Capítulo 3

Introducción a la programación

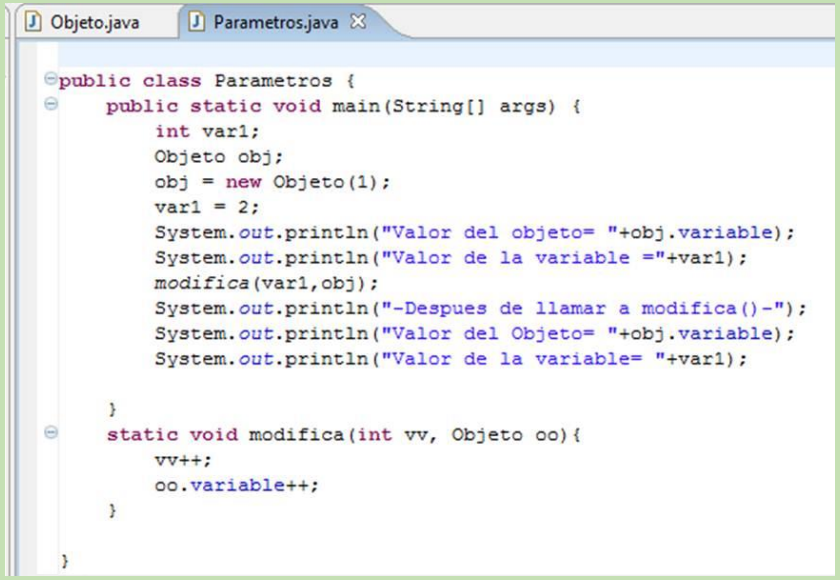
Continuar

Introducción

Java es un lenguaje que tiene muchas ventajas frente a otros lenguajes de programación: es *open source* (código abierto), esto permite ver el código fuente y modificarlo; es gratuito, no es necesario pagar una licencia para utilizarlo; es portable, puede escribirse el código en un sistema operativo y ejecutarse en otro diferente; es simple, tiene estructuras fáciles de utilizar; además, es ideal para aprender a programar.

Lenguaje de programación Java

Java es un lenguaje de programación de alto nivel orientado a objetos que se creó con el patrocinio de Sun Microsystems en 1991 como parte del *Green Proyect* compuesto por trece personas y dirigido por James Goslin. El proyecto cumplió con el objetivo principal que era desarrollar un lenguaje basado en C++, al cual se le llamó inicialmente *Oak* debido a un roble que tenía Goslin a la vista desde su ventana en las oficinas de Sun; luego pasó a denominarse *Green* tras descubrir que *Oak* era ya una marca comercial registrada para adaptadores de tarjetas gráficas y finalmente se le cambió el nombre a Java.



Características del lenguaje Java

- Lenguaje simple
- Orientado a objetos
- Multiplataforma
- OpenSource
- Libre

Ventana.java

```
1  package ventanas;
2
3  import java.awt.Graphics; // importar la clase Graphics
4  import javax.swing.JFrame;
5
6  public class Ventana extends JFrame {
7
8      // constructora
9      public Ventana() {
10         setTitle("Gráfica"); // título
11         setSize(600,250); // ventana de 600x250
12         setDefaultCloseOperation(EXIT_ON_CLOSE);
13     }
14
15     // función que dibuja la gráfica
16     public void paint(Graphics g) {
17         // para cada coordenada x ...
18         for (int x = 0 ; x < 600 ; x++)
19             // dibujamos una línea entre (x,f(x)) y (x+1,f(x+1))
20             g.drawLine(x, (int)f(x), x + 1, (int)f(x + 1));
21     }
22
23     // función a dibujar
24     double f(double x) {
25         // en Java "cos" se escribe "Math.cos" y "sen" "Math.sin"
26         return 250-(Math.cos(x/3) + Math.sin(x/5) +3) * 40 ;
27     }
28
29 }
30
```

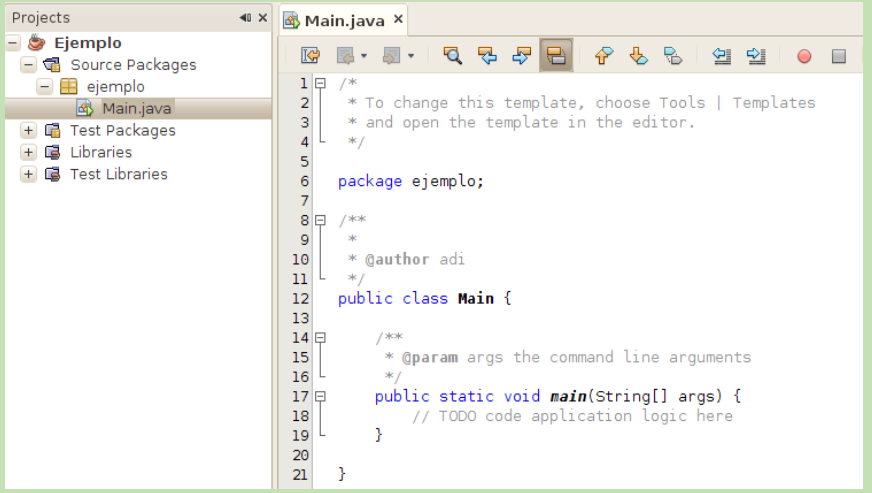
Entornos de desarrollo para Java

Un entorno de desarrollo es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. se considera que NetBeans IDE es una buena opción para aprender a programar en Java porque aunque utiliza muchos recursos de la computadora es muy didáctico, permite identificar errores de sintaxis en el momento en el cual se está escribiendo un programa y no hasta después de compilarlo.

Traducción de un programa

Los programas escritos en Java pasan a través de tres fases: edición, compilación y ejecución.

- **Edición:** El programa se crea en un editor, este código (código fuente) se guarda en la computadora en un archivo con la extensión **.java.**
- **Compilación:** El compilador (javac.exe) convierte el código fuente a código intermedio (bytecode) y lo guarda en archivos **.class.**
- **Ejecución:** La JVM (*Java Virtual Machine*, Máquina Virtual de Java) carga los .class en memoria, verifica el bytecode, lo traduce a lenguaje máquina y lo ejecuta.



Estructura básica de un programa

En Java la estructura básica de un programa es la siguiente:

```
public class Programa {
public static void main (String args []){
instrucciones;
instrucciones;
}
}
```

La primera línea indica que la clase se llama Programa y que todo lo que esté dentro de la llave que abre “{” y que cierra “}” será parte de ella, de esta manera se puede observar que las llaves marcan el “inicio” y “fin” del programa. La segunda línea del ejemplo corresponde a la instrucción de inicio de ejecución del programa, todo lo que esté dentro de las llaves que abren y cierran de la sentencia **public static void main (String args [])**será ejecutado por el compilador de Java. Finalmente es necesario mencionar que en Java se utiliza el símbolo ;(punto y coma) para separar las instrucciones.

Elementos del lenguaje Java

- **Comentarios:** Es un texto adicional que se añade al código para explicar su funcionalidad o para agregar una nota a manera de recordatorio o explicación.
- **Tipos primitivos de variables:** Se llaman tipos primitivos de variables a aquellas variables que contienen los tipos de datos más habituales y que no requieren invocación para ser creados.
- **Cadenas de caracteres:** las cadenas son objetos de la clase String y se declara e inicializa de la siguiente manera: **String** cadena = "cadena de ejemplo";
- **Definición de variables:**

- **Declarar las variables:** Para declarar una variable se debe definir primero su tipo y después su nombre, para lo cual se utiliza la siguiente sintaxis: **tipo_variable nombre_variable;**
- **Declarar e inicializar las variables:** Para declarar una variable e inicializarla es necesario definir el tipo de la variable, seguida de su nombre y un signo de =(igual) para finalmente escribir el valor que guardará inicialmente la variable, la sintaxis es la siguiente: **tipo_variable nombre_variable = valor_variable;**

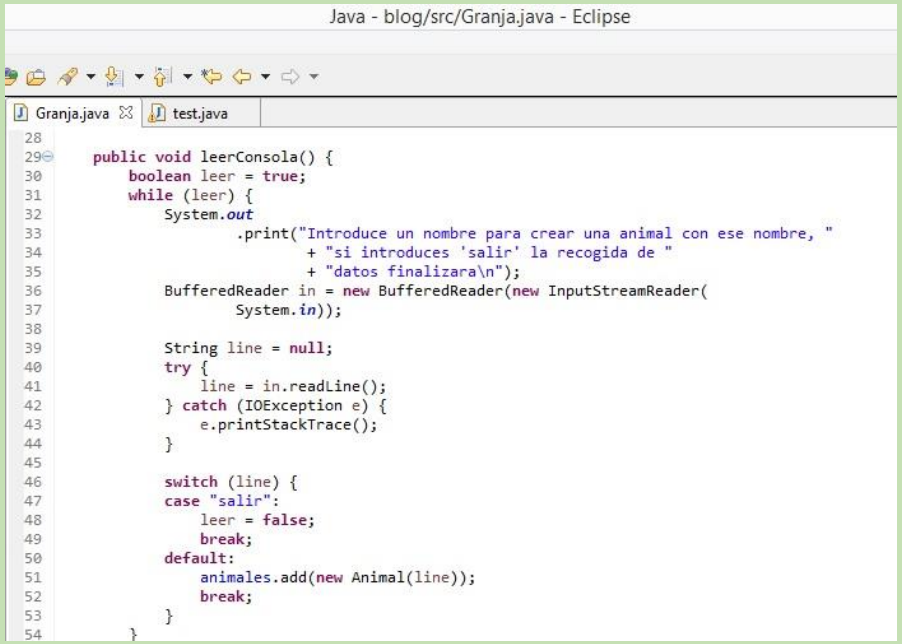
Elementos del lenguaje Java

- **Identificadores constantes:** Las constantes son aquellas que una vez que se les asigna un valor, éste no puede ser modificado. Para definir constantes en Java sólo es necesario anteponer a la declaración la palabra **final**, por ejemplo: **final float** PI = 3.1416f;
- **Operadores:** Como su nombre lo indica, los operadores permiten realizar operaciones en el lenguaje de programación. En Java existen diferentes tipos de operadores:

- **Operadores ariméticos:** Son operadores binarios (requieren siempre de dos operandos) que realizan las operaciones aritméticas habituales: suma (+), resta (-), multiplicación (*), división (/) y el módulo (%).
- **Operadores de asignación:** La asignación consiste en trasferir un valor o expresión a una variable, para lo cual se utiliza el signo de igualdad “**=**” y su forma general es: **variable = valor o expresión**.

Salida de datos

Los programas requieren de imprimir información, ya sea para desplegar un mensaje en pantalla, indicándole al usuario la entrada de datos o bien para imprimir textos y números obtenidos de bases de datos o cálculos que se realizaron en el transcurso de la ejecución del programa. Los métodos más sencillos para desplegar información en java son: **System.out.print( )** y **System.out.println( ).** El hábil manejo de los parámetros en combinación con estas instrucciones, permite obtener mejores salidas.



Lectura de datos

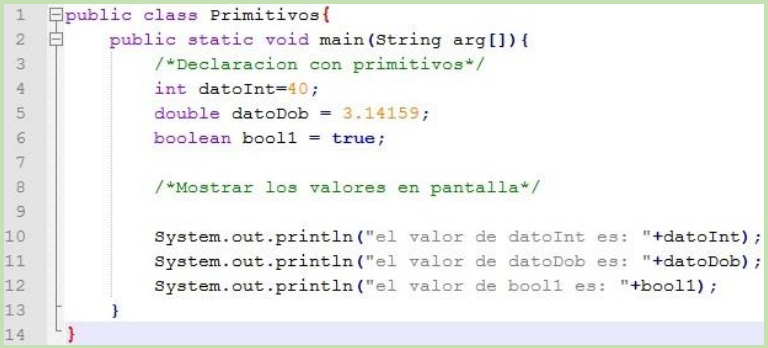
Todos los sistemas necesitan información, esa información puede venir de la pantalla, CD, base de datos, mouse, teclado, entre otras; pero seguramente cuando se trata de un programa la entrada es por medio de asignación o bien por medio del teclado usando para ello el modo consola.Usando la clase **java.util.Scanner** es posible crear una instancia que permite la lectura de información en consola de la siguiente manera.

Scanner leer = **new** Scanner(System.in);

System.out.print ("Dame un número entero: ");

//la sig línea lee una cantidad entera y se la asigna a la variable num

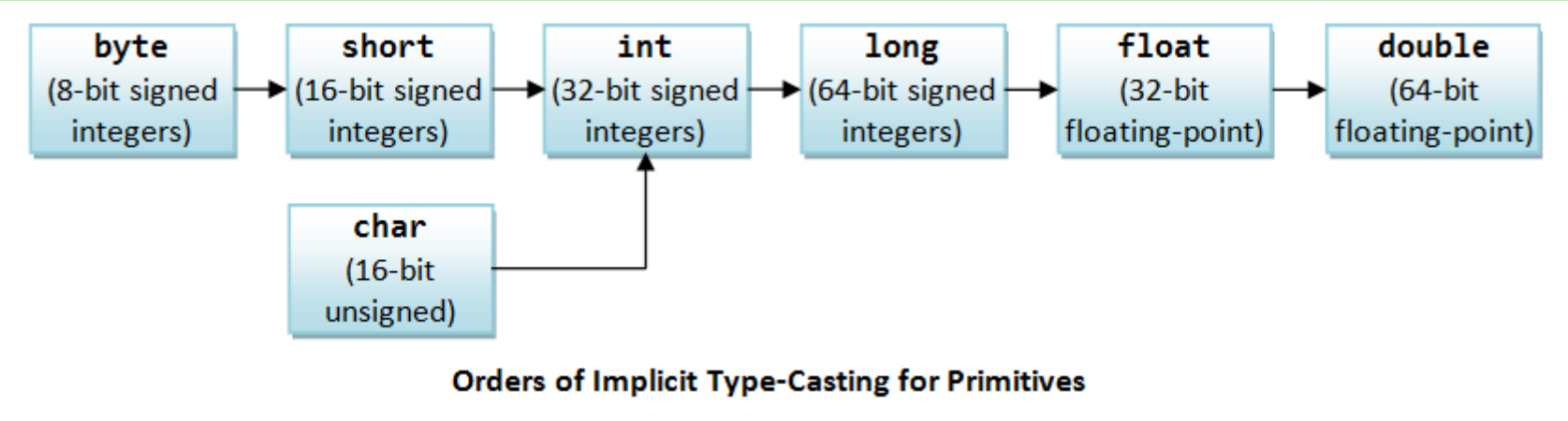
int num = leer.nextInt();



```
1  public class Primitivos{
2      public static void main(String arg[]){
3          /*Declaracion con primitivos*/
4          int datoInt=40;
5          double datoDob = 3.14159;
6          boolean bool1 = true;
7
8          /*Mostrar los valores en pantalla*/
9
10         System.out.println("el valor de datoInt es: "+datoInt);
11         System.out.println("el valor de datoDob es: "+datoDob);
12         System.out.println("el valor de bool1 es: "+bool1);
13     }
14 }
```

Conversión de tipo de datos

Cuando se programa es muy común tener que convertir de un tipo de dato a otro, en Java se pueden hacer las siguientes conversiones de tipos primitivos entre variables. Es posible realizar promociones no válidas (no incluidas en la tabla anterior), pero estas pueden producir errores de truncamiento o pérdida de información, como por ejemplo convertir de un **float** a un **int**. Para realizar la conversión de los tipos primitivos es necesario anteponer entre paréntesis el tipo al cual se desea convertir una variable o valor.



Orders of Implicit Type-Casting for Primitives

Clase math

La clase Math proporciona una colección de métodos que permiten realizar cálculos matemáticos comunes. No es necesario importar la clase Math porque se importa automáticamente por el compilador.

Funciones Matemáticas	Significado	Ejemplo de uso	Resultado
abs	Valor absoluto	int x = Math.abs(2.3);	x = 2;
atan	Arcotangente	double x = Math.atan(1);	x = 0.78539816339744;
sin	Seno	double x = Math.sin(0.5);	x = 0.4794255386042;
cos	Coseno	double x = Math.cos(0.5);	x = 0.87758256189037;
tan	Tangente	double x = Math.tan(0.5);	x = 0.54630248984379;
exp	Exponenciación neperiana	double x = Math.exp(1);	x = 2.71828182845904;
log	Logaritmo neperiano	double x = Math.log(2.7172);	x = 0.99960193833500;
pow	Potencia	double x = Math.pow(2.3);	x = 8.0;
round	Redondeo	double x = Math.round(2.5);	x = 3;
random	Número aleatorio	double x = Math.ramdom();	x = 0.20614522323378;

Errores en tiempo de ejecución

- El nombre de la clase y el nombre del archivo deben tener el mismo nombre.
- Dependiendo del idioma de NetBeans IDE, los números con decimales leídos de consola se deberán introducir con “,” (coma) o “.” (punto) para separar la parte entera de la decimal.
- Si se desea leer una línea después de un nextDouble(), nextInt() o nextFloat() es necesario el uso de dos nextLine().

Java incorpora la gestión de excepciones, las excepciones indican que existe un problema o situación inesperada durante la ejecución de un programa. El proceso de gestión de excepciones consiste en atrapar y manejar los errores que ocurren durante la ejecución.