

Prácticas: Introducción a la programación en Java



Informática (1º Ingeniería Civil)
Curso 2011/2012



Índice

- Introducción a Java y al entorno de desarrollo NetBeans
- Estructura de un programa
- Tipos de datos
- Operadores
- Sentencias condicionales
- Sentencias repetitivas
- Funciones

Introducción a Java (I)

- Objetivos:
 - Describir las características del lenguaje de programación Java.
 - Describir las herramientas ligadas a la construcción y ejecución de programas escritos en Java.
 - Construir las primeras aplicaciones en Java.

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Introducción a Java (II)

- Características:

- **Sencillo.** Elimina la complejidad de otros lenguajes.
- **Orientado a objetos.** La filosofía de programación orientada a objetos facilita la creación y mantenimiento de programas.
- **Independiente** de la arquitectura y portable. Al compilar un programa en Java, el código resultante es un tipo de código binario conocido como Java Bytecode. Este código es interpretado por diferentes computadoras de igual manera. Como el código compilado de Java es interpretado, un programa compilado de Java puede ser utilizado por cualquier computadora que tenga implementado el intérprete de Java.
- **Robusto.** Java simplifica la gestión de la memoria.
- **Multitarea.** Java puede ejecutar diferentes líneas de código al mismo tiempo.
- **Dinámico.** En Java no es necesario cargar completamente el programa en memoria, sino que las clases compiladas pueden ser cargadas bajo demanda en tiempo de ejecución.

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

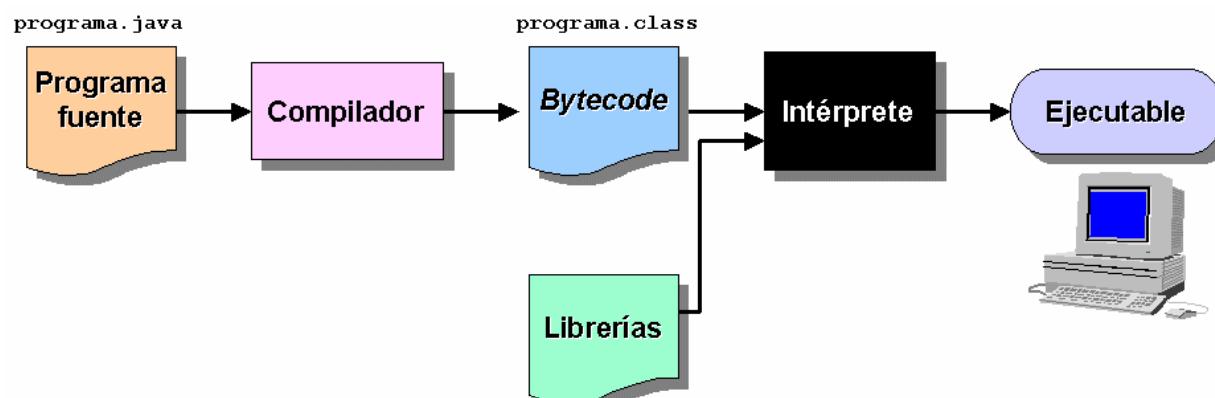
Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

- Mecanismo de creación de un programa Java
 - Java es a la vez compilado e interpretado. Con el compilador, el programa fuente con extensión .java es traducido a un lenguaje intermedio llamado Java bytecodes generándose un programa compilado almacenado en un archivo con extensión .class. Este archivo puede ser posteriormente interpretado por el intérprete de Java (Máquina Virtual de Java). La compilación se produce una vez y la interpretación cada vez que el programa se ejecuta.



Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

- Plataforma Java
 - Intérprete, Máquina Virtual Java o Java Virtual Machine
 - Interfaz de Programación de Aplicaciones o Java Application Programming Interface (Java API).
- El API de Java es una colección de componentes de software que facilitan muchas necesidades de programación, por ejemplo para construir una interfaz de usuario (GUI). El API de Java se agrupa en librerías o paquetes (packages) de componentes.



Introducción a Java (V)

- Kit de Desarrollo Java
 - Para escribir un programa Java es necesario tener instalado el Kit de desarrollo de Java o JDK (*Java Development Kit*), también llamado Java SDK (*Software Development Kit*).
 - Contiene el software necesario para que los programadores compilen, depuren y ejecuten programas escritos en Java.
 - Tanto el software como la documentación son gratuitos según la licencia de Sun Microsystems.
 - En la sala está instalado el JDK 6.16 y el entorno de desarrollo NetBeans 6.7.1.
 - www.java.com
 - <http://netbeans.org/>
 - <http://java.sun.com>
 - <http://www.sun.com>

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

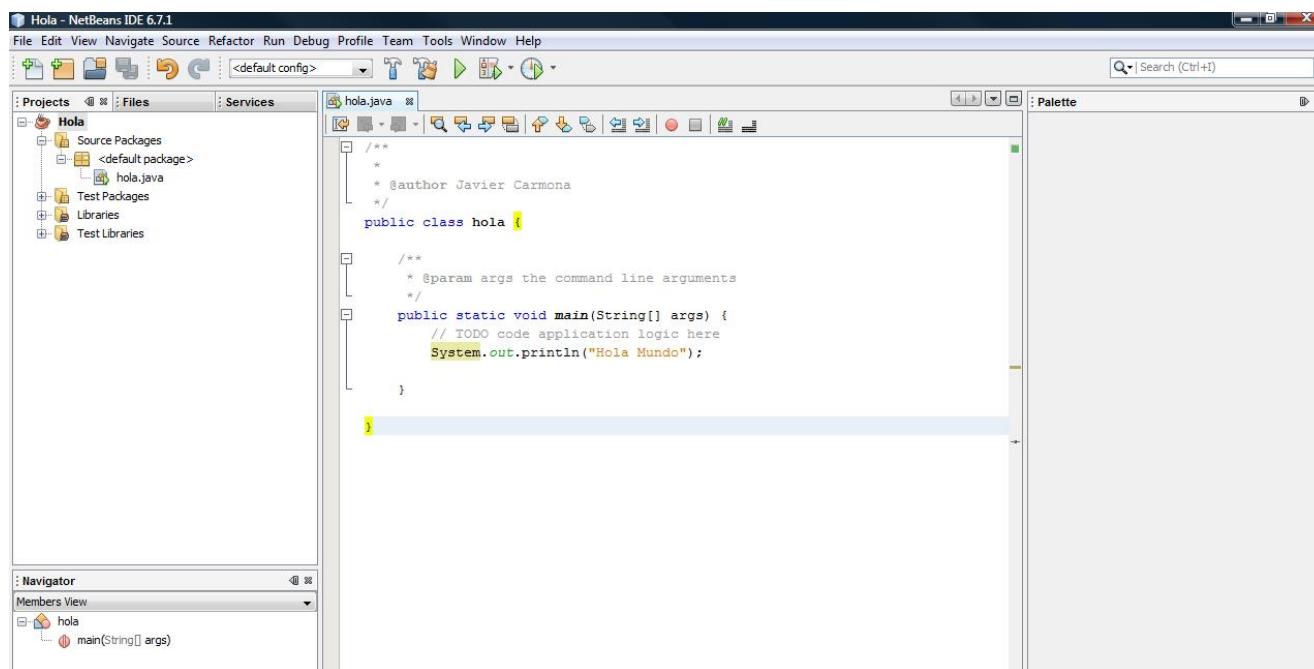
Sentencias
condicionales

Sentencias repetitivas

Funciones

Entorno NetBeans (I)

- NetBeans IDE (*Integrated Development Environment*)
- Ejemplo: “Hola Mundo”
 - <http://netbeans.org/kb/docs/java/quickstart.html>



Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Entorno NetBeans (II)

- NetBeans IDE
- Ejemplo: “Hola Mundo”

Bloque del programa

```
hola.java
/*
 * @author Javier Carmona
 */
public class hola {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        System.out.println("Hola Mundo");
    }
}
```

Entorno NetBeans (III)

- Compilación
 - Ejecutar → Limpiar y generar Main Project
- Ejecución
 - Ejecutar → Ejecutar Main Project
- Ejecución Manual
 - En D:/NetBeansProjects/NombreProyecto
 - Código fuente (.java):
D:/NetBeansProjects/NombreProyecto/src
 - Ejecutable (.jar):
D:/NetBeansProjects/NombreProyecto/dist
 - Ejecución manual:
java -jar nombreProyecto.jar

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Entorno NetBeans (IV)

- Los entornos de desarrollo permiten la ejecución “paso a paso” de los programas para realizar tareas de depuración.
- Desde el menú Depurar de NetBeans, están las opciones “Paso a paso” y “Continuar ejecución”, que se ejecutan con F7 y F8 respectivamente.
 - Paso a paso (F7): Ejecuta línea a línea. En las llamadas a funciones, ejecuta la función también paso a paso.
 - Continuar ejecución (F8): Ejecuta línea a línea. En las llamadas a funciones, ejecuta la función de un único salto.
- En este modo de ejecución podemos ver el valor de las variables y cómo van cambiando.

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Estructura de un programa (I)

- Objetivos:
 - Describir la estructura del código fuente de una aplicación Java
 - Presentar los conceptos de comentario y de identificador dentro del código fuente de un programa.
- Java siempre emplea la Programación Orientada a Objetos por lo que el código se incluye dentro de las clases. Una clase es combinación de datos (constantes y variables) y métodos (o funciones).

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Estructura de un programa (II)

- La clase principal y la función *main*
 - Un programa puede construirse empleando varias clases.
 - En el caso más simple se utilizará una única clase, que contiene el programa o función principal: *main ()* y es ahí donde se incluyen las sentencias (o instrucciones) del programa principal.
 - Las sentencias se separan entre sí por caracteres punto y coma.
 - Estructura de un programa simple en Java:

```
public class ClasePrincipal {  
    public static void main(String[] args) {  
        sentencia_1;  
        sentencia_2;  
        // ...  
        sentencia_N;  
    }  
}
```

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Estructura de un programa (III)

- Los **identificadores** son nombres que se les asignan a variables, métodos, clases, ... en el código fuente de un programa.
- Todo identificador que se use en un programa Java debe definirse antes de utilizarlo.
- Existen una serie de palabras reservadas por el lenguaje que el programador no puede usar.

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

Ejercicio

- Modificar el programa *Hola* para que visualice otro nombre por pantalla al ejecutarse.

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

**Estructura de un
programa**

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Tipos de datos (I)

- Objetivos:
 - Describir los tipos de datos primitivos (numéricos, booleano y de tipo carácter) en Java y su formato de representación.
 - Escribir la declaración de constantes y variables de cualquiera de los tipos de datos primitivos.
- Todo lenguaje de programación consta de elementos específicos que permiten realizar las operaciones básicas de la programación: tipos de datos, operadores e instrucciones o sentencias.

Contenido

Introducción a Java y al entorno de desarrollo NetBeans

Estructura de un programa

Tipos de datos

Operadores

Sentencias condicionales

Sentencias repetitivas

Funciones

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Tipos de datos (II)

- Tipos de datos primitivos en Java.
 - A todo dato (constante, variable o expresión) le corresponde un *tipo* específico en Java.

Tipo de datos simple	Representación / Valor	Valor por defecto
int	Número entero	0
double	Número real	0.0
boolean	true o false	false
char	Carácter	\u0000

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Tipos de datos (II)

- Declaración de variables:
 - Una variable es un espacio de la memoria correspondiente a un dato cuyo valor puede modificarse durante la ejecución de un programa y que está asociado a un identificador. Toda variable ha de declararse antes de ser utilizada en el código de un programa Java. En la declaración debe indicarse el identificador y el tipo de dato asociado.
 - Identificaremos los datos de entrada y de salida de nuestro programa y los definiremos como variables con el tipo de datos correspondiente.
 - La declaración de una variable en el código fuente de un programa Java puede hacerse:
 - tipo_de_dato identificador_de_la_variable;
 - tipo_de_dato ident_1, ident_2, ..., ident_n;
 - Ejemplo:
 - int n;
 - double x, y;

Una variable queda definida dentro del
bloque {} de sentencias en el que ha sido
declarada

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Tipos de datos (II)

- Declaración de variables:
 - La declaración e inicialización de una variable de tipo primitivo puede realizarse de forma simultánea en la misma línea empleando el **operador asignación (=)**.

```
int n = 15;
```

Es similar a escribir:

```
int n;  
n = 15;
```

- El valor asignado a la variable puede modificarse las veces que se quiera durante la ejecución del programa.

Tipos de datos (II)

- Declaración de variables finales o constantes:
 - Las variables finales se declaran con la palabra reservada “final” antes del identificador del tipo de dato. Una vez inicializada una variable final su valor no puede ser modificado.
 - Ejemplo:
 - `final int MAXIMO = 15;`

Contenido

Introducción a Java y al entorno de desarrollo NetBeans

Estructura de un programa

Tipos de datos

Operadores

Sentencias condicionales

Sentencias repetitivas

Funciones

Ejercicio

- Realizar un programa que muestre la suma de las edades de tus compañeros de fila
 - (Puedes suponer que tienes 3 compañeros de fila, de 17, 21 y 18 años).

Contenido

Introducción a Java y al entorno de desarrollo NetBeans

Estructura de un programa

Tipos de datos

Operadores

Sentencias condicionales

Sentencias repetitivas

Funciones

Ejercicio

- Realizar un programa que, dado el radio de una esfera, calcule y devuelva por pantalla el valor de la superficie y el volumen de la esfera correspondiente. Salida esperada:

Radio de la esfera: X metros

Superficie de la esfera: X metros cuadrados

Volumen de la esfera: X metros cúbicos

- Realizar un programa que dado el peso (en kg) y la altura (en m) de una persona calcule y muestre su Índice de Masa Corporal (IMS). Este índice pretende determinar el intervalo de peso más saludable que puede tener una persona. El valor de este índice se calcula mediante la siguiente expresión:
 - $\text{IMS} = \text{peso} / \text{altura}^2$

Contenido

Introducción a Java y al entorno de desarrollo NetBeans

Estructura de un programa

Tipos de datos

Operadores

Sentencias condicionales

Sentencias repetitivas

Funciones

Operadores (I)

- Objetivos:
 - Describir los operadores (aritméticos, incrementales, de relación, lógicos y de asignación y los tipos de dato sobre los que actúan).
 - Evaluar expresiones que empleen datos primitivos, operadores y paréntesis.
 - Construir expresiones que empleen combinaciones de datos primitivos, operadores y paréntesis.
- Un operador lleva a cabo operaciones sobre uno (*operador unario*), dos (*operador binario*) o tres (*operador ternario*) datos u operandos de tipo primitivo devolviendo un valor determinado también de tipo primitivo.
- Los operadores se pueden clasificar en distintos grupos.

Contenido

Introducción a Java y al entorno de desarrollo NetBeans

Estructura de un programa

Tipos de datos

Operadores

Sentencias condicionales

Sentencias repetitivas

Funciones

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Operadores (II)

- Operador asignación (=)
 - Es un operador binario que asigna el valor del término de la derecha al operando de la izquierda. El operando de la izquierda suele ser el identificador de una variable. El término de la derecha es, en general, una expresión de un tipo de dato compatible; en particular puede ser una constante u otra variable.

Operador	Descripción	Ejemplo de expresión	Resultado del ejemplo
=	Operador asignación	n = 4;	n vale 4

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Operadores (III)

- Ejemplo del operador asignación

```
public class operadorAsignacion {  
    public static void main(String[] args) {  
        int i,j;  
        double x;  
        char c;  
        boolean b;  
        String s;  
        i = 15;  
        j = i;  
        x = 12.345;  
        c = 'A';  
        b = false;  
        s = "Hola";  
        System.out.println("i = " + i);  
        System.out.println("j = " + j);  
        System.out.println("x = " + x);  
        System.out.println("c = " + c);  
        System.out.println("b = " + b);  
        System.out.println("s = " + s);  
    }  
}
```

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Operadores (IV)

- Operadores aritméticos

Operador	Descripción	Ejemplo de expresión	Resultado del ejemplo
-	Operador unario de cambio de signo	-4	-4
+	Suma	2.5 + 7.1	9.6
-	Resta	235.6 – 103.5	132.1
/	División (entera y real)	0.050 / 0.2 7 / 2	0.25 3
%	Resto de la división entera	20 % 7	6

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Operadores (V)

- Ejemplo de los operadores aritméticos

```
public class operadoresAritmeticos {  
    public static void main(String[] args) {  
        int i,j;  
        double a,b;  
        i = 7;  
        j = 3;  
        System.out.println("* Operandos enteros: i = " + i + " ; j = " + j);  
        System.out.println(" Operador suma: i + j = " + (i+j));  
        System.out.println(" Operador resta: i - j = " + (i-j));  
        System.out.println(" Operador producto: i * j = " + (i*j));  
        System.out.println(" Operador division: i / j = " + (i/j));  
        System.out.println(" Operador resto: i % j = " + (i%j));  
        a = 12.5;  
        b = 4.3;  
        System.out.println("* Operandos reales: a = " + a + " ; b = " + b);  
        System.out.println(" Operador suma: a + b = " + (a+b));  
        System.out.println(" Operador resta: a - b = " + (a-b));  
        System.out.println(" Operador producto: a * b = " + (a*b));  
        System.out.println(" Operador division: a / b = " + (a/b));  
        System.out.println(" Operador resto: a % b = " + (a%b));  
    }  
}
```

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Operadores (VI)

- Operadores relacionales

Operador	Descripción	Ejemplo de expresión	Resultado del ejemplo
<code>==</code>	Igual que	<code>7 == 38</code>	<code>False</code>
<code>!=</code>	Distinto que	<code>'a' != 'k'</code>	<code>True</code>
<code><</code>	Menor que	<code>'G' < 'B'</code>	<code>False</code>
<code>></code>	Mayor que	<code>'b' > 'a'</code>	<code>True</code>
<code><=</code>	Menor o igual que	<code>7.5 <= 7.38</code>	<code>False</code>
<code>>=</code>	Mayor o igual que	<code>38 >=7</code>	<code>true</code>

Operadores (VII)

- Ejemplo de los operadores relacionales

```
public class operadoresRelacionales {  
    public static void main(String[] args) {  
        int i,j;  
        i = 7;  
        j = 3;  
        System.out.println("* Operandos enteros: i = "+ i +" ; j = " + j);  
        System.out.println(" Operador igualdad: i == j es " + (i==j));  
        System.out.println(" Operador desigualdad: i != j es " + (i!=j));  
        System.out.println(" Operador mayor que: i > j es " + (i>j));  
        System.out.println(" Operador menor que: i < j es " + (i<j));  
        System.out.println(" Operador mayor o igual que: i >= j es " + (i>=j));  
        System.out.println(" Operador menor o igual que: i <= j es " + (i<=j));  
    }  
}
```

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Operador	Descripción	Ejemplo de expresión	Resultado del ejemplo
!	Negación (unario)	<code>!false</code> <code>!(5==5)</code>	<code>true</code> <code>false</code>
	OR (binario)	<code>true false</code> <code>(5==5) (5<4)</code>	<code>true</code> <code>true</code>
^	XOR (binario)	<code>true ^ false</code> <code>(5==5) ^ (5<4)</code>	<code>true</code> <code>true</code>
&	AND (binario)	<code>true & false</code> <code>(5==5) & (5<4)</code>	<code>false</code> <code>false</code>
	OR lógico con cortocircuito	<code>true false</code> <code>(5==5) (5<4)</code>	<code>true</code> <code>true</code>
&&	AND lógico con cortocircuito	<code>true && false</code> <code>(5==5) && (5<4)</code>	<code>false</code> <code>false</code>

Operadores (IX)

- Ejemplo de los operadores lógicos o booleanos

```
public class operadoresBooleanos {  
    public static void main(String [] args) {  
        System.out.println("Demostracion de operadores logicos");  
        System.out.println("Negacion: ! false es : " + (! false));  
        System.out.println(" ! true es : " + (! true));  
        System.out.println("OR: false | false es : " + (false | false));  
        System.out.println(" false | true es : " + (false | true));  
        System.out.println(" true | false es : " + (true | false));  
        System.out.println(" true | true es : " + (true | true));  
        System.out.println("AND: false & false es : " + (false & false));  
        System.out.println(" false & true es : " + (false & true));  
        System.out.println(" true & false es : " + (true & false));  
        System.out.println(" true & true es : " + (true & true));  
    }  
}
```

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Operadores (X)

- Operador concatenación de cadenas

Operador	Descripción	Ejemplo de expresión	Resultado del ejemplo
+	Operador concatenación	“Aprendiendo” + “Java”	“AprendiendoJava”

Contenido

Introducción a Java y al entorno de desarrollo NetBeans

Estructura de un programa

Tipos de datos

Operadores

Sentencias condicionales

Sentencias repetitivas

Funciones

Ejercicio

- Escribir un programa que calcule y muestre por pantalla las raíces de la ecuación de segundo grado de coeficientes reales. Los valores de los coeficientes se indican en el propio código del programa mediante sentencias de asignación a variables reales. Siendo la ecuación de segundo grado de la forma: $f(x) = a \cdot x^2 + b \cdot x + c$, entonces las expresiones de las raíces correspondientes se indican a continuación:

$$x_1 = \frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$$

$$x_2 = \frac{-b - \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$$

Nota: En Java no existe un operador que calcule la raíz cuadrada de una expresión x. Para el cálculo de la raíz cuadrada puede emplearse el método Math.sqrt (x). Ocurre lo mismo para las potencias. Se utiliza Math.pow (a,b), donde a es la base y b el exponente

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

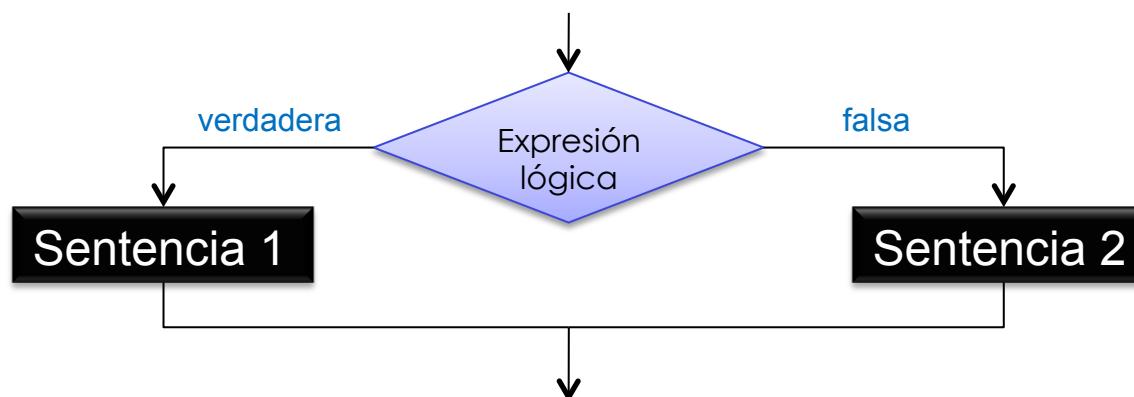
**Sentencias
condicionales**

Sentencias repetitivas

Funciones

Sentencias condicionales (I)

- Hasta ahora, la secuencia del programa ha sido ejecutar las sentencias una tras otra conforme aparecen en el programa.
- Podemos ejecutar sentencias condicionalmente, repetir un conjunto de sentencias o, en general, cambiando el flujo secuencial de la ejecución.
- En este bloque veremos sentencias condicionales y el próximo las sentencias repetitivas.
- Flujo de la sentencia condicional:



Contenido

Introducción a Java y al entorno de desarrollo NetBeans

Estructura de un programa

Tipos de datos

Operadores

Sentencias condicionales

Sentencias repetitivas

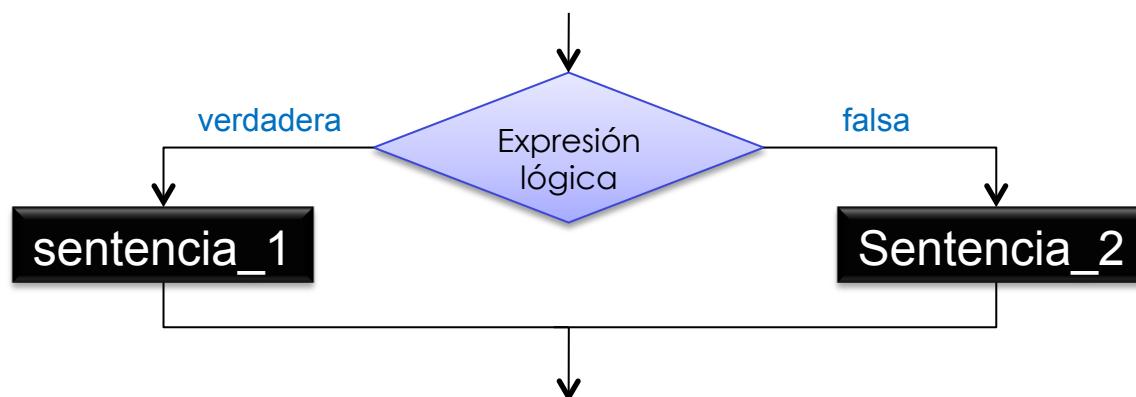
Funciones

Sentencias condicionales (II)

- Implementación en Java:

```
If (expresiónLogica) {  
    sentencia_1;  
}  
else {  
    sentencia_2;  
}
```

- Se ejecutará la sentencia_1 si la evaluación de la expresiónLógica es verdadera. En caso contrario se ejecutará la sentencia_2.



Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

**Sentencias
condicionales**

Sentencias repetitivas

Funciones

Sentencias condicionales (III)

- Ejemplo de las sentencias condicionales

```
public class esPar{  
    public static void main(String[] args) {  
        int valor;  
        valor = 5;  
        if (valor % 2 == 0) {  
            System.out.println("El numero " + valor + " es par");  
        }  
        else {  
            System.out.println("El numero " + valor + " es impar");  
        }  
    }  
}
```

Sentencias condicionales (IV)

- Ejercicio:
 - Realiza un programa en JAVA que calcule el mayor y el menor de 3 números enteros.

Contenido

Introducción a Java y al entorno de desarrollo NetBeans

Estructura de un programa

Tipos de datos

Operadores

Sentencias condicionales

Sentencias repetitivas

Funciones

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

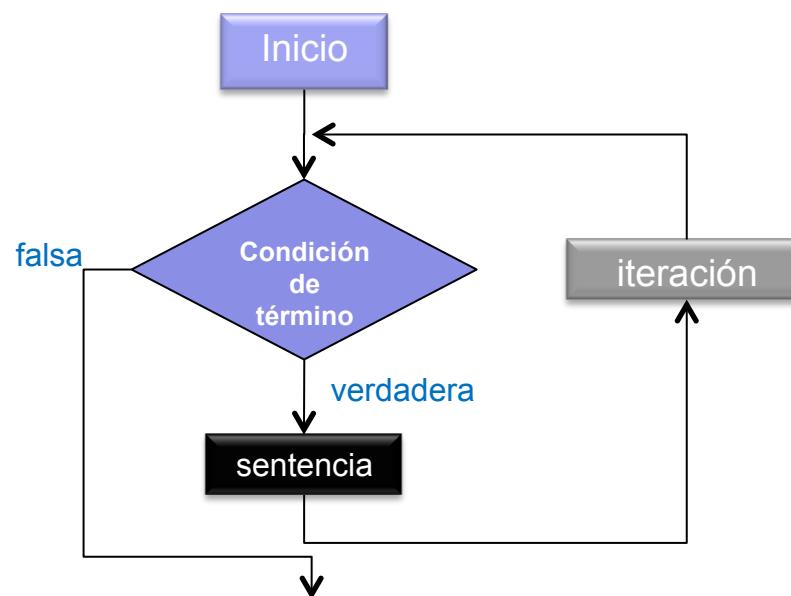
Sentencias
condicionales

Sentencias repetitivas

Funciones

Sentencias repetitivas o bucles

- Los bucles, iteraciones o sentencias repetitivas modifican el flujo secuencial de un programa permitiendo la ejecución reiterada de una sentencia o sentencias.
- Sentencia *for*



Contenido

Introducción a Java y al entorno de desarrollo NetBeans

Estructura de un programa

Tipos de datos

Operadores

Sentencias condicionales

Sentencias repetitivas

Funciones

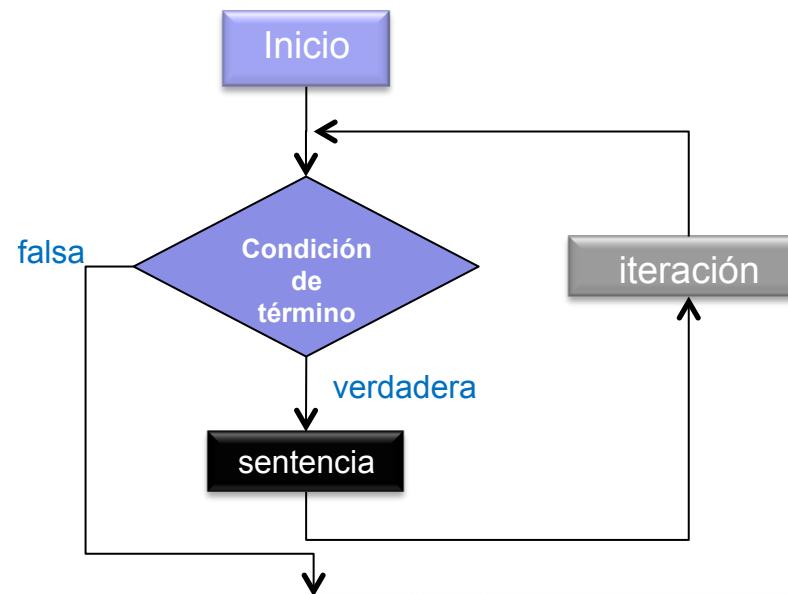
Sentencias repetitivas o bucles

- Implementación en Java.

```
for (inicio; termino; iteracion) {  
    sentencia_1;  
    sentencia_2;  
    sentencia_n;  
}
```

Ejemplo:

```
for (i = valor_inicial; i <= valor_final; i++) {  
    sentencia;  
}
```



Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Sentencias repetitivas o bucles

- Ejemplo de bucles en Java.

```
public class tablaMultiplicar{  
    public static void main(String[] args) {  
        int valor;  
        valor = 8;  
        System.out.println("Tabla de multiplicar del numero " + valor);  
        for (int i=1; i<=10; i++) {  
            System.out.println(valor + " x " + i + " = " + valor*i );  
        }  
    }  
}
```

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Funciones

- Una función (o método) es un trozo de código que puede ser llamado o invocado por el programa principal o por otro método para realizar alguna tarea específica.
- El método es llamado por su nombre seguido por una secuencia de parámetros (datos usados por el propio método para sus cálculos) entre paréntesis.
- Cuando el método finaliza sus operaciones, devuelve habitualmente un valor simple al programa que lo llama, que utiliza dicho valor de la forma que le convenga.
- Hemos utilizado varias funciones:
 - Math.sqrt (x);
 - System.out.println ();

Funciones

- Un método está compuesto por:
 - Nombre,
 - parámetros,
 - el tipo de retorno: hace referencia al tipo del valor devuelto por el método utilizando la sentencia **return**.
 - el cuerpo.
- Definimos las funciones a continuación del método *main*.

Definición de un método:

```
public static TipoRetorno NombreMetodo ( lista de parámetros){  
    Cuerpo del método  
}
```

Llamada a un método:

```
NombreMetodo (par1,par2,par3);
```

```
public static void main (String [] args){  
    int a,b,c;  
    a=5;  
    b=3;  
    c=suma (a,b);  
    System.out.println ("El resultado de la suma es" + c);  
}
```

```
public static int suma (int a, int b) {  
    return a+b;  
}
```

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Funciones

- Un ejemplo sencillo:

```
public class PruebaCubo {  
    public static void main (String [] args){  
        System.out.println ("El cubo de 6 es: " + cubo(6)); // Llamada  
    }  
  
    //Declaración del método cubo  
    public static double cubo (double x) {  
        return x*x*x;  
    }  
}
```

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

Contenido

Introducción a Java y
al entorno de
desarrollo
NetBeans

Estructura de un
programa

Tipos de datos

Operadores

Sentencias
condicionales

Sentencias repetitivas

Funciones

```
public class esPar{  
    public static void main(String[] args) {  
        int valor;  
        System.out.println("Introduce un numero entero");  
        valor = Leer.datToInt();  
        if (valor % 2 == 0) {  
            System.out.println("El numero " + valor + " es par");  
        }  
        else {  
            System.out.println("El numero " + valor + " es impar");  
        }  
    }  
}
```

Funciones

- Ejemplos:

1. Realizar alguno de los ejemplos anteriores utilizando la clase leer.java para leer los datos de entrada por teclado.
2. Completar el programa que hay en el fichero calculadora.java en el campus virtual con las funciones necesarias para que el programa funcione correctamente.
3. Incluir en la calculadora una nueva opción que realice la división entre dos números enteros.

Contenido

Introducción a Java y al entorno de desarrollo NetBeans

Estructura de un programa

Tipos de datos

Operadores

Sentencias condicionales

Sentencias repetitivas

Funciones