de escape, pero no podemos referirnos a ellos como números. Incluyen prácticamente los caracteres de todos los idiomas.

Ocupan 16 bits.

Por ejemplo: a \t \u???? [????] es un número *unicode*(hay secuencias unicode para letras en griego, cirílico, hebreo, árabe...)

Se pueden comparar caracteres.

VARIABLES

Variables

Zona de memoria donde se almacena información de un tipo de dato simple.

Su declaración consiste en definir el nombre de la misma con su tipo correspondiente. Además se pueden inicializar en la declaración

Tipo nombre;
Tipo nombre1, nombre2,…;
Tipo nombre=valor;

Los nombres deben empezar por una letra o por el caracter _ o $, después pueden combinar letras y números. Se distinguen las mayúsculas de las minúsculas y no hay longitud máxima.

Por convenio empiezan por minúscula y si contienen varias palabras, se unen y todas las palabras excepto la primera empiezan por mayúscula.

Se pueden inicializar en la declaración

```
int x=0;
float y =3.5;
char letra='a';
```

Las variables se pueden declarar en cualquier sitio dentro de un programa, sólo se podrán usar en el bloque {} en el que se declaran

Palabras clave

Las siguientes son las palabras clave que están definidas en Java y que no se pueden utilizar como indentificadores:

abstract	continue	for	new	switch
boolean	default	goto	null	synchronized
break	do	if	package	this
byte	double	implements	private	threadsafe
byvalue	else	import	protected	throw
case	extends	instanceof	public	transient
catch	false	int	return	true
char	final	interface	short	try
class	finally	long	static	void
const	float	native	super	while

Palabras Reservadas

Además, el lenguaje se reserva unas cuantas palabras más, pero que hasta ahora no tienen un cometido específico. Son:

cast	future	generic	inner
operator	outer	rest	var

TIPOS DE OPERADORES

Operadores aritméticos

Suma: +
Resta: -
Producto: *
División: /
Módulo o resto de una división de números enteros (no para float); %

Operadores relacionales: Evalúan las desigualdades tradicionales

Mayor: >	Mayor o igual: > =
Menor: <	Menor o igual: < =
Igual: = =	Distinto: ! =

Hay que distinguir entre el igual relacional y el igual asignación (=)

```
boolean esMenorDeEdad, esPensionista;
int edad;

esMenorDeEdad = edad < 18;
esPensionista = edad >= 65;
```

Operadores lógicos: Son necesarios para describir condiciones en las que aparecen conjunciones -y- (&&), disyunciones –o- (||) y negaciones (!)

```
boolean  trabaja, esUnTrabajadorJoven, esUnVotante;
esUnTrabajadorJoven = esMenorDeEdad & trabaja;
esUnVotante = !esMenorDeEdad;
```

Operadores de incremento y decremento: Son dos operaciones típicas de para aumentar o disminuir en una unidad el valor de una variable contador. Son ++ y --

Ejemplo

```
int a, b;
a = 5;
b = 7;
a = b++  : asigno y sumo a=7, b=8
a = ++b  : sumo y asigno b=8, a=8
```

a++ equivale a de a = a + 1 . Igual que ++a

Operadores de asignación

=	A = B	Asignación.
*=	A *= B	Multiplicación y asignación. La operación A*=B equivale a A=A*B.
/=	A /= B	División y asignación. La operación A/=B equivale a A=A/B.
%=	A %= B	Módulo y asignación. La operación A%=B equivale a A=A%B.
+=	A += B	Suma y asignación. La operación A+=B equivale a A=A+B.
-=	A -= B	Resta y asignación. La operación A-=B equivale a A=A-B.

CONSTANTES

Si en la declaración de una variable se pone por delante la palabra reservada “final”, significa que se trata de una constante, es decir que no se puede modificar su valor a lo largo del programa. Es el equivalente al define de C.

Ej. final double pi=3.1416;

Nociones básicas para realizar los ejercicios

Salida por pantalla

Instrucciones de salida:

```
System.out.println(elementos)
System.out.println()
System.out.print(elementos)
```

System es una clase especial que contiene un objeto llamado out que pertenece a la clase PrintStream.
Sólo se diferencian en que println salta de línea después de escribir.

Si tiene como elemento una cadena de caracteres, se escribe delimitada por comillas. Si en el interior lleva comillas se escribe \".

Ej. System.out.print("El dijo \"No");

El salto de línea \n, \b retroceso, \t tabulador, \f cambio de página, \r volver al principio de la l ínea.
Si la cadena de caracteres es demasiado larga para que coja en una línea partimos la cadena y usamos el signo +.

```
System.out.println("Esta cadena" +
                                " se escribe" +
                                " en tres líneas");
```

Para imprimir números o variables:

```
Int  s=80;

System.out.println("Voy a escribir una var:" + s + "y un número" + 30*15);
```

EJEMPLOS

1. ¿Qué tipo de variable usarías para guardar los siguientes datos:
   El precio de un artículo,
   la inicial de un nombre,
   el número de hijos de una familia,
   si una persona pertenece a una familia numerosa,
   la altura de un armario.

2. ¿Compilarán y funcionarán los siguientes códigos?¿Cuál será la salida

```
int a = ‘a’;
System.out.printl(a);

int pi = 3.14
System.out.println(pi);

double pi = 3,14;
System.out.println(“pi”);

boolean adivina = (1 == 4);
System.out.println(adivina);
```

3. Modifica los siguientes programas para hacer que compilen y funcionen:

```
public static void main(String [] args)
{
int n1=50;
int n2=30, suma=0; n3,
suma=n1+”n2”;
System.out.println(“LA SUMA ES: “ +” suma”);
suma=suma+n3;
System.out.println(sum);
}
```

```
public static void main(String [] args)
{
int n1=50,n2=30,
boolean suma=0;
suma=n1+n2;
System.out.println(“LA SUMA ES: “ + suma);
}
```

4. El siguiente programa tiene tres fallos

```
public static void main(String [] args)
{
int numero=2,
cuad=numero * número;
System.out.println(“EL CUADRADO DE “+NUMERO+” ES: “ + cuad);
}
```

5. ¿Qué mostrará es siguiente código en pantalla?

```
int num=5;
num +=  1 ;
num++;
int resto=num%2;
System.out.println(num);
System.out.println(resto);
boolean  par= (resto==0);
```

EJERCICIOS

HOJA_1

Leer desde teclado(Scanner)

1. Importar java.util
2. Crear un objeto Scanner al principio del programa:

```
Scanner entrada=new Scanner(System.in);
```

- Leer números: métodos nextXXX() donde XXX es el tipo de dato

```
System.out.println("Anota un número entero:");
int edad=entrada.nextInt();
```

Usamos el método useLocale(Locale.ENGLISH), para poder leer números decimales con punto. Al método useLocale sólo hay que usarlo cuando se crea el Scanner.

```
Scanner sc=new Scanner(System.in);
sc.useLocale(Locale.ENGLISH);
System.out.println("Anota un número, usa punto para los decimales:");
double area=entrada.nextDouble();
```

- Leer un carácter:

```
System.out.println("Anota un carácter:");
char letra=sc.nextLine().charAt(0);
```

Si después de leer un dato numérico queremos leer un dato tipo char tenemos que limpiar el buffer con nextLine o si no nos saltará la excepción StringIndexOutOfBoundsException. La forma de hacerlo sería:

```
System.out.println("Anota un número:");
int numero=sc.nextInt();
System.out.println("Anota un carácter:");
sc.nextLine(); //Instrucción para limpiar el buffer
char letra=sc.nextLine().charAt(0);
```

- Leer cadenas (más adelante)

```
String nombre=entrada.next();  lee hasta que encuentra carácter blanco
String nombreCompleto=entrada.nextLine(); lee hasta que encuentra INTRO (ojo con el buffer y el carácter INTRO)
```

Otra forma de leer desde teclado (BufferedReader)

Siempre que necesitemos obtener datos por teclado tenemos que:

1. Importar java.io.*
2. Crear un objeto BufferedReader que es sobre el que se llama a los métodos.
3. Escribir: "throws IOException" en el método o métodos que realicen lecturas.

```
import java.io.*;
public class Leer{
        public static void main(String[] arg) throws IOException {
                // Creación de un objeto BufferedReader
        BufferedReader br=new BufferedReader( new InputStreamReader(System.in) );

                System.out.println("Introduce una cadena:");
                String nombre=br.readLine();
                // La cadena puede contener blancos
                System.out.println("Cadena leida:"+nombre);

                System.out.println("Introduce un byte:");
                byte b= Byte.parseByte(br.readLine().trim());
                //byte b= Byte.valueOf(br.readLine().trim()).byteValue();
                System.out.println("byte leido:"+b);

                System.out.println("Introduce un entero:");
                int i=Integer.parseInt(br.readLine().trim());
                //int i= Integer.valueOf(br.readLine().trim()).intValue();
                System.out.println("int leido:"+i);

                System.out.println("Introduce un long:");
                long l=Long.parseLong(br.readLine().trim());
                //long l= Long.valueOf(br.readLine().trim()).longValue();
                System.out.println("long leido:"+l);

                System.out.println("Introduce un float:");
                float f=Float.parseFloat(br.readLine().trim());
                //float f= Float.valueOf(br.readLine().trim()).floatValue();
                System.out.println("float leido:"+f);

                System.out.println("Introduce un double:");
                double d=Double.parseDouble(br.readLine().trim());
                //double d= Double.valueOf(br.readLine().trim()).doubleValue();
                System.out.println("double leido:"+d);
```

```
System.out.println("Introduce un boolean:");
boolean bo= Boolean.valueOf(br.readLine().trim()).booleanValue();
System.out.println("boolean leido:"+bo);

// Leer un caracter. No se lee hasta que no se teclea
// enter y éste queda residual.

System.out.println("\nIntroduce un caracter:");
char c=(char)br.read();
System.out.println("caracter leido:"+c); }}
```