Objetos

```
Clase NombreObjeto = new Clase(parametros);
```

Clase Random

`nextInt(`**`max`**`)`: entero aleatorio entre o y max - 1

Clases Wrapper (Integer, Double, Boolean, Char)

```
Integer.parseInt(String s)
Double.parseDouble(String s)
Integer.toString(int num)
Double.toString(double num)
```

Clase String

`contains(`**`str`**`)`: true si la cadena contiene los caracteres de la otra

`endsWith(`**`str`**`), startsWith(`**`str`**`)`: true si la cadena empieza o acaba con los caracteres de la otra

`equals(`**`str`**`)`: true si una cadena es igual a la otra

`equalsIgnoreCase(`**`str`**`)`: true si una cadena es igual a la otra, sin tener en cuenta mayúsculas

`indexOf(`**`str`**`)`: índice en la cadena de caracteres por el que empieza el string dado (-1 si no está)

`length()`: número de caracteres del string

`substring(`**`i, j`**`)`: caracteres que van desde el índice i (incluido) hasta j (excluido)

`substring(`**`i`**`)`: caracteres que van desde el índice i (incluido) hasta el final

`toLowerCase(), toUpperCase()`: string con todas sus letras en minúsculas o en mayúsculas

`charAt(`**`i`**`)`: carácter con índice i

```
String.format("%s, %d, %f", "hola", 5, 5.5)
```

Clase Scanner

`nextInt(), hasNextInt()`: lee un dato como int y comprueba que un dato se puede leer como int

`next(), hasNext()`: lee un dato como String y comprueba que es posible leerlo

`nextDouble(), hasNextDouble()`: lee un dato como double y comprueba que es posible leerlo

`nextLine(), hasNextLine()`: lee una línea como String y comprueba que es posible leerla

Clase Math

`Math.abs(`**`valor`**`)`: valor absoluto

`Math.min(`**`v1, v2`**`)`: valor menor de los 2 dados

`Math.max(`**`v1, v2`**`)`: valor mayor de los 2 dados

`Math.round(`**`valor`**`)`: número entero más cercano

`Math.pow(`**`b, e`**`)`: b elevado a e

Clase PrintStream

```
println(texto);
print(texto);
printf("%-30s %d %.2f", "HOLA", 12, 3.1416)
```

Clase File

`canRead()`: true si el fichero existe y se puede leer

Arrays

```
Clase[] nombre = new Clase[longitud];
Clase[] nombre = {VAL1, VAL2, VAL3, ...};
nombre[indice] = valor;
nombre.length  // Número de elementos
Arrays.toString(array)
Arrays.sort(array)
```

ArrayList

```
ArrayList <Clase> lista = new ArrayList <Clase>();
Clase valor = lista.get(i)
lista.set(i, valor);
lista.size()        // Número de elementos
lista.add(valor);
lista.add(i, valor);
lista.remove(i)
```

For-each

```
for (Clase elemento : lista) {
   ...
}
```

Iterador

```
Iterator<Clase> nombre = lista.iterator();
nombre.hasNext();
nombre.next();
nombre.remove();
```

Clases

```
public class ClaseHija extends ClasePadre implements intefaz {
   // Atributos private / protected

   // Constructor

   // Métodos heredados
   @Override

   // Resto de métodos
}
```

Polimorfismo

objeto instanceof clase: Devuelve true si objeto es de la clase indicada. False en caso contrario

(ClaseHija) objetoClasePadre: Realiza casting para acceder a los métodos de la clase hija

JoptionPane

```
showMessageDialog()
showInputDialog()
showConfirmDialog()      // YES_OPTION, NO_OPTION CANCEL_OPTION
```

JFrame – JPanel - JDialog

```
setTitle(String titulo)
setLocation(int x, int y)
setSize(int ancho, int alto)
setLayout(LayoutManager manager)
setDefaultCloseOperation(int operacion)
// Valores para operación: DO_NOTHING_ON_CLOSE, HIDE_ON_CLOSE, DISPOSE_ON_CLOSE, EXIT_ON_CLOSE
pack()
setVisible(boolean valor)
add(componente)
setJMenuBar(JMenuBar)
```

Layout

```
FlowLayout
BorderLayout      // NORTH, SOUTH, WEST, EAST, CENTER
GridLayout
```

Componentes

```
Jbutton(String texto): setText(texto), getText(), setBackground(Color.BLUE)
Jlabel(String texto): setText(texto), getText()
JtextField(int numCaracteres): setText(texto), getText(), setInputVerifier(InputVerifier)
JtextArea(int filas, int cols): setText(texto), appendText(texto), getText()
JcomboBox<Clase>(array): getSelectedItem()
JMenuBar - JMenu(texto) - JmenuItem(texto): add
addActionListener(ActionListener)
```

Action Listener: public void actionPerformed(ActionEvent evento)

```
ActionEvent: getSource()      // Devuelve el objeto pulsado
```

BBDD

`Class.forName("oracle.jdbc.driver.OracleDriver")`

`Connection: DriverManager.getConnection("jdbc:oracle:thin:@dirIP:xe", usuario, password)`

`close()`

Statement: conexion.createStatement()

`ResultSet executeQuery(consultaSQL): next(), getTipo(posicion / nombreColumna)`

`int executeUpdate(consultaSQL)`

PreparedStatement: conexion.prepareStatement(consultaSQL);

`setTipo(posicion, valor)`

`ResultSet executeQuery(): next(), getTipo(posicion / nombreCol), ResultSetMetaData getMetaData()`

`int executeUpdate()`

DatabaseMetaData: conexion.getMetaData();

`getDriverVersion()`

`getDatabaseProductName()`

`getDatabaseProductVersion()`

ResultSetMetaData

`getColumnCount()`

`getColumnLabel(posicion)`

`getColumnDisplaySize(posicion)`

Excepciones (FileNotFoundException, IOException, SQLException, ClassNotFoundException)

`throws`

`try - catch - finally`

`getMessage(), printStackTrace(), toString()`