**Java 20 – 2019 JSE EIT - Métodos**

****

Hay métodos que retornan un valor (uno solo), y tienen en su enunciado un tipo, como por ejemplo un tipo de variables, un **String** u otro objeto.

Estos métodos para retornar un valor, deben usar al menos una instrucción “**return**”.

Hay otros **métodos** que no devuelven un valor, y en su enunciado se especifica “**void**”, por lo que no requieren de la instrucción “**return**”.

Hay variables que son “**locales**”, y son definidas dentro de un **método**.

La vida de estas variables es dentro del ámbito del **método** en el que fueron creadas.

Si las variables son definidas fuera de un **método**, son llamadas “**globales**” y su valor se mantiene aún terminado cualquiera de los **métodos** que la usen, sin perder su valor.

Otra definición: una “**caja negra**” es un módulo que resuelve un único problema, dejando “oculto” el cómo realiza su trabajo.

Podemos distinguir entre **métodos** creados por el programador, y los suministrados por **Java** (por ejemplo: “**Math.sqrt()”**).

Los usos de los **métodos** son múltiples:

* Para evitar escribir el mismo programa más de una vez.
* Para trabajar en equipo: cada programador hace sus **métodos**.
* Su uso correcto hace que sea más fácil la detección de errores.
* Modularidad: se pueden usar **métodos** ya hechos y probados, que funcionen en otras aplicaciones.

Veamos un ejemplo, en el que usamos el método al que llamamos “**Lee()**” para leer un número:

**package hm;**

**import javax.swing.\*;**

**public class HolaMundo {**

**public static void main( String args[] ) {**

**double a, b, hipotenusa;**

**a = Lee(); // "Lee" sirve para leer un valor.**

**// No pasa valores, por lo que los paréntesis**

**//están vacíos.**

**b = Lee(); // "Lee" sirve para leer en este caso otro valor.**

**// No pasa valores, por lo que los paréntesis**

**//están vacíos.**

**hipotenusa = Math.sqrt( a\*a + b\*b ); // "Math.sqrt" pertenece**

**//a la biblioteca "java.lang".**

**JOptionPane.showMessageDialog( null, "Respuesta: " + hipotenusa );**

**}**

**static double Lee() { // No recibe valores, por lo que los**

**//paréntesis están en vacíos.**

**// Sin embargo, "double" significa que**

**//devuelve un valor de este tipo usando**

**//la instrucción "return".**

**String s;**

**double x;**

**s = JOptionPane.showInputDialog( "Ingrese " );**

**x = Double.parseDouble( s );**

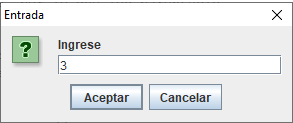
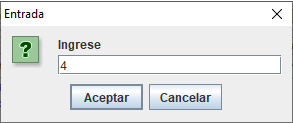
**return x; // Esta es la instrucción que devuelve el valor de**

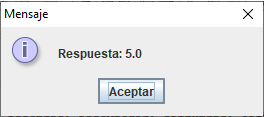
**//tipo "double".**

**}**

**}**

Una posible ejecución y salida del ejercicio es:



En este otro ejemplo pasamos un argumento al método, para que después lo mostremos:

**package hm;**

**import javax.swing.\*;**

**public class HolaMundo {**

**public static void main( String args[] ) {**

**double a, b, hipotenusa;**

**a = Lee( 'a' ); // "Lee" sirve para leer un número.**

**// Se pasa un valor, que es el carácter 'a'.**

**b = Lee( 'b' ); // "Lee" sirve para leer otro número.**

**// Se pasa un valor, que es el carácter 'b'.**

**hipotenusa = Math.sqrt( a\*a + b\*b );**

**JOptionPane.showMessageDialog(**

**null,**

**"Respuesta: " + hipotenusa**

**);**

**}**

**static double Lee( char nombre ){ // Recibe un valor de tipo**

**//"char" y se lo asigna a**

**//"nombre".**

**// "double" significa que**

**//devuelve un valor de este**

**//tipo usando la instrucción**

**//"return".**

**String s;**

**double x;**

**s = JOptionPane.showInputDialog( // Mostramos el valor recibido.**

**"Ingrese \"" +**

**nombre + "\""**

**);**

**x = Double.parseDouble( s );**

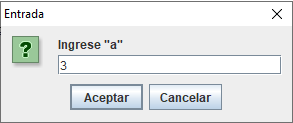
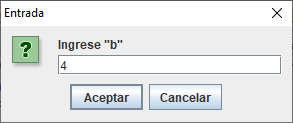
**return x; // Esta es la instrucción que devuelve el valor de**

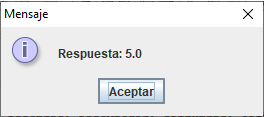
**//tipo "double".**

**}**

**}**

Una posible ejecución y salida del ejercicio es:



Crearemos otro método, llamado “**hipo**” que recibe dos parámetros: “**a**” y “**b**”:

**package hm;**

**import javax.swing.\*;**

**public class HolaMundo {**

**public static void main(String args[]) {**

**double a, b, hipotenusa;**

**a = Lee( 'a' );**

**b = Lee( 'b' );**

**hipotenusa = hipo( a , b ); //**

**JOptionPane.showMessageDialog( null,**

**"Respuesta: " + hipotenusa**

**);**

**}**

**static double Lee( char nombre ) {**

**String s;**

**double x;**

**s = JOptionPane.showInputDialog("Ingrese \"" + nombre + "\"" ) ;**

**x = Double.parseDouble(s);**

**return x;**

**}**

**static double hipo( double a , double b ) {**

**double h ;**

**h = Math.sqrt( a\*a + b\*b );**

**return h;**

**}**

**}**

También pudimos haber puesto:

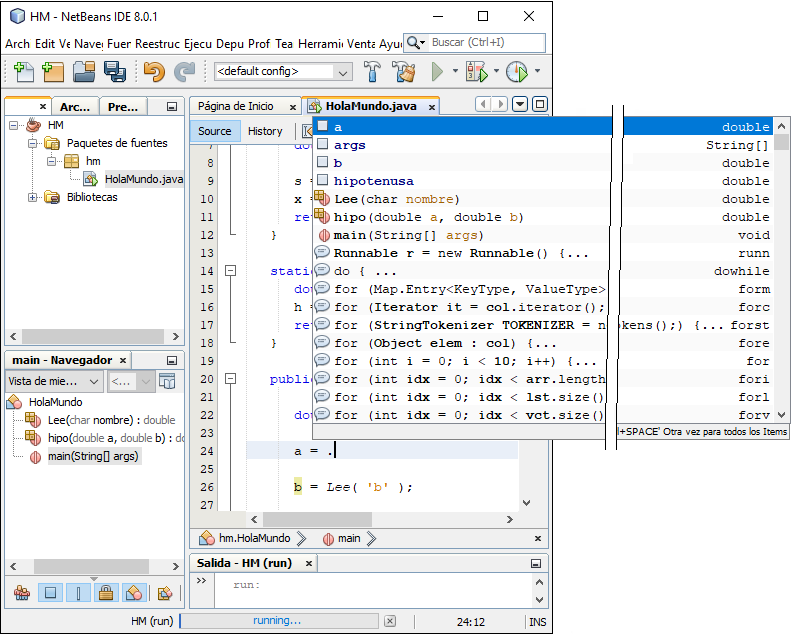
**static double hipo( double a , double b ) {**

**return Math.sqrt( a\*a + b\*b );**

**}**

Supongamos que tenemos ingresados los métodos y propiedades, escribimos un punto en un método…

…entonces los podemos ver en la ayuda que nos brinda el **IDE**.



Ahora agregaremos un método “**cuadrado**” que es llamado por otro, que es “**hipo**”:

**package hm;**

**import javax.swing.\*;**

**public class HolaMundo {**

**public static void main(String args[]) {**

**double a, b, hipotenusa;**

**a = Lee( 'a' );**

**b = Lee( 'b' );**

**hipotenusa = hipo( a , b );**

**JOptionPane.showMessageDialog( null, "Respuesta: " + hipotenusa );**

**}**

**static double Lee( char nombre) {**

**String s;**

**double x;**

**s = JOptionPane.showInputDialog("Ingrese \"" + nombre + "\"" ) ;**

**x = Double.parseDouble(s);**

**return x;**

**}**

**static double hipo( double a, double b ) {**

**return Math.sqrt( cuadrado( a ) + cuadrado( b ) );**

**}**

**static double cuadrado( double z ) {**

**return z\*z;**

**}**

**}**

Entonces, tenemos un método principal “**main**” que llama al método “**hipo**”, y éste a su vez llama al método “**cuadrado**”.

Otro ejercicio:

Ingrese un número y calcule su cuadrado.

**package hm;**

**import javax.swing.\*;**

**public class HolaMundo {**

**public static void main(String args[]) {**

**double a;**

**a = cuadrado( Lee( 'a' ) );**

**JOptionPane.showMessageDialog(null, "Respuesta: " + a );**

**}**

**static double Lee( char nombre ) {**

**String s;**

**double x;**

**s = JOptionPane.showInputDialog( "Ingrese \"" + nombre + "\"" ) ;**

**x = Double.parseDouble( s );**

**return x;**

**}**

**static double cuadrado( double z ) {**

**return z\*z;**

**}**

**}**

Igual al anterior, pero usando la variable global “**nombre**”:

**package hm;**

**import javax.swing.\*;**

**public class HolaMundo {**

**static char nombre='a';**

**public static void main(String args[]) {**

**double a;**

**a = cuadrado( Lee() );**

**JOptionPane.showMessageDialog(null, "Respuesta: " + a );**

**}**

**static double Lee( ) {**

**String s;**

**double x;**

**s = JOptionPane.showInputDialog("Ingrese \"" + nombre + "\"" ) ;**

**x = Double.parseDouble(s);**

**return x;**

**}**

**static double cuadrado( double z ) {**

**return z\*z;**

**}**

**}**

Ahora hacemos un programa que:

1. Lee “**n**” a través del método “**Lee**”, para generar el orden del vector,

2. carga de datos el vector con el método “**cargaVector**”,

3. ordena el vector con el método “**ordenaVector**”, y

4. muestra el vector con el método “**muestraVector**”.

**package hm;**

**import javax.swing.\*;**

**public class HolaMundo {**

**private static int MiVector[];**

**public static void main(String args[]) {**

**int n = Lee();**

**MiVector = new int[n];**

**cargaVector( n );**

**ordenaVector( n );**

**muestraVector( n );**

**}**

**static int Lee() {**

**String s;**

**int x;**

**s = JOptionPane.showInputDialog("Ingrese \"N\"" ) ;**

**x = Integer.parseInt(s);**

**return x;**

**}**

**static void cargaVector( int n ) {**

**int i;**

**for( i=0 ; i<n ; i++ ) {**

**s = JOptionPane.showInputDialog("Ingrese MiVector" ) ;**

**MiVector[i] = Integer.parseInt(s);**

**}**

**}**

**static void ordenaVector( int n ) {**

**int i, j, Auxiliar;**

**String s;**

**for( i=0 ; i<n-1 ; i++ ) {**

**for( j=i+1 ; j<n ; j++ ) {**

**if( MiVector[i] < MiVector[j] ) {**

**Auxiliar = MiVector[j];**

**MiVector[j] = MiVector[i];**

**MiVector[i] = Auxiliar;**

**}**

**}**

**}**

**}**

**static void muestraVector( int n ) {**

**int i;**

**for( i=0 ; i<n ; i++ ) {**

**System.out.println( MiVector[i] );**

**}**

**}**

**}**

Para ver los 3 mayores valores, reemplazamos el último método del programa anterior por:

**static void muestraVector( int n ) {**

**int i;**

**for( i=0 ; i<3 ; i++ ) {**

**System.out.println( MiVector[i] );**

**}**

**}**