**Clase Math**

int x = Math.abs(2.3); x = 2;

double x = Math.atan(1); x = 0.78539816339744;

double x = Math.sin(0.5); x = 0.4794255386042;

double x = Math.cos(0.5); x = 0.87758256189037;

double x = Math.tan(0.5); x = 0.54630248984379;

double x = Math.exp(1); x = 2.71828182845904;

double x = Math.log(2.7172); x = 0.99960193833500;

double x = Math.pow(2.3); x = 8.0;

double x = Math.round(2.5); x = 3;

double x = Math.ramdom(); x = 0.20614522323378;

int x = Math.ceil(2.3) x = 3;

int x = Math.floor(2.3) x = 2;

**Clase String**

String cadena="El que se fue a Sevilla perdio su silla y el que se fue al Torreon, su sillon";

System.out.println(cadena.charAt(0)); // Nos devolvera E

System.out.println(cadena.charAt(11)); //Nos devolvera u

System.out.println(cadena.endsWith("n")); //Nos devuelve true

System.out.println(cadena.startsWith("e")); //Nos devuelve false, Java distingue entre mayusculas y minusculas

System.out.println(cadena.equals("El que se fue a Sevilla perdio su silla y el que se fue al Torreon, su sillon")); //Nos devuelve true

System.out.println(cadena.indexOf("fue")); //localiza la posicion donde se encuentra "la"

System.out.println(cadena.length()); // longitud: 77

System.out.println(cadena.replace('a', 'e')); // Cambia todas las a por e

System.out.println(cadena.toLowerCase()); //Transforma el String a minusculas

System.out.println(cadena.toUpperCase()); //Transforma el String a mayusculas.

compareTo/compareToIgnoreCase devuelve num>0 primera cadena mayor, 0 iguales y num<0 primera cadena menor

lastIndexOf como indexOf pero desde el final.-1 si no existe. Podemos añadir un parámetro int con la posición de inicio.

trim elimina los espacion del principio y del final

valueOf variable primitiva a String

**Herencia**

**Uso de super**

public class Disco extends Material{

String discografica;

Disco(int codigo, String titulo, String autor, String discografica){

super(codigo, titulo, autor); //constructor

this.discografica = discografica;

}

public void ver(){

super.ver(); //método ver clase padre

System.out.print(" Discografica: "+discografica);

}

}

**Polimorfismo**

class Instrumento {

public void tocar() {

System.out.println("Instrumento.tocar()");

}

public String tipo() {

return "Instrumento";

}

public void afinar() {}

}

class Guitarra extends Instrumento {

public void tocar() {

System.out.println("Guitarra.tocar()");

}

public String tipo() { return "Guitarra"; }

public void afinar() {}

}

class Piano extends Instrumento {

public void tocar() {

System.out.println("Piano.tocar()");

}

public String tipo() { return "Piano"; }

public void afinar() {}

}

**Clases abstractas**

abstract class Vehiculo

{

public abstract void arrancar();

}

class Coche extendas Vehiculo

{

public void arrancar(String s)

{

System.out.println("El coche arranca con " + s);

}

// sobrescribiendo arrancar

public void arrancar()

{

System.out.println("Arracando");

}

}

**Interfaces**

interface InstrumentoMusical {

void tocar();

void afinar();

String tipoInstrumento();

}

class InstrumentoViento extends Object implements InstrumentoMusical {

void tocar() { . . . };

void afinar() { . . .};

String tipoInstrumento() {}

}

class Guitarra extends InstrumentoViento {

String tipoInstrumento() {

return "Guitarra";

}

}

**Visibilidad**

MODIFICADOR CLASE PACKAGE SUBCLASE TODOS

public Sí Sí Sí Sí

protected Sí Sí Sí No

No especificado Sí Sí No No

private Sí No No No