La POO se basa en la idea de un mundo lleno de objetos y que la resolución de problemas se realiza mediante el modelado/encapsulamiento de objetos en la vida real. Los objetos pueden comunicarse entre sí mediante mensajes.

Se basa en el concepto de agrupar código y datos juntos dentro de una misma unidad llamada clase.

Sus principales características son :

* Abstracción. La posibilidad de traer algo del mundo real y abstraerlo al mundo lógico.
* Encapsulamiento
* Polimorfismo
* Herencia
* UML. Todas nuestras clases o modelado va a poder representarse mediante el diagrama UML.

Clases

Es una plantilla, o molde que permite construir un objeto. Por ejemplo, la clase vehículo sería el molde para construir el Objeto (coches, motos…). Ejemplos: Persona, Figura, Factura…

Representan entidades del mundo real. No se pueden utilizar directamente, sino mediante instancias a objetos (ejemplares).

Todos los nombres de una clase tienen que empezar en mayúscula y tienen que estar en singular, son el molde para 1 persona… Cada clase tiene atributos (variables) y métodos. Los métodos y variables empiezan en minúscula.

Métodos

Los métodos u operaciones son acciones contenidas en una clase, y definen su comportamiento. Dentro de un sistema los métodos se suelen detectar como verbos y pueden tener valores de entrada (parámetros) y valores de salida (valores de retorno).

Si devuelven algo son funciones, si no son procedimientos.

Constructores

Un objeto es una instancia de una clase, es la representación de un objeto que existe en la vida real. Posee un estado(atributos) y comportamiento(métodos).

Son funciones especiales que contienen las clases para permitir la creación de objetos. Pueden recibir sus parámetros (constructor) como no (constructores vacíos). Se llama SIEMPRE igual que la clase y no retorna ningún valor(ni VOID). Los constructores se diferencian por los parámetros de entrada.

HERENCIA

Hay clases que comparten gran parte de sus características. La herencia permite reutilizar clases: Se crea una nueva clase que extiende la funcionalidad de una clase existente sin tener que reescribir el código asociado a esta última.

La nueva clase, a la que se denomina subclase, puede poseer atributos y métodos que no existan en la clase original. Los objetos de la nueva clase heredan los atributos y los métodos de la clase original, que se denominan superclase.

Persona es una clase genérica que sirve para almacenar datos en común de todas las personas (nombre, dirección, número de teléfono)

Empleado (subclase) hereda los atributos de la clase Persona(superclase), pero puede incluir atributos y métodos propios (puesto, cargo). Lo mismo ocurre con jefe.

Para ello en las subclases se pone extends “nombre de la superclase”.

POLIMORFISMO

El polimorfismo (sobrecarga de parámetros) significa muchas formas, consiste en que a partir de un objeto que pertenece a una clase madre, puedo tener otro objeto creado por una clase hija, a partir de esa existencia puedo hacer que como comparten atributos en común puedan ser considerados como si tuvieran la forma de un objeto.

A partir de vehículo (clase padre), tengo moto, coche, bus (clase hijo). Esto significa que comparten atributos entre ellos, compartiendo esa clase cualquiera de la clase hijo se puede comportar como la clase padre.

Function acelerar(vehículo) 🡪 acelerar(coche), acelerar(moto), acelerar(bus).

En un objeto de un tipo no puedo meter otro tipo de datos. Pero el polimorfismo nos permite hacer esto, ya que si a un vector de tipo vehículo se le asigna un coche no hay ningún error, porque es una subclase. Lo mismo pasa con bus y moto. Pero no puede

ENCAPSULAMIENTO

Significa reunir a todos los elementos que pueden considerarse pertenecientes a una misma entidad, al mismo nivel de abstracción. Nos permiten 3 tipos de niveles de acceso:

* PUBLIC (Se puede ver desde cualquier clase)
* PRIVATE (Se puede ver desde esa misma clase)
* PROTECTED (Se puede ver en las clases hijas)

CLASES ABSTRACTAS

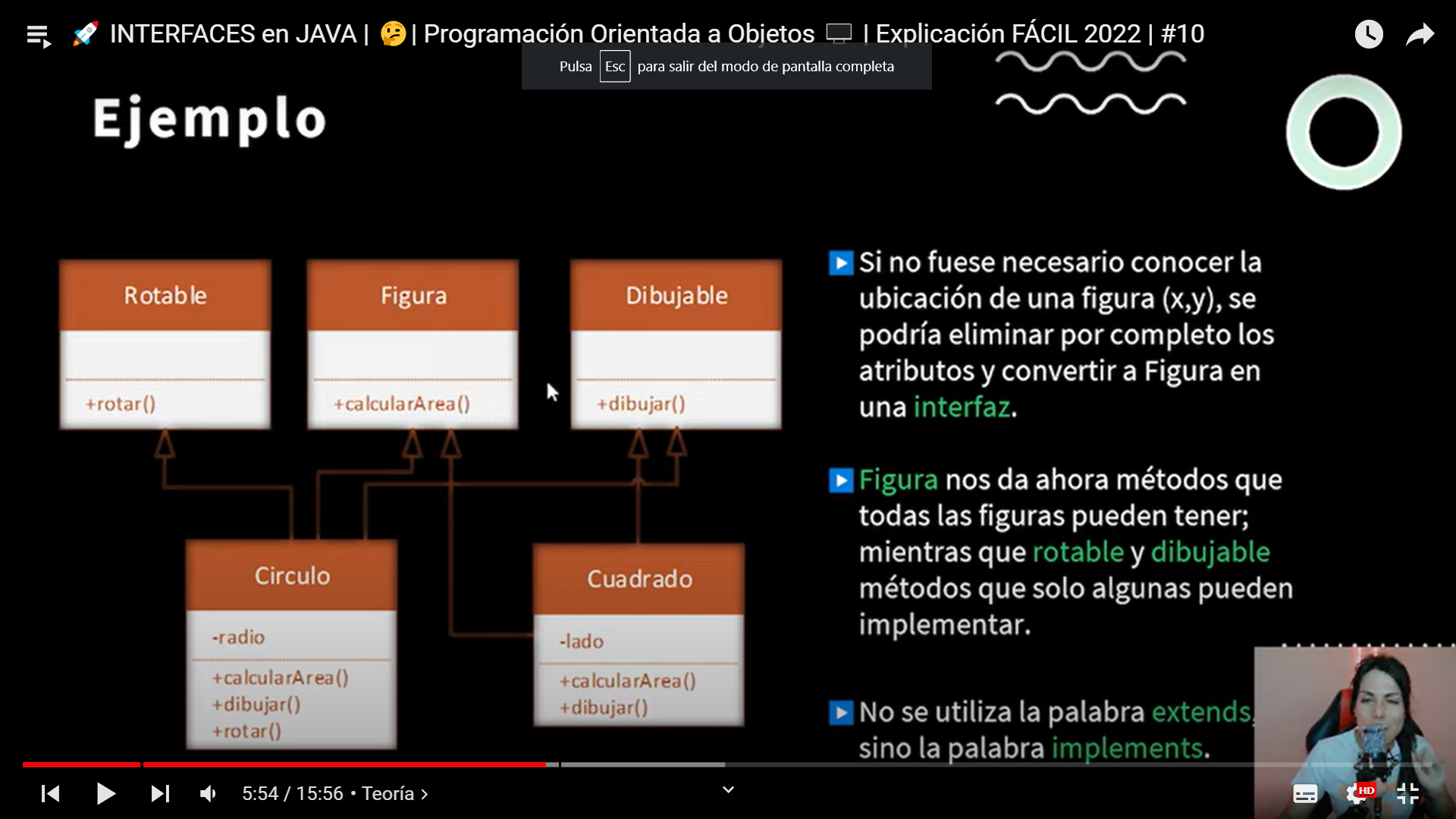
Son un tipo de clase cuya característica es que no pueden ser instanciadas, declarará la existencia de métodos, pero no su implementación (su funcionamiento), convirtiéndola así en una clase padre. Al menos uno de sus métodos debe ser abstracto y sus niveles de visualización de ser public o protected. Cuando se usan clases abstractas una clase no puede heredar de varias clases abstractas a la vez (interfaces). ´

Un ejemplo de abstracción y polimorfismo: Si una figura tiene dos hijos, círculo y cuadrado. Figura es una clase abstracta, ya que tiene el método calcularÁrea pero no muestra su funcionamiento, cada subclase actuará de manera diferente. Para ello se usa @override, sobreescritura del método abstracto de figura en el del hijo cuadrado, calcularArea tiene un funcionamiento distinto en cuadrado en figura. Se usa la palabra extends.

INTERFACES

Las interfaces complementan las clases abstractas y permiten la herencia múltiple, cosa que las clases abstractas no permitían. Son una colección de métodos abstractos con propiedades constantes. Una interfaz solamente puede extender o implementar otras interfaces. Solo puede tener métodos con acceso público y abstractos. Solo puede tener variables public static final, la palabra abstract en los métodos no es necesaria.

No se puede utilizar la palabra extends sino la palabra implements.



COLECCIONES

Las colecciones son estructuras similares a rrays pero con la principal característica de que son dinámicos (su tamaño y cantidad puede variar en el tiempo).

En Java, las colecciones se emplean mediante la interfaz “Collection”, que permite

implementar una serie de métodos comunes como ser: add(), remove(), size()…

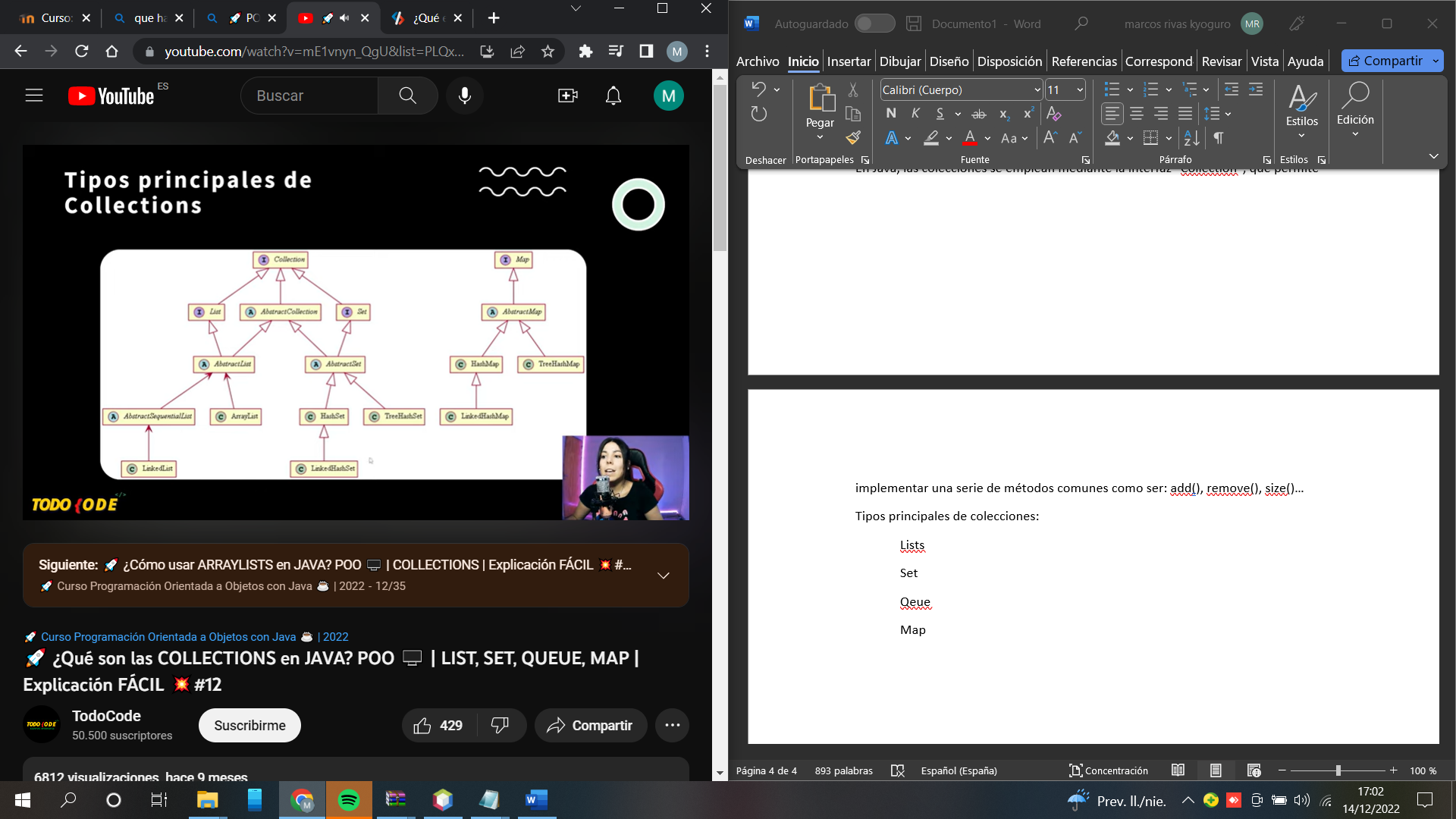
Tipos principales de colecciones:

Lists

Set

Qeue

Map



LISTAS

Las listas son un conjunto de elementos relacionados entre sí que tienen un determinado orden. Su tamaño es dinámico y existen 3 implementaciones:

ArrayLists

Es una clase que se representa como una matriz dinámica que permite almacenar elementos. Hereda de la clase AbstractList, la cual es implementada de la interfaz List. Permite elementos duplicados y el orden es el orden en el que fueron registrados. Permite acceso aleatorio, índices y tiene buen rendimiento en el recorrido pero no en el cambio

LinkedLists

Es una clase que representa una lista doblemente enlazada(ida y vuelta). Permite duplicados y mantiene el orden de inserción. Además tiene una manipulación más rápida. Se puede usar como lista o como pila o cola, permite añadir y eliminar al principio y al final de una lista.

Stack

FACTORY

Mediante una combinación de polimorfismo y métodos estáticos podemos facilitar mucho el intercambio de información, esto es, la transformación de datos procedentes de un archivo de intercambio CSV o similar en una colección de ejemplares de una clase.

La transformación incluye validación, un paso esencial cuando no es claro que la información recibida sea coherente. Téngase en cuenta que muchos datos proceden de transacciones que nunca estuvieron destinadas a un posterior análisis, por ejemplo de registros extraídos de servidores web de propósito general sin una configuración específica para el almacenamiento de información en sus registros.

