**Clase 1**

**Java**

Main = “Programa principal”. { } delimita el cuerpo.

Sentencias de código separadas por punto y coma (;).

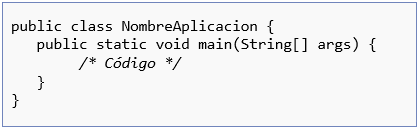
Se recomienda indentar el código para facilitar su lectura.

Comentarios:

De líneas múltiples /\* Esto es un comentario \*/.

De línea única // Este es un comentario

Case-sensitive (sensible a las mayúsculas y minúsculas)



**Variables**

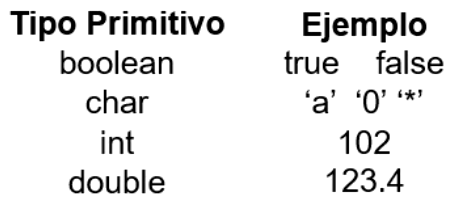
Se declaran en zona de código (no toman valor por defecto).

Tipo nombreVariable; (Opcional: dar valor inicial)

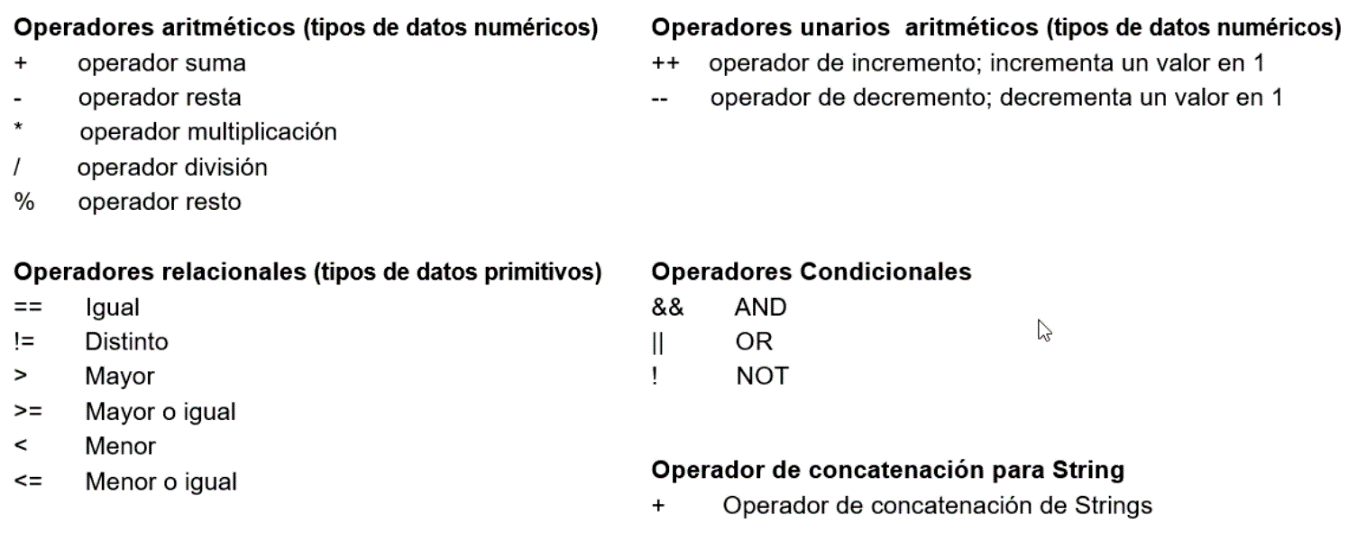
Convención de nombres: comenzar con minúscula, luego cada palabra en mayúscula (CamelCase).

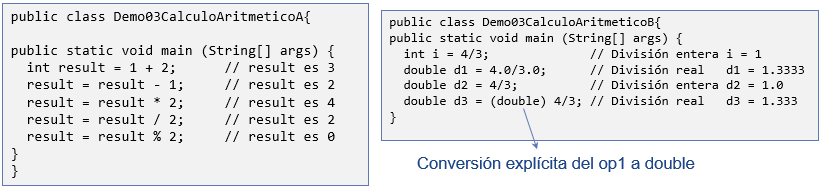
Asignación: nombreVariable = valor;

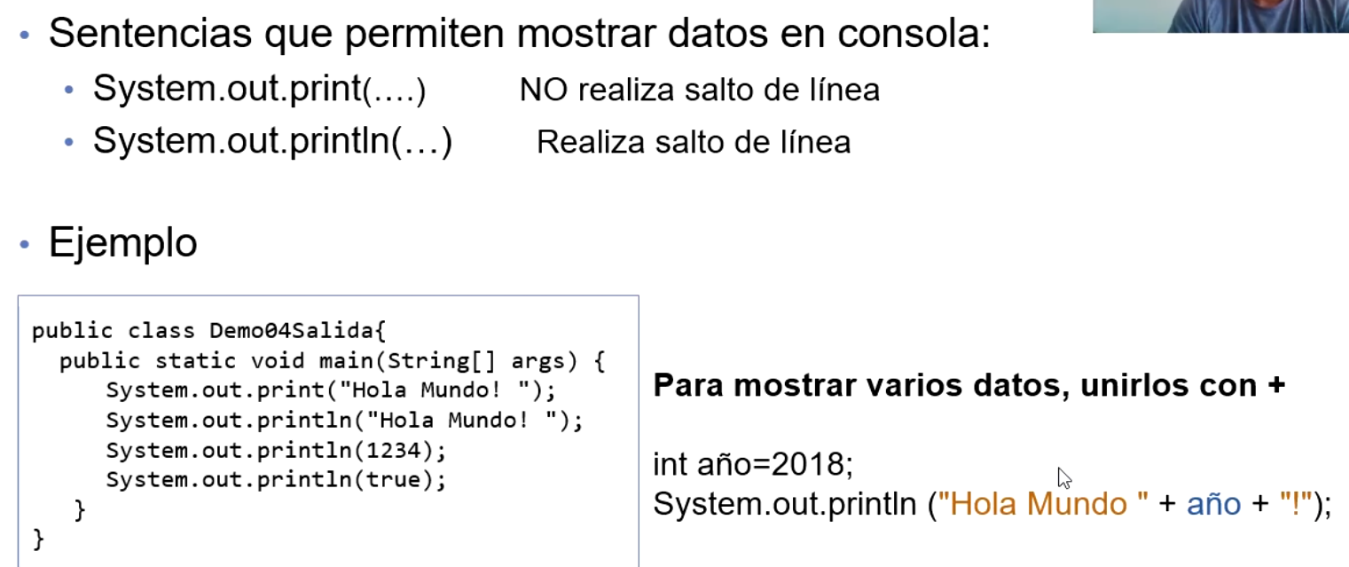
Tipos primitivos: la variable almacena un valor

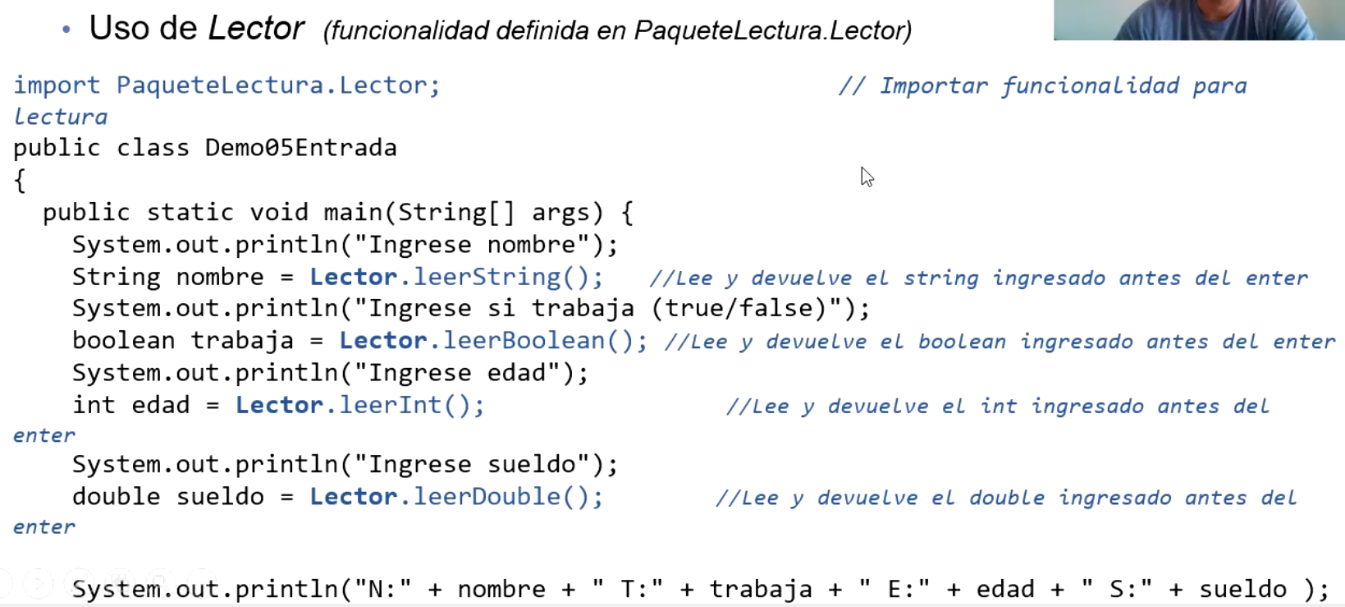


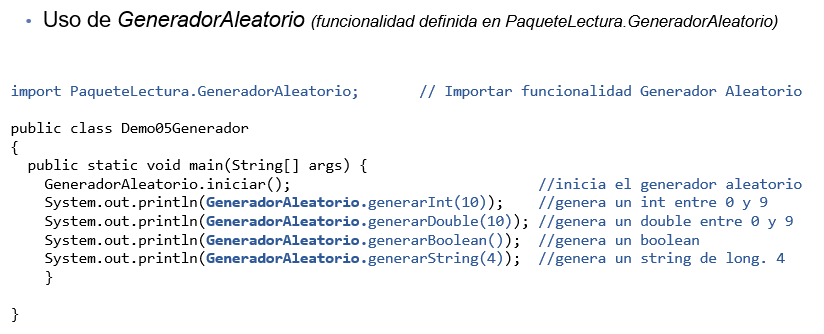
String para manipular cadenas. Ejemplo “esto es un string”.

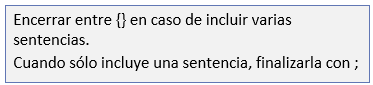
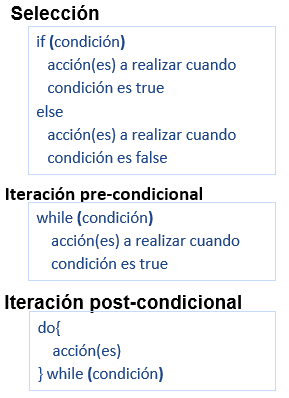


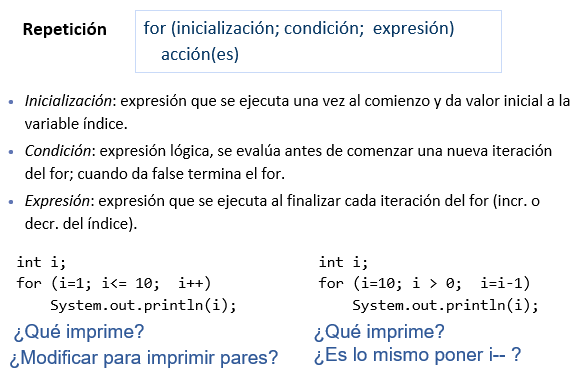


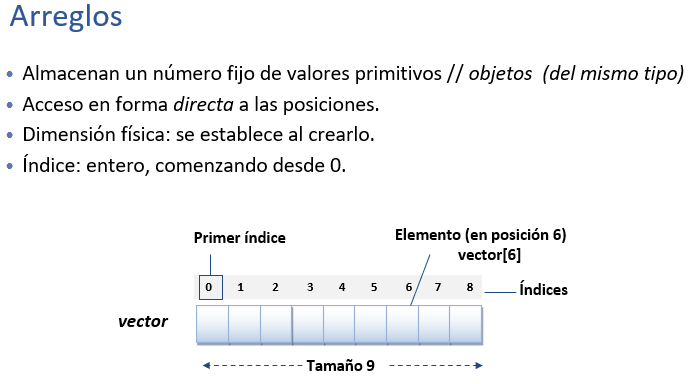




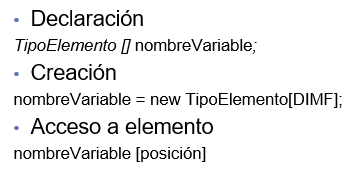


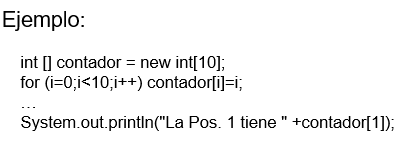
(en el if)

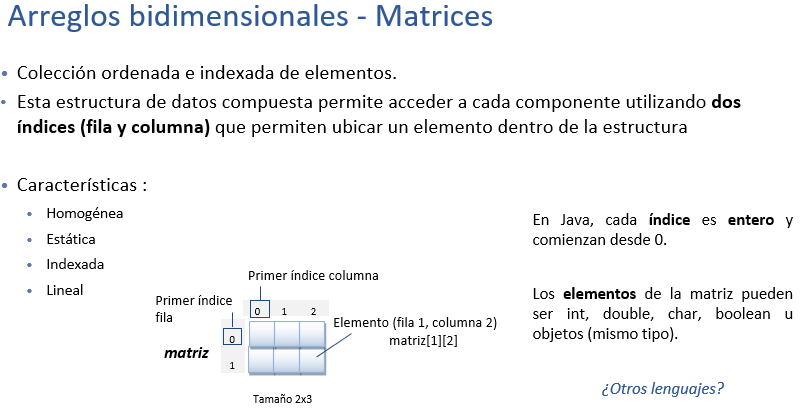


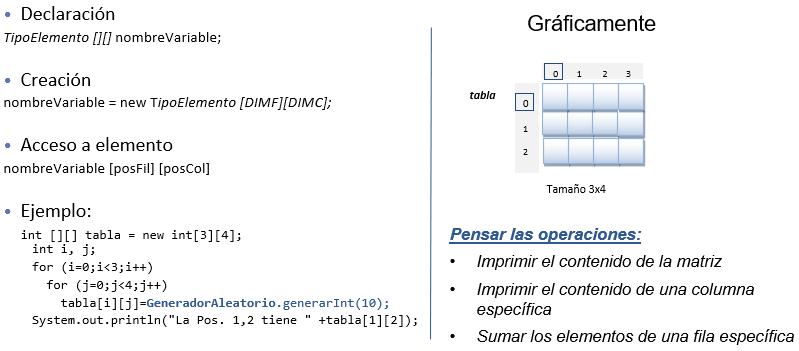


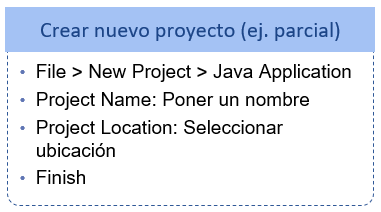


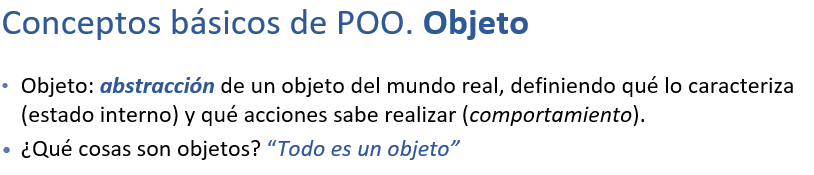




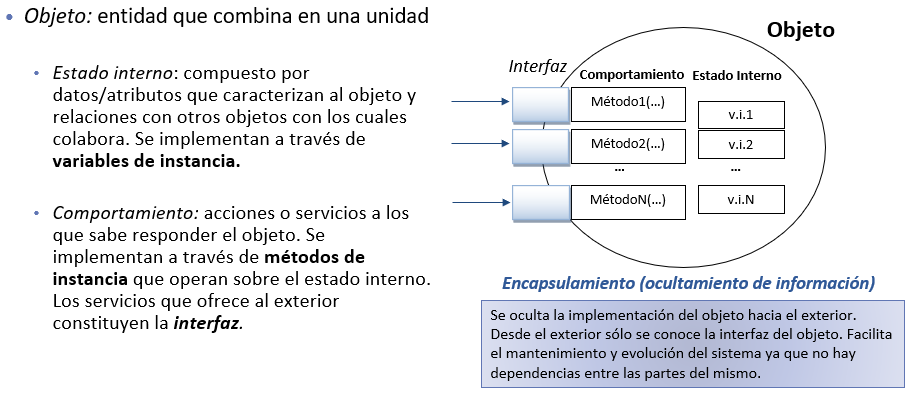


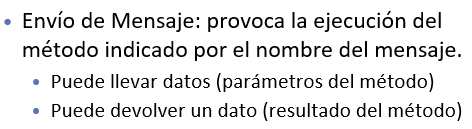


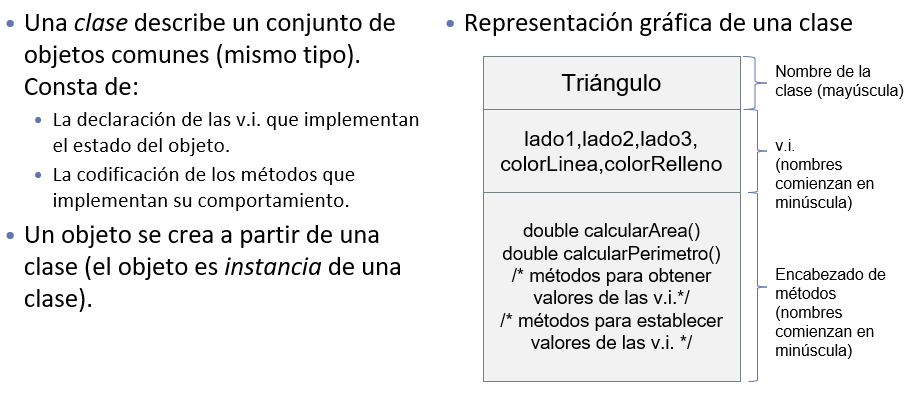


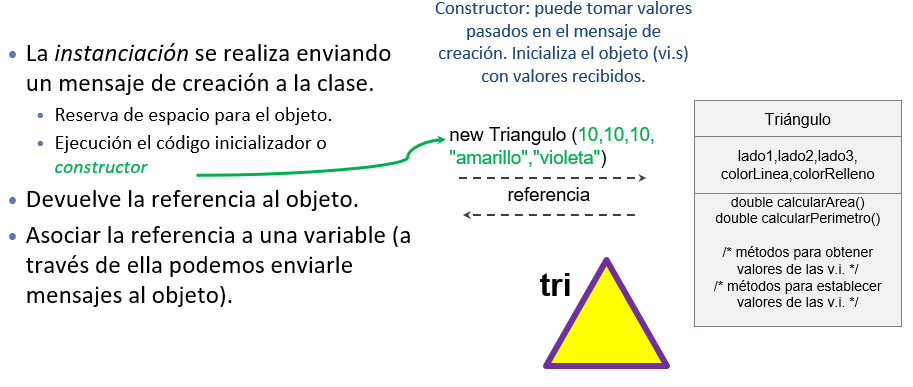


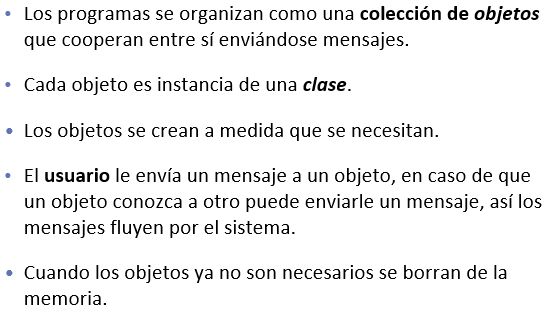


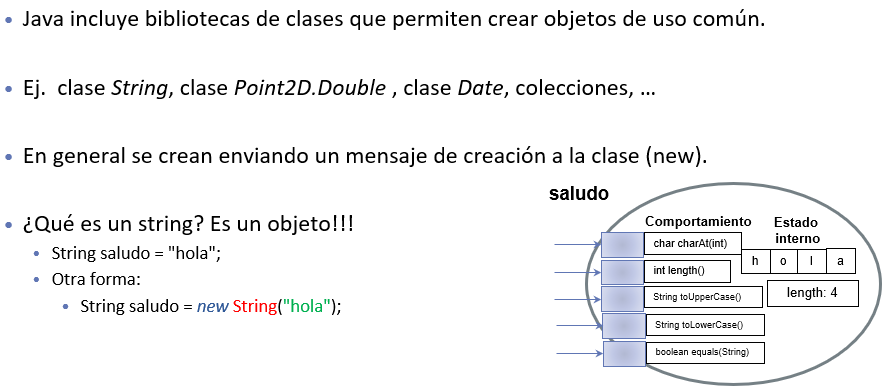


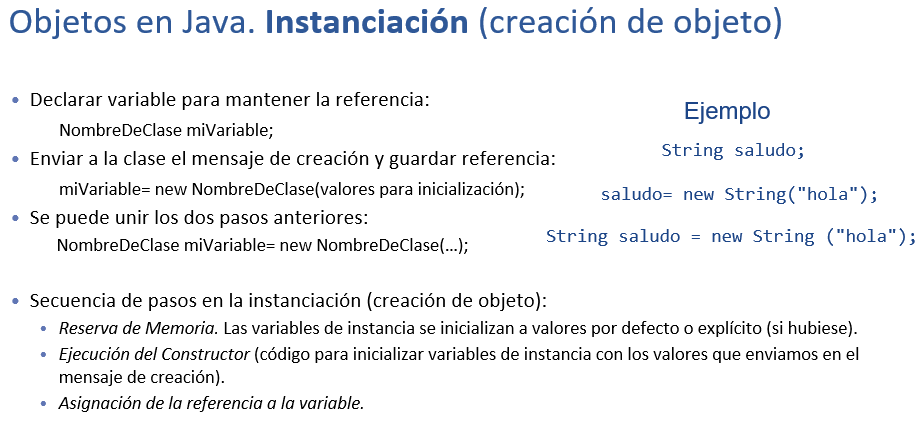




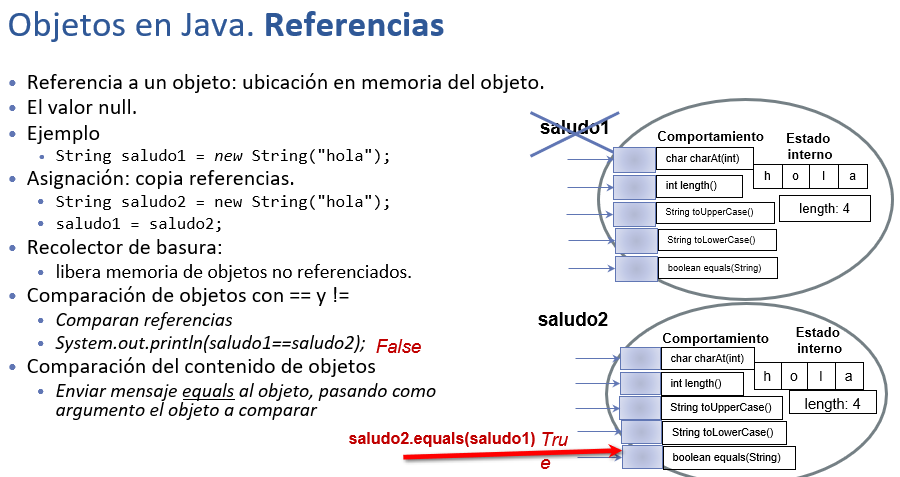


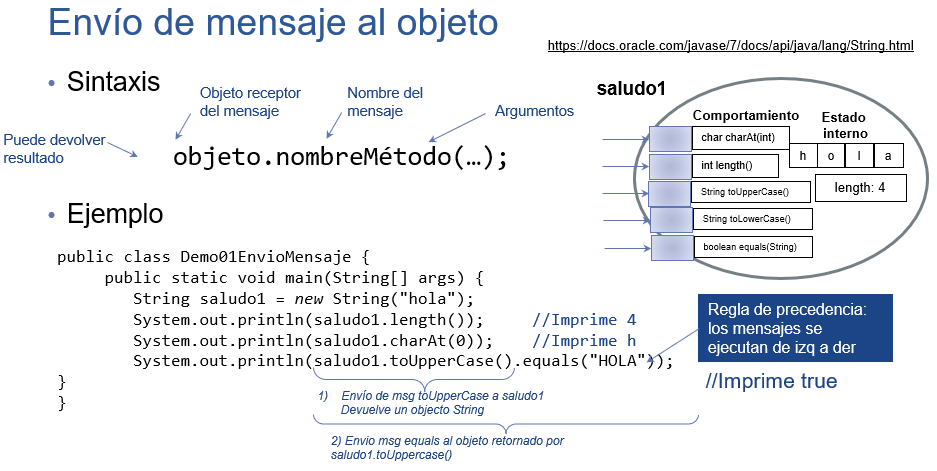




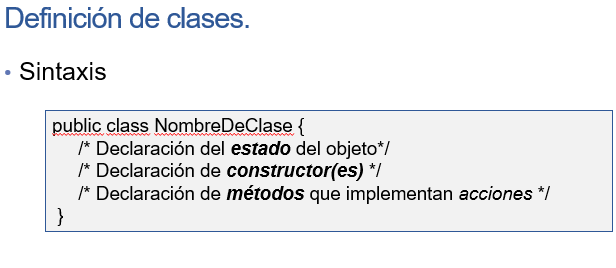


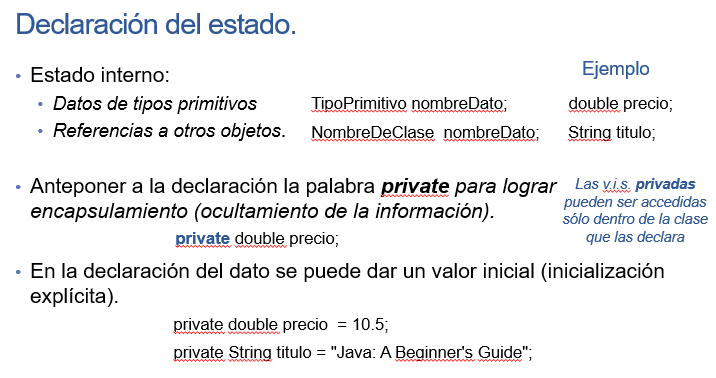
Si queda algún objeto que no esta referenciado por ninguna variable, tenemos un recolector de basura que se encarga de limpiar ese objeto que queda sin ser referenciado y sin poder acceder.

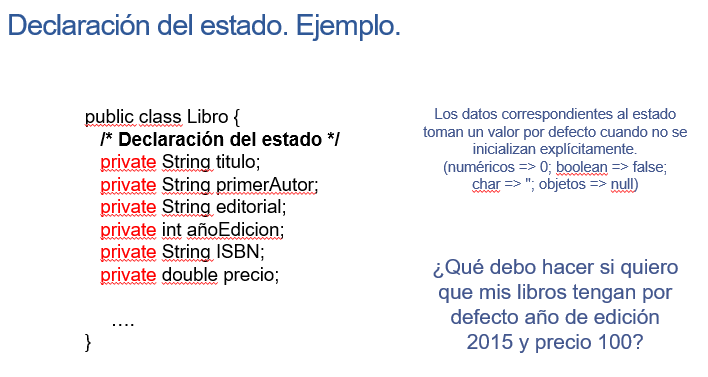


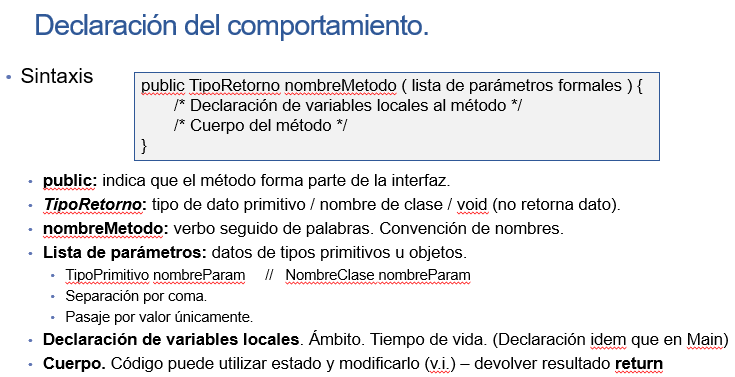


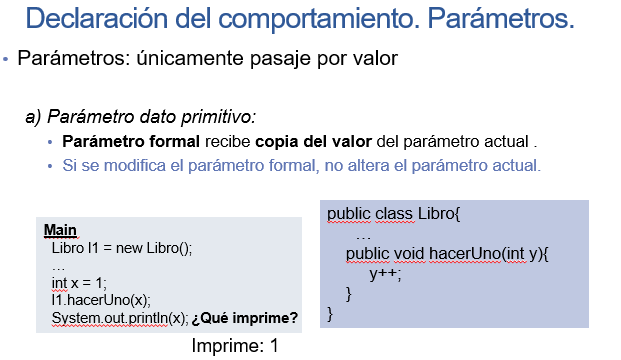
**Clase 2**



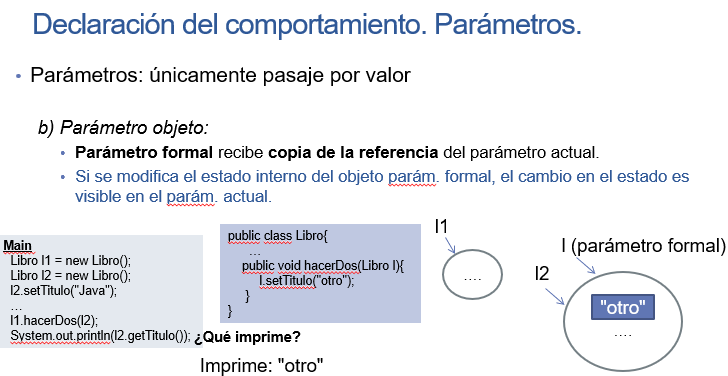


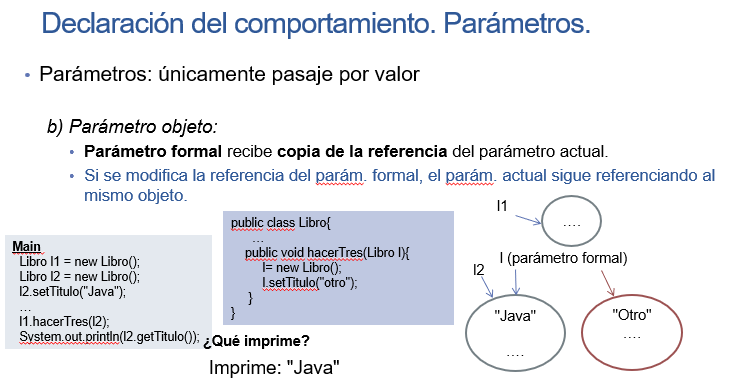






No se modifica porque solamente se pueden pasar parámetros por valor, por mas que se modifique en hacer uno no se modifica en el main.





Imprime java porque esta modificación la hicimos sobre la nueva instancia creada ‘l’.

Instanciación (creación de objetos)

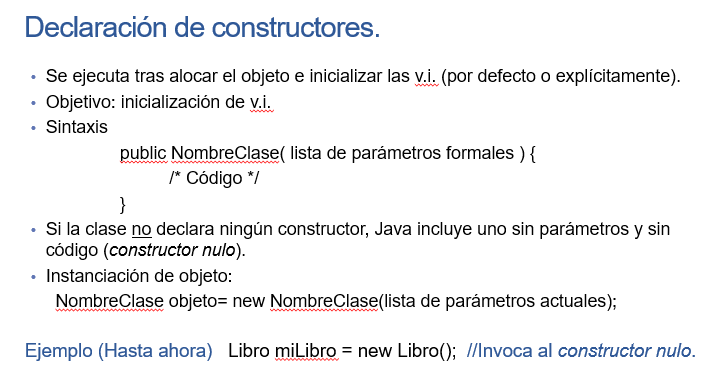
Secuencia de pasos en la creación:

Reserva de memoria. Las variables de instancia se inicializan a valores por defecto o explicito (si hubiese).

Ejecución del constructor (código para inicializar variables de instancia con los valores que enviamos en el mensaje de creación).

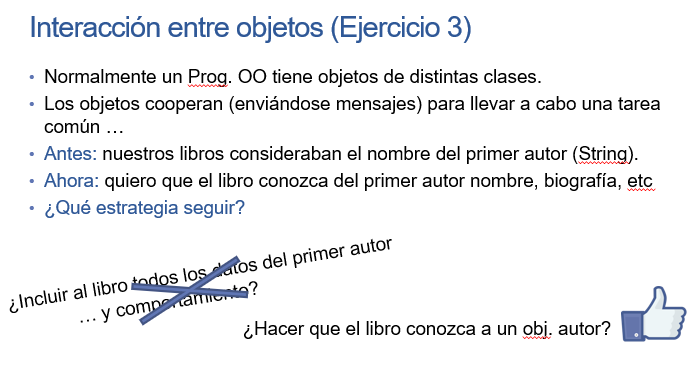
Asignación de la referencia a la variable.

**Segunda parte**



Si el programador genero un constructor, Java no incluye el constructor nulo.

Puede haber varios constructores para la clase (sobrecarga). Java identifica el constructor que esta siendo llamado teniendo en cuenta la cantidad de parámetros y que tipo de parámetros tenga.



La referencia this.

Dentro de una clase puedo disparar la ejecución de un método X desde otro método Y poniendo this.nombreMetodo. El objeto que esta ejecutando (this) se enviara un mensaje a si mismo. El método a ejecutar se busca a partir de la clase de la cual es instancia el objeto. Obviando la palabra this obtenemos el mismo efecto.

Otro uso: para referirse a las variables de instancia del objeto dentro de un método/constructor, que posee parámetros con igual nombre que las variables de instancia del objeto.

Por ej

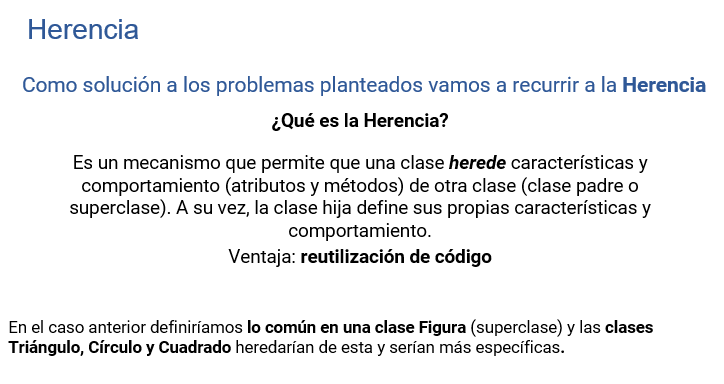
Public void setTitulo (String titulo){

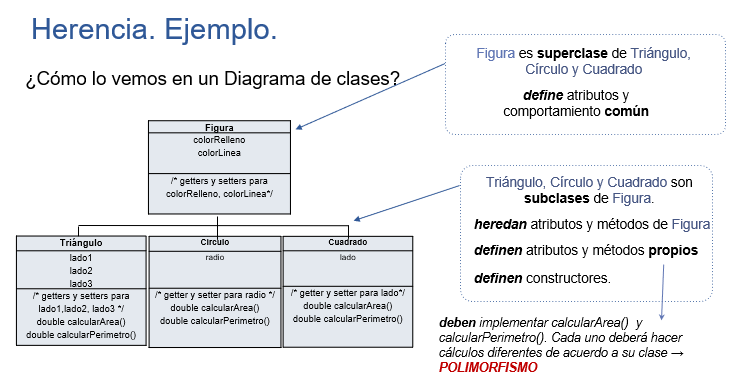
This.titulo = titulo;

}

**Clase 3**

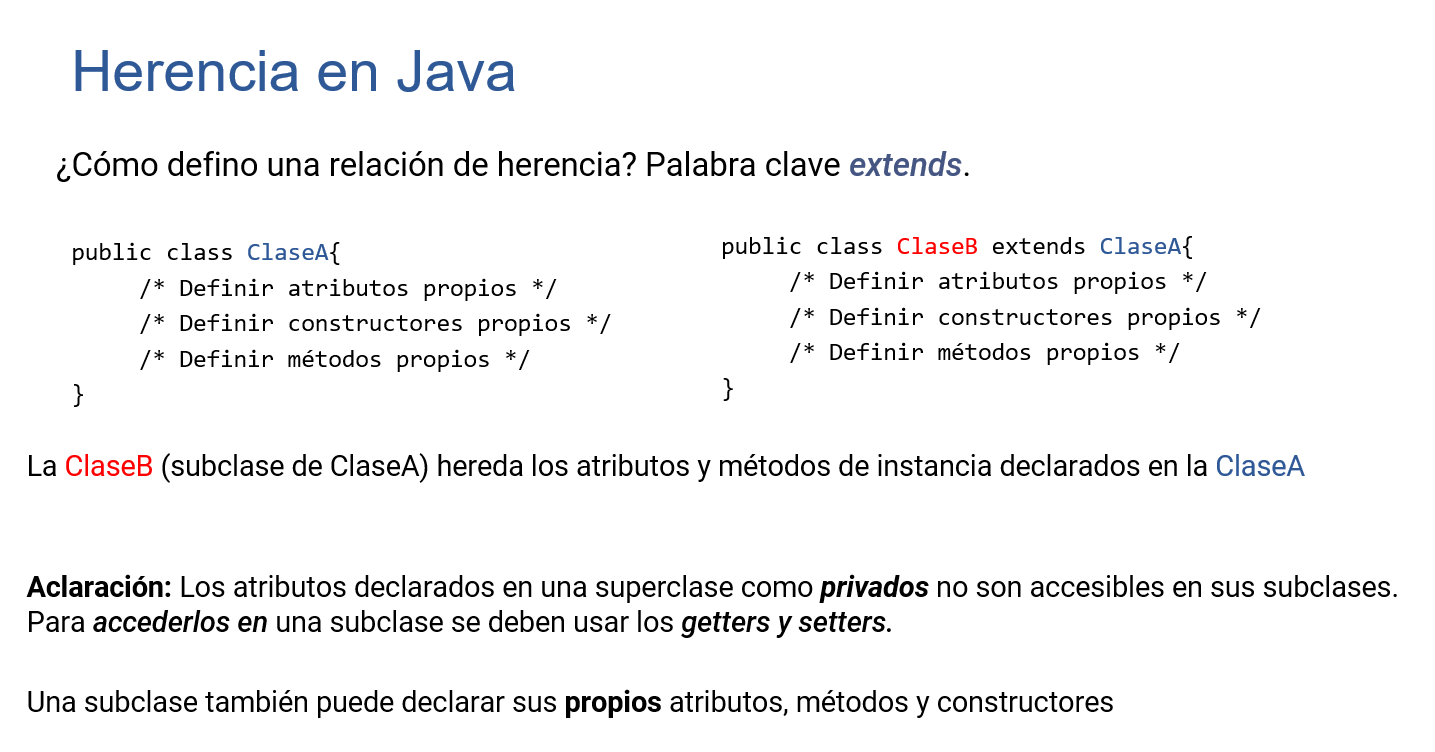
Herencia y polimorfismo



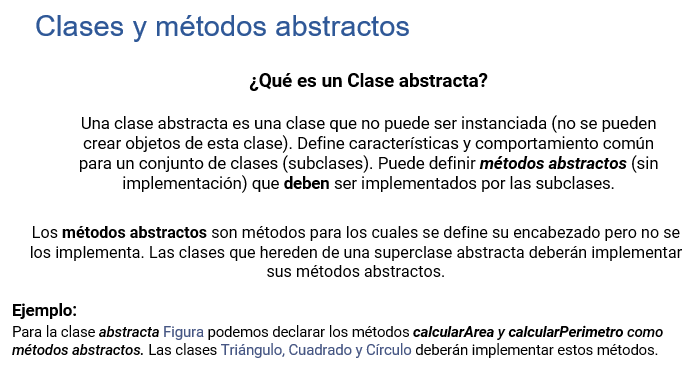


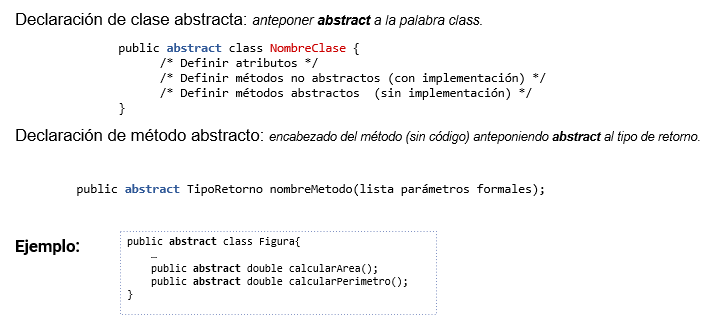
Búsqueda de método en la jerarquía de clases.

Cuando un objeto recibe un mensaje lo primero que hace es fijarse si ese método esta definido en la clase. Si esta definido en la clase termina la búsqueda y ejecuta el cuerpo de ese método. En caso de no estar definido lo que hace es pasar a la superclase, se fija si esta definido allí, finaliza la búsqueda y lo ejecuta. Si no esta definido se fija si existe otra superclase, si no existe da error, y vuelve a repetir el método anterior hasta encontrarlo o hasta que no exista una superclase.



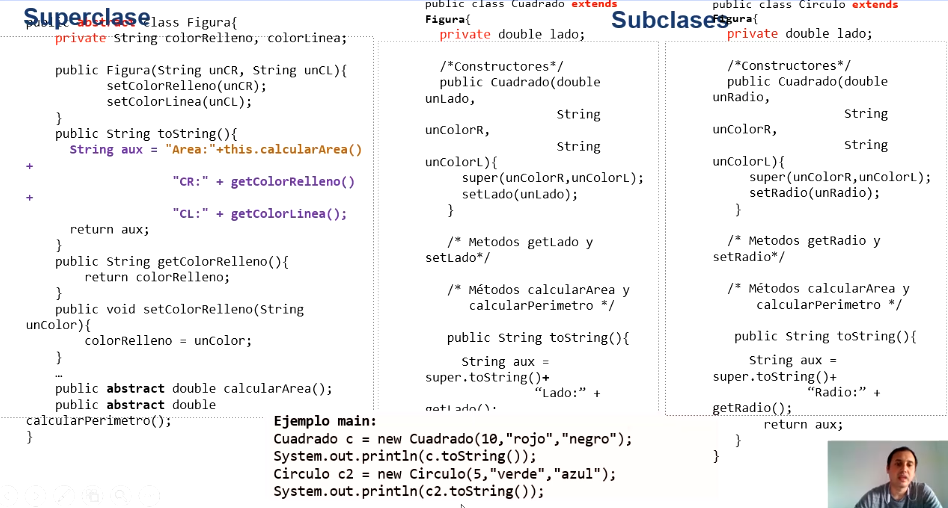
En algunas situaciones podemos encontrarnos con superclases de las cuales no nos resulta útil definir un nuevo objeto, sino que los nuevos objetos se van a instanciar desde sus subclases. Este sería el caso de la superclase Figura. Lo que terminaremos definiendo serán Triángulos, Cuadrados o Círculos. La clase Figura seria una clase abstracta.





Super(…) : el objeto se envia un mensaje a si mismo pero la búsqueda comienza en la clase superior a la actual. Lo puedo utilizar por ej cuando la superclase tiene definido un constructor:

Super(variables) o para llamar un metodo cualquiera de la superclase: super.toString()



Como instancie un cuadrado c, cuando invoque al super.toString(), ire al metodo de la superclase, y al hacer this.calcularArea(). Calculare el área del objeto que fue instanciado y que me llamo, en este caso c.

**Encapsulamiento.**

Permite construir componentes autónomos de software, es decir independientes de los demás componentes. La independencia se logra ocultando detalles internos (implementacion) de cada componente. Una vez encapsulado, el componente se puede ver como una caja negra de la cual solo se conoce su interfaz.

**Herencia.**

Permite definir una nueva clase en términos de una clase existente. La nueva clase hereda automáticamente todos los atributos y métodos de la clase existente, y a su vez puede definir atributos y métodos propios.

**Polimorfismo.**

Objetos de clases distintas pueden responder a mensajes con nombre (selector) sintácticamente idénticos. Esto permite realizar código genérico altamente reusable.

**Binding dinamico.**

Mecanismo por el cual se determina en tiempo de ejecución el método (código) a ejecutar para responder a un mensaje.

**Entre los beneficios de la POO, podemos mencionar producir software que sea:**

Natural: el programa queda expresado usando términos del problema a resolver, haciendo que sea mas fácil de comprender.

Fiable: la POO facilita la etapa de prueba del software. Cada clase se puede probar y validar independientemente.

Reusable: las clases implementadas pueden reusarse en distintos programas. Además gracias a la herencia podemos reutilizar el código de una clase para generar una nueva clase. El polimorfismo también ayuda a crear un código mas genérico.

Fácil de mantener: para corregir un problema nos limitamos a corregirlo en un único lugar.