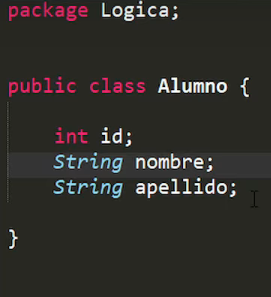
Programación Orientada a Objetos → POO

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

Principales características:

* Abstracción.
* Encapsulamiento.
* Polimorfismo.
* Herencia.
* UML.



Se trata de traer un objeto de la vida real.

Los nombres de las clases siempre van en singular.

Los atributos nos dicen cuales son las características de un objeto que traeremos de la vida real.

Los métodos nos dicen lo que puede hacer dicho objeto.

Objeto = Auto

Atributos = Número de ventanas, número de puertas, tipo de motor, color del auto, etc.

Métodos = Acelerar, frenar, estacionar, moverse, etc.

Crearemos un método para que el alumno pueda decir su nombre completo.

Forma de armar un método:

Modificador de acceso:

Public

Tipo de dato que devolverá dicho método en caso que devuelva un valor. (En este caso el método no devolverá ningún valor si no que mostrará algo por pantalla):

Void

El tipo de dato VOID se utiliza para esos métodos que indican un procedimiento, es decir, se realiza algo pero no voy a obtener una respuesta para utilizarla luego, si no que hago una acción y listo. (En este caso la acción será mostrar su nombre completo por pantalla)

Al crear un método siempre debemos ponerle un verbo en infinitivo (Verbos que terminan en ir, er, ar) que indique lo que hará dicho método:

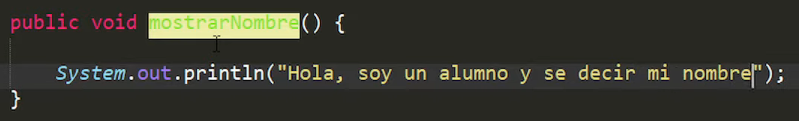


Al crear el nombre del método debemos usar camelCase (iniciamos con minúscula y si hay una segunda palabra, la iniciamos con mayúscula).

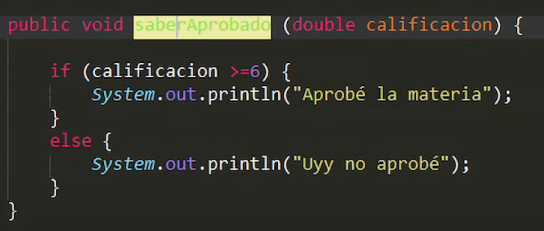
Al crear un atributo debemos usar snakecase.

Al crear una clase debemos usar Capitalcase.

Se ponen los paréntesis para indicar que se recibirá un parámetro.

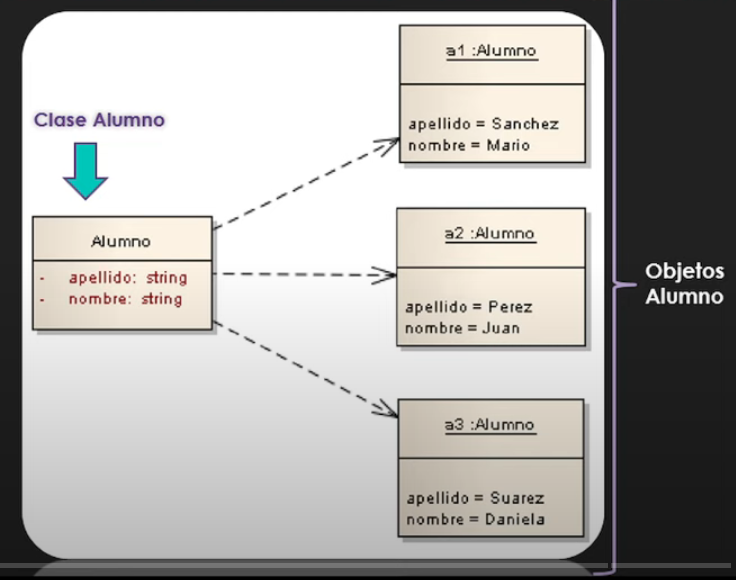


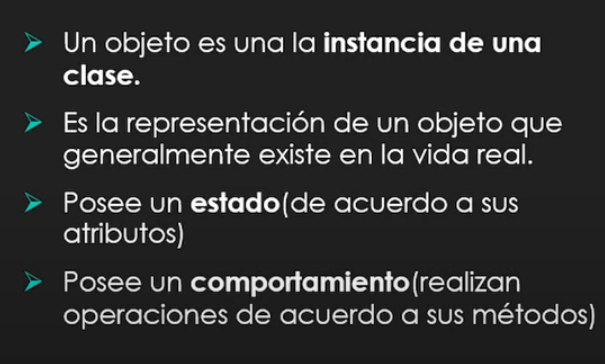
También podemos hacer lo siguiente:



Todo lo que puede hacer el alumno estará dentro del alumno, no esparcido en el main u otra clase.

Crear una instancia es crear un objeto de tipo alumno.



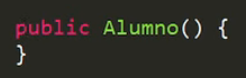


Los constructores no retornan ningún valor (Ni siquiera VOID)

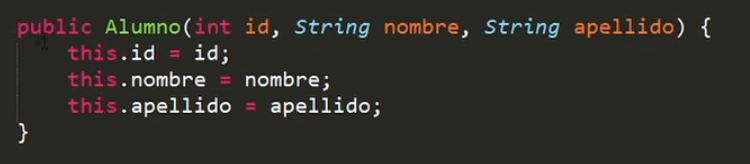
El constructor vacío nos permite crear objetos sin datos, es decir, con unicamente sus atributos y métodos pero sin datos guardados.

El constructor con todos los parámetros donde le podemos ir pasando los datos que queremos que tenga cada uno de esos objetos.

Esto es un constructor vacío:

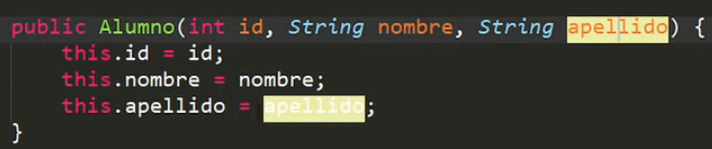


Esto es un constructor con parámetros:

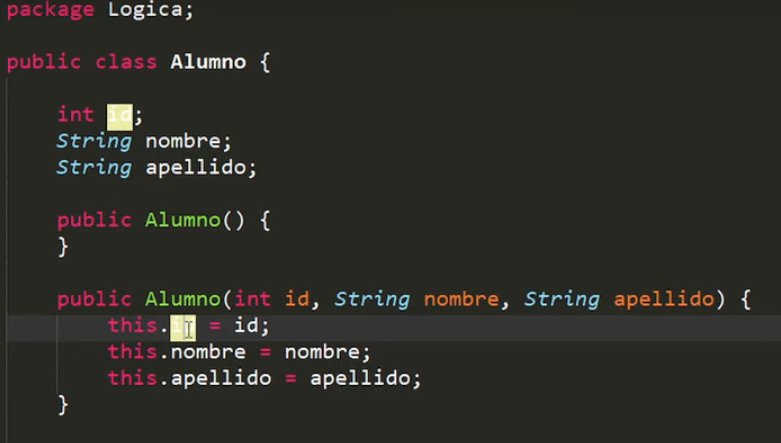


Allí se muestran los datos a los cuales deseamos ingresar algo con su respectivo tipo de dato.

El constructor con parámetros va a recibir datos los cuales le asignará a los atributos que creamos inicialmente.



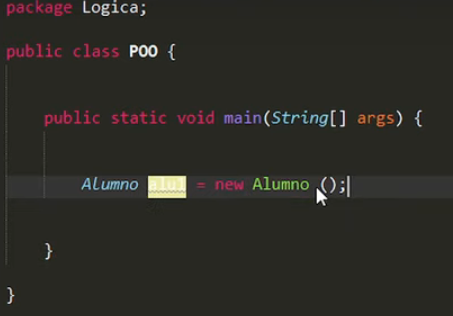
Las variables que se encuentran a la derecha hacen referencia a los parámetros ingresados en el constructor.



Y las variables que están con la palabra reservada “this.” hacen referencia a las variables inicializadas.

Cómo crear un alumno:

Primero lo declaramos cómo si fuera una variable

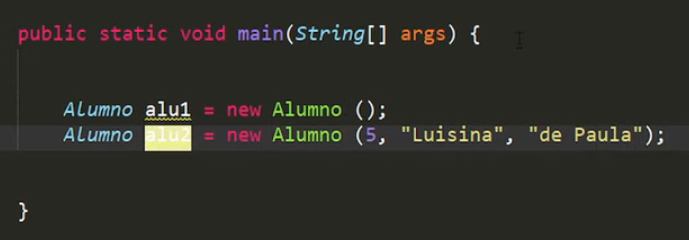


La primera palabra “Alumno” nos indica que el objeto que estamos creando es de tipo alumno, la segunda palabra que en este caso sería “alu1” es el nombre del objeto que estamos creando.

Después con la palabra reservada “new” estamos creando el objeto y depende de lo que se ingrese o no se ingrese dentro de los paréntesis estaremos creando un objeto mediante un constructor vacío o con parámetros.

En este caso estamos creando un Alumno vacío mediante nuestro constructor que también se encuentra vacío.

Si por el contrario deseamos crear un objeto Alumno con parámetros, debemos ingresar los mismos parámetros que tiene el constructor con parámetros y en el mismo orden que recibe el mismo y separados por comas.



En el segundo objeto Alumno podemos ver que le estamos pasando parámetros (en este caso 5, “Luisina”, “de Paula”) que indican los parámetros del constructor (en este caso int id, String nombre, String apellido).

Para establecer los Getters y Setters debemos hacerlo en la clase Alumno.

Los Getters y Setters pueden ser ingresados en el código siempre y cuando sea antes de los métodos que ingresamos nosotros a mano, personalmente, de forma personalizada (en este caso “mostrarNombre”) o en su defecto pueden ser ingresados después de dicho métodos siempre y cuando los Getters y Setters estén después de los métodos Constructor.

Los Getter y Setter son métodos especiales que nos permiten traer los valores de cada uno de los atributos que tengamos de una clase (en este caso id, nombre, apellido)

Por cada atributo de la clase nos va a generar un get y un set.

Get = Trae

Set = Coloca

Con los métodos Get podré obtener los datos para por ejemplo hacer una operación, mostrarlos en pantalla o lo que yo necesite hacer.

Con los métodos Set yo puedo colocar o establecer valores a cada uno de estos atributos.(Sirve para los casos en los que creemos un objeto vacío o cuando yo quiera modificar un valor por x o y motivo, razón o circunstancia).