

Programación orientada a  
objetos con Java

---

Microlearning

## Diferencias entre clases y objetos, y cómo se inicializan.

¡Has acabado de ver el video de microlearning con los temas:  
diferencias entre clases y objetos, y constructores!

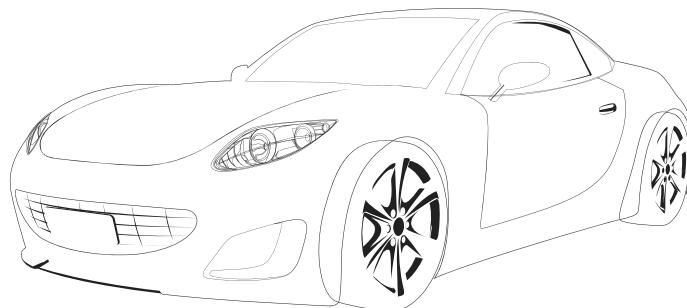
Estás dando los primeros pasos pero pronto estarás llegando a la meta.

Durante el video los conceptos más importantes fueron:

- ▶ Programación orientada a objetos
- ▶ Diferencias entre clases y objetos
- ▶ Atributos y métodos
- ▶ Constructores

La POO es un estilo de programación que consiste en un sistema de objetos separados pero que se comunican entre ellos y que buscan resolver un problema o generar programas y aplicaciones.

Una clase es un diseño de cómo se construye un objeto. En el video mostramos un ejemplo de cómo a través del diseño de un coche es posible producir varios de ellos sin tener que reinventar su proceso de fabricación desde cero y retrabajar en defectos que seguramente ya han sido resueltos anteriormente.



Se puede decir que cada uno de los coches que se produce es un objeto. Es importante entender y concluir de este ejemplo que pueden existir cualquier cantidad de objetos (coches) pero sólo una clase (diseño).

Este mismo concepto se extiende a la POO. Primero debemos crear un diseño o una clase del objeto que vamos utilizar en nuestro programa y con este diseño creamos una o más instancias de la clase (objetos).

Recuerda que al igual que en la vida real los objetos pueden tener atributos y métodos.

Anteriormente, vimos el ejemplo de una pelota la cual puede tener los siguientes atributos y métodos:

Objeto:  
Pelota

ATRIBUTOS = PROPIEDADES  
MÉTODOS = FUNCIONES

Métodos: Rozar, saltar  
Atributos: Color, tamaño, textura

Si este objeto lo definieramos en una clase se representaría de la siguiente manera:

```
class Pelota {  
    String color = "Naranja";  
    String tamaño = "Mediana";  
    String textura = "Lisa"; } Atributos  
    void rodar() {  
        System.out.println("La pelota " + color + " y " + tamaño + " rueda."); } Métodos  
    void saltar() {  
        System.out.println("La pelota " + color + " y " + tamaño + " salta."); } }
```

Una vez definida la clase, podremos utilizarla desde nuestro programa principal o desde cualquier otra parte del proyecto al crear una o más instancias de ella.

```

1 package Paquete1;
2
3 public class Main {
4
5     public static void main(String[] args) {
6
7         Pelota pelota_naranja = new Pelota();
8
9         pelota_naranja.rodar();
10        pelota_naranja.saltar();
11
12        System.out.println();
13
14
15        Pelota pelota_azul = new Pelota();
16
17        pelota_azul.color = "Azul";
18        pelota_azul.tamaño = "Chica";
19
20        pelota_azul.rodar();
21        pelota_azul.saltar();
22
23    }
24
25
26 }

```

Console

```

terminated: Main (5) [Java Application] /Library/Java/JavaVirtualMachines/jdk-11.0.12.jdk/Contents/Home/bin/java [Mar 7, 2022, 10:39:57 PM - 10:39:57 PM]
La pelota Naranja y Mediana rueda.
La pelota Naranja y Mediana salta.
La pelota Azul y Chica rueda.
La pelota Azul y Chica salta.

```

Los constructores nos ayudan a inicializar un objeto, esto quiere decir que en el momento que se crea una nueva instancia se ejecuta el código que se encuentra en el constructor. Para poder definir un constructor debe de crearse un método con el mismo nombre de la clase.

En el ejemplo del video vimos como un constructor de la clase Coche es creado:

```

1 public class Coche {
2
3     String motor;
4     int puertas;
5     int ruedas;
6
7     Coche(){
8         System.out.println("El objeto coche inicializa.");
9     }
10
11     void arrancar() {
12         System.out.println("El coche arranca.");
13     }
14
15     void frenar() {
16         System.out.println("El coche frena.");
17     }
18
19     void acelera() {
20         System.out.println("El coche acelera.");
21     }
22 }

```

Si ejecutamos una instancia de esta clase veríamos que el primer código que se ejecuta en el programa es el que se encuentra en el constructor.

```

terminated: Main (4) [Java Application] /Library/Java/JavaVirtualMachines/jdk-11.0.12.jdk/Contents/Home/bin/java [Mar 7, 2022, 10:39:57 PM - 10:39:57 PM]
El objeto coche inicializa.
El coche arranca.
El coche frena.
El coche acelera.

```

Una clase puede tener uno o más constructores y estos pueden contener argumentos. Los argumentos nos ayudarán a inicializar los objetos con valores desde el programa principal.

```
1 public class Coche {  
2     String motor;  
3     int puertas;  
4     int ruedas;  
5  
6     Coche(){  
7         System.out.println("El objeto coche inicializa.");  
8     }  
9  
10    Coche(String motor, int puertas, int ruedas){  
11        System.out.println("El objeto coche inicializa con argumentos.");  
12        this.motor = motor;  
13        this.puertas = puertas;  
14        this.ruedas = ruedas;  
15    }  
16  
17    void imprimir_argumentos(){  
18        System.out.println("El coche tiene motor de " + this.motor +  
19                    " con " + this.puertas + " puertas y " +  
20                    this.ruedas + " ruedas.");  
21    }  
22  
23    void arrancar(){  
24        System.out.println("El coche arranca.");  
25    }  
26  
27    void frenar(){  
28        System.out.println("El coche frena.");  
29    }  
30  
31    void acelera(){  
32        System.out.println("El coche acelera.");  
33    }  
34}
```

El constructor que se mande a llamar será el que cumpla con el mismo tipo de dato y orden de los argumentos al crear una nueva instancia.

## Nueva instancia sin argumentos:

The screenshot shows a Java application running in an IDE. The code editor displays Main.java with the following content:

```
1 public class Main {  
2     public static void main(String[] args) {  
3         // TODO Auto-generated method stub  
4         Coche coche = new Coche();  
5     }  
6 }  
7  
8  
9  
10 }  
11 }  
12 }  
13 }
```

The terminal window below shows the output of the application:

```
<terminated> Main [4] [Java Application] /Library/Java/JavaVirtualMachines/jdk-11.0.12.jdk/Contents/Home/bin/java (Mar 7, 2022, 11:06:16 PM - 11:06:17 PM)  
El objeto coche inicializa.
```

## Nueva instancia con argumentos:



The screenshot shows an IDE interface with two tabs: 'Coche.java' and 'Main.java'. The 'Main.java' tab is active, displaying the following code:

```
1  public class Main {  
2      public static void main(String[] args) {  
3          // TODO Auto-generated method stub  
4          Coche coche = new Coche("3 cilindros", 2, 4);  
5      }  
6  }
```

The 'Console' tab at the bottom shows the output of the application:

```
<terminated> Main (4) [Java Application] /Library/Java/JavaVirtualMachines/dk-11.0.12.jdk/Contents/Home/bin/java (Mar 7, 2022, 11:05:49 PM – 11:05:51 PM)  
El objeto coche inicializa con argumentos.
```

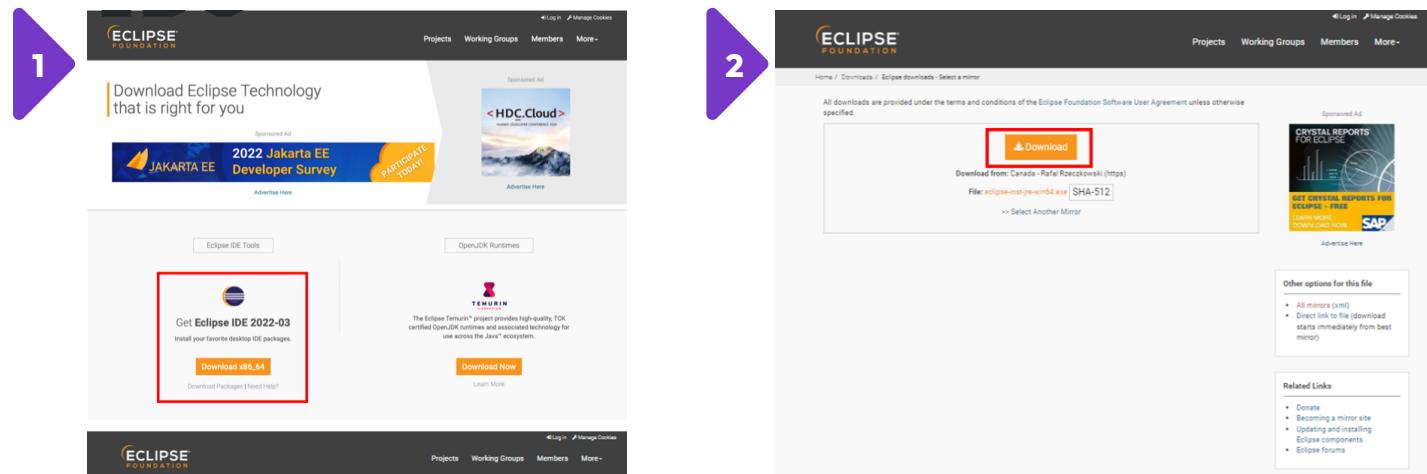
No olvides que clases y objetos son términos que todo el tiempo estarás utilizando al realizar un programa con programación orientada a objetos (POO) y son el punto de partida para que construyas tu código.

Por último te dejamos un resumen de los pasos necesarios para que puedas crear una clase tú mismo, comprobar estos ejemplos y resolver los ejercicios que te proporcionaremos más adelante.

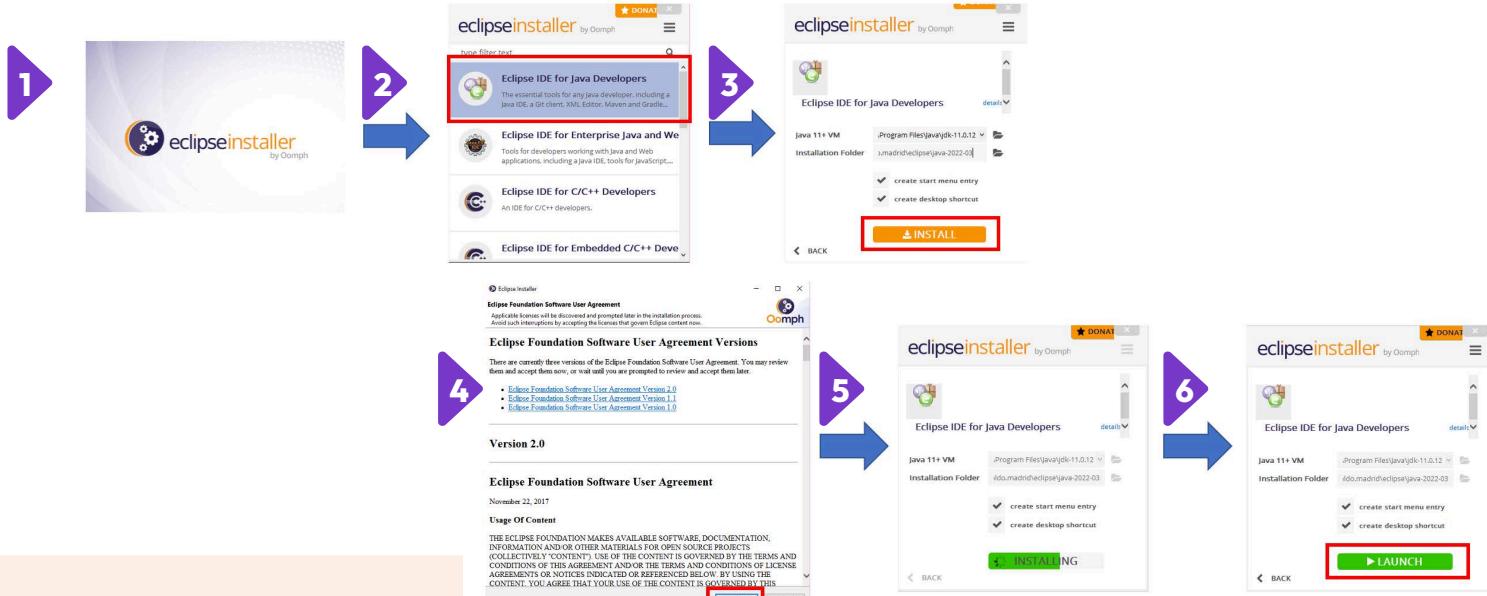
1. Para comenzar a crear nuestra primera clase procedemos a abrir nuestra herramienta de desarrollo: Eclipse.

- Si aún no tienes instalado Eclipse, puedes descargarlo desde este enlace:  
<https://www.eclipse.org/downloads/>

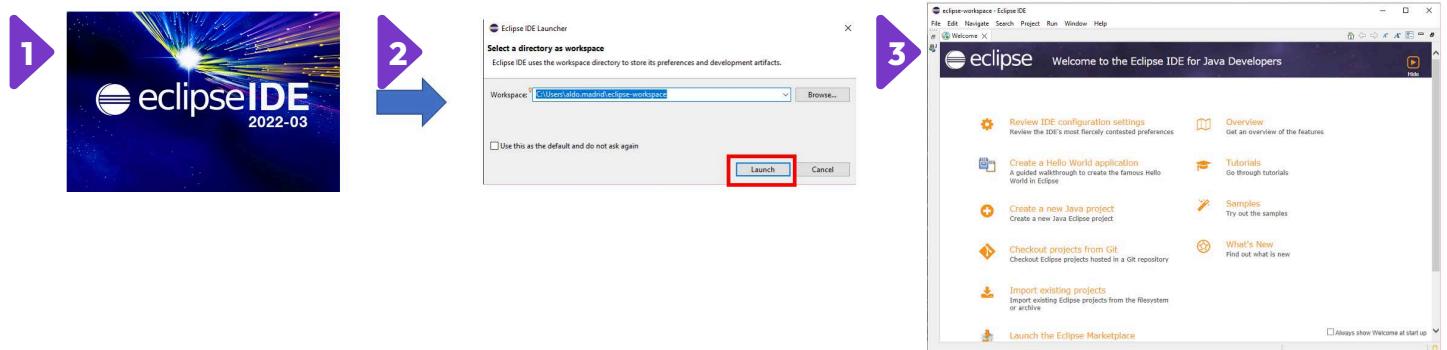
- Ahí solo debes hacer click en Download.



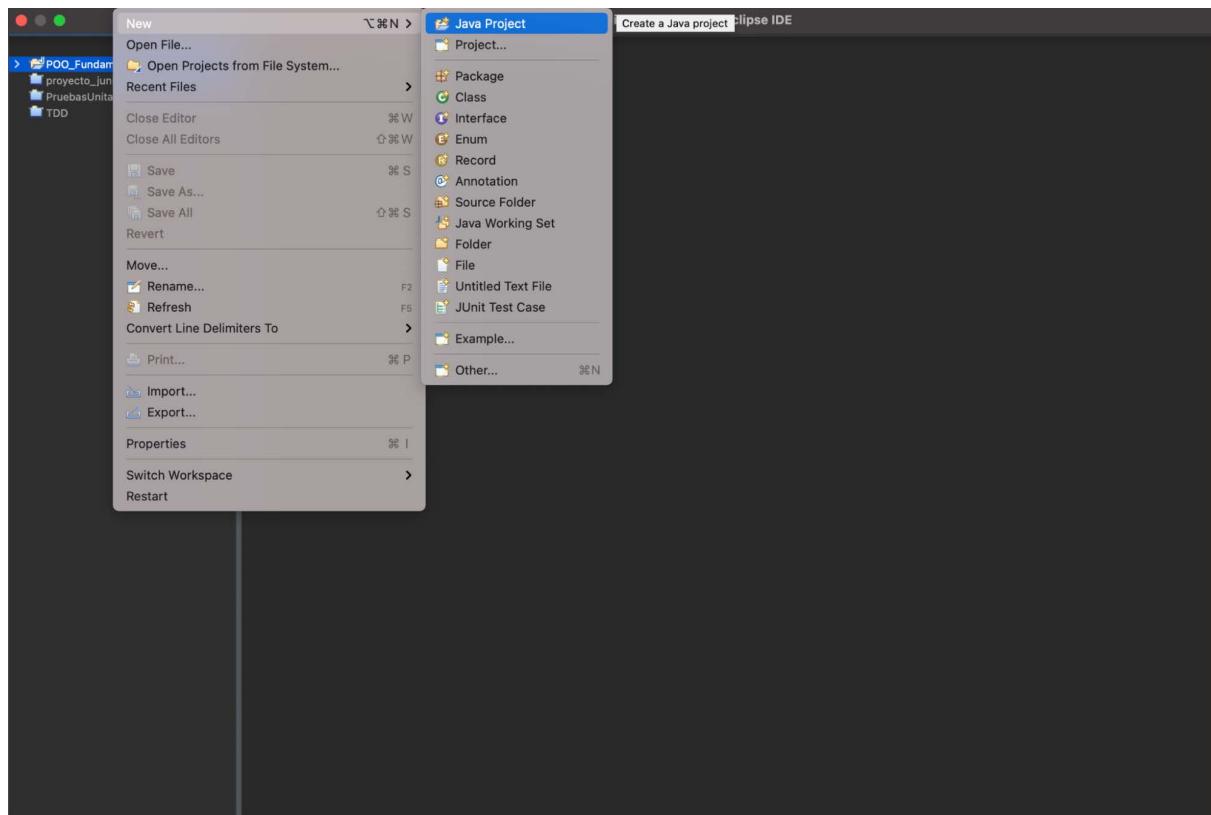
- Una vez que se haya concluido la descarga debes hacer doble click en el ejecutable y seguir estos pasos:



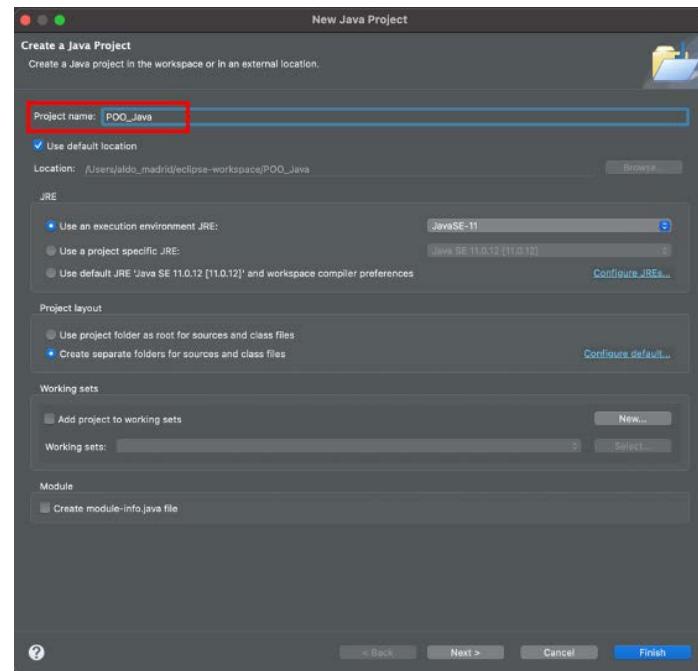
- La herramienta Eclipse se abrirá automáticamente.



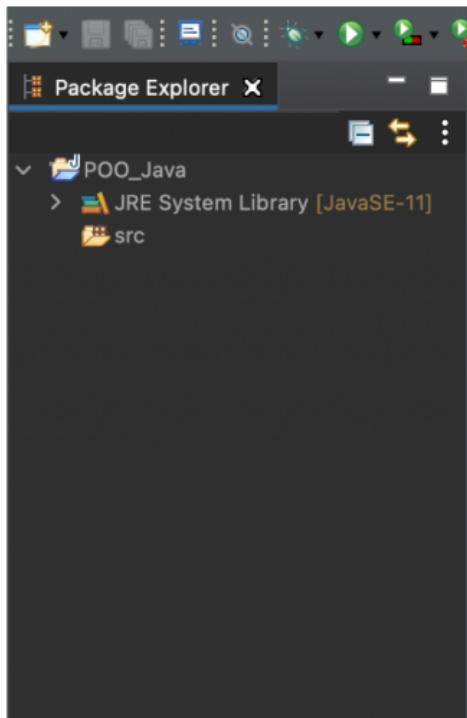
2. Una vez abierto nos dirigimos a la pestaña Archivo -> Nuevo -> Nuevo Proyecto.



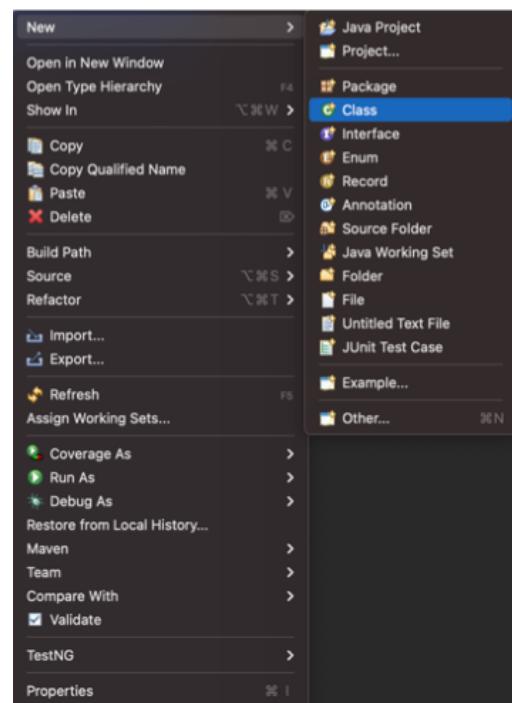
3. En la siguiente ventana colocamos un nombre al proyecto como se muestra en la imagen, y una vez agregado pulsamos el botón Terminar:



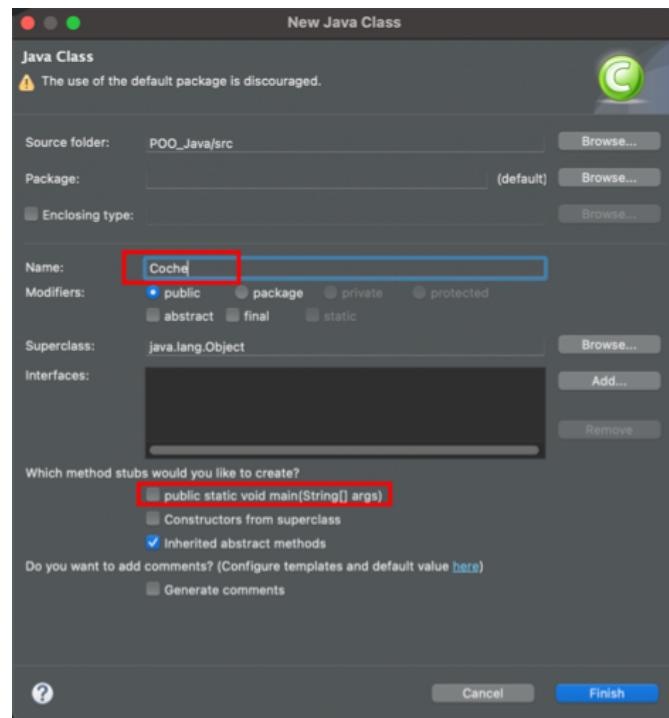
4. El proyecto se creará en el explorador de paquetes.



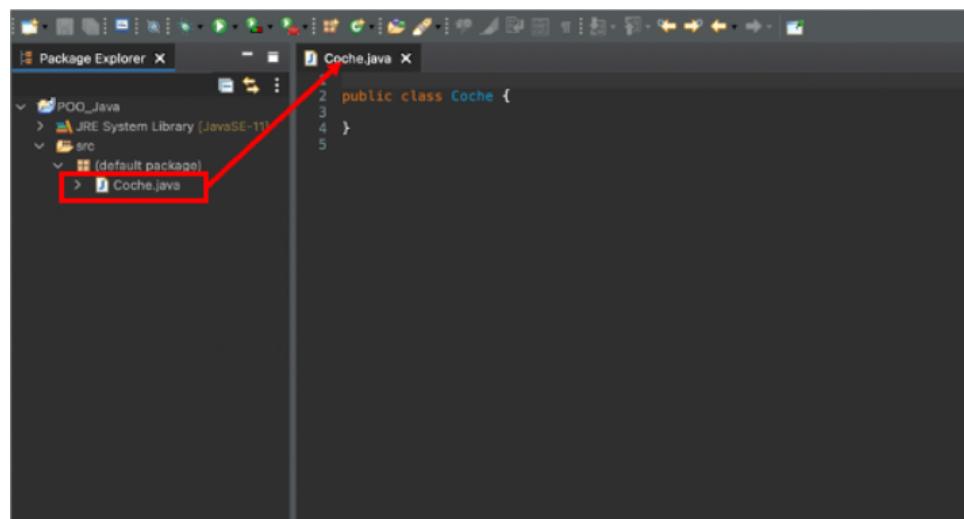
5. Para continuar, hacemos click derecho en folder "src" y seleccionamos Nuevo -> Class.



6. En la siguiente ventana agregamos el nombre de la clase “Coche” y nos aseguraremos que la opción de “public static void main” esté desactivada.



7. Ahora veremos que un archivo con extensión java se crea en el explorador de paquetes que contiene el código de la clase “Coche”.



Ahora pongamos lo aprendido en práctica:

El primer ejercicio que haremos es: Operaciones con fracciones

Te tomará 10 minutos.

Y para hacerlo necesitas:

- Crear una clase llamada Fracciones.
- Agregar los métodos y atributos necesarios para sumar, restar, multiplicar y dividir dos fracciones, los métodos no deben de recibir argumentos, deberás usar los atributos de la clase. ¿Cuántos atributos identificas? Revisa el siguiente punto para más información.
- Para que puedas implementar el código de esta clase te proporcionaremos la siguiente información relacionada a operaciones con fracciones.

Elementos de una fracción

#### Elementos de una fracción.

$$\begin{array}{l} \frac{a}{b} \xrightarrow{\text{Numerador}} \\ \xrightarrow{\text{Denominador}} \end{array}$$

#### Suma

$$\frac{a}{b} + \frac{c}{d} = \frac{ad + bc}{bd}$$

#### Multiplicación

$$\frac{a}{b} * \frac{c}{d} = \frac{ac}{bd}$$

#### Resta

$$\frac{a}{b} - \frac{c}{d} = \frac{ad - bc}{bd}$$

#### División

$$\frac{a}{b} / \frac{c}{d} = \frac{ad}{cb}$$

- Una vez implementada la clase, debes de crear una instancia de esta clase en el programa principal y ejecutar los métodos sumar, restar, multiplicar y dividir y verificar que funcionen correctamente.

## ¿Listo?

Aquí te dejamos un ejemplo del resultado que se debe imprimir en consola al ejecutar el programa para las siguientes fracciones:

Fracción 1

$$\frac{4}{2}$$

Fracción 2

$$\frac{5}{3}$$

```
Console ✘
<terminated> Main (5) [Java Application] /Library/Java/JavaVirtualMachines/jdk-11
El resultado de la suma de fracciones es: 22/6
El resultado de la resta de fracciones es: 2/6
El resultado de la multiplicación de fracciones es: 20/6
El resultado de la división de fracciones es: 12/10
```

Recuerda que parte esencial de la POO es dedicar tiempo a diseñar y entender cómo resolverías el problema, es por eso que debes identificar todos los atributos y métodos que necesitarás y entender cómo acomodarlos dentro de la clase.

El segundo ejercicio que haremos es: Calificación de alumnos

Te tomará 10 minutos.

- Imaginemos que en un curso de ciencia de datos se inscribieron los siguientes alumnos y obtuvieron las siguientes calificaciones finales:

Nombre	Apellido	Matrícula	Calificación
Clara	Ramos	F34A024	90
Jorge	Rodriguez	F27A031	46
Ivan	Hernandez	F24A123	71
Jazmin	Peralta	F30A078	58

- Se requiere diseñar una clase que se llame Alumno, la cual tendrá un constructor que reciba argumentos para inicializar los atributos necesarios para cada alumno.
- Definir un método dentro de la clase que se llame “imprimir\_resultado”, este método deberá imprimir un mensaje de calificación reprobatoria (< 70), calificación aprobatoria ( $\geq 70$  y  $< 90$ ) y calificación excelente ( $\geq 90$ ), dependiendo de la calificación de cada alumno (deberás utilizar condiciones if, else if y else).
- En el programa principal se deberá crear una instancia para cada alumno pasando los argumentos necesarios al constructor.
- Cada vez que se mande a llamar el método “imprimir\_resultado” para cada alumno se deberá imprimir el mensaje correcto.

## ¿Listo?

Aquí te dejamos un ejemplo de lo que se debe imprimir en consola y como debería de verse tu programa principal:



```

Main.java  D Alumno.java
1 package Paquete1;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Alumno a1 = new Alumno();
7         Alumno a2 = new Alumno();
8         Alumno a3 = new Alumno();
9         Alumno a4 = new Alumno();      Argumentos constructor.
10
11         a1.imprimir_resultado();
12         System.out.println();
13         a2.imprimir_resultado();
14         System.out.println();
15         a3.imprimir_resultado();
16         System.out.println();
17         a4.imprimir_resultado();
18         System.out.println();
19     }
20
21
22 }
23

```

Resultado en consola:

```
Console ×
<terminated> Main (5) [Java Application] /Library/Java/JavaVirtualMachines/jdk-11.0.12.jdk/Contents/Home/bin/java (Mar 19, 2022,
Alumn@: Clara, Apellido: Ramos, Matricula: F34A024
Felicitaciones tu calificación es excelente!!

Alumn@: Jorge, Apellido: Rodriguez, Matricula: F27A031
Tu calificación no es aprobatoria. Te sugerimos estudies nuevamente los temas vistos.

Alumn@: Ivan, Apellido: Hernandez, Matricula: F24A123
Bien hecho tu calificación es aprobatoria.

Alumn@: Jazmin, Apellido: Peralta, Matricula: F30A078
Tu calificación no es aprobatoria. Te sugerimos estudies nuevamente los temas vistos.
```

Has terminado.

## ¡Muchas felicidades!

El paso siguiente es realizar un cuestionario interactivo para reforzar tus conocimientos y habilidades.

¡Estamos emocionados por seguir aquí contigo!



## IMPORTANTE

Toda la información contenida en el presente documento es propiedad única, exclusiva y reservada de Exponential Education, S. de R.L. de C.V. (en lo sucesivo “Bedu”) y es considerada como Información Confidencial en términos de la legislación de Propiedad Industrial aplicable.

Derivado de lo anterior, cualquier copia, retransmisión u otros usos distintos sin autorización expresa de Bedu, se encuentra estrictamente prohibida.