

Guía de aprendizaje

Aplicaciones en Java a partir de variables asociadas a datos

Exp 2 – Semana 5

Fundamentos de Programación (PRY2201)

**Facilitador disciplinar:** Josué Oteiza

**Asesor par:** Luis Ávila

# Índice

[Índice 2](#_Toc158805976)

[Introducción a la semana 3](#_Toc158805977)

[Resultado de aprendizaje 3](#_Toc158805978)

[Conceptos relevantes 4](#_Toc158805979)

[Preguntas activadoras 4](#_Toc158805980)

[Actividad 4](#_Toc158805981)

[Variables en Java 5](#_Toc158805982)

[Recordando sus características 5](#_Toc158805983)

[Tipos de variables en Java 7](#_Toc158805985)

[Declaración e inicialización de variables en Java 13](#_Toc158805987)

[Links de Interés 19](#_Toc158805989)

[Cierre de la semana 20](#_Toc158805990)

[Referencias 21](#_Toc158805991)

[Lecturas de la semana 21](#_Toc158805992)

[Apuntes 22](#_Toc158805993)

# Introducción a la semana

Ya te encuentras a mitad de camino de la experiencia 2 donde seguiremos profundizando en contenidos ya vistos a grandes rasgos anteriormente. Ahora es el tiempo de las variables en Java, elementos cruciales que permiten almacenar información que puede cambiar durante la ejecución de un programa. Conocerás distintos tipos, su propósito y alcances dentro del código. Además, aprenderás el proceso de declarar e inicializar una variable para gestionar el estado y el flujo de un programa, facilitando la escritura de código y haciéndolo eficiente y comprensible.

# Resultado de aprendizaje

### El estudiante será capaz de:

**RA2**. Aplica elementos básicos de Java para el desarrollo de programas que solucionen problemáticas planteadas.

### Indicador de logro:

**IL5.** Emplea Variables en Java en la resolución de problemáticas planteadas, asegurando el correcto almacenamiento y manipulación de datos.

# Conceptos relevantes

|  |  |  |
| --- | --- | --- |
| Optimización del Código | Declaración de una variable | Inicialización de una variable |
| Tipos de Datos Básicos | Concatenación de Cadenas | Validación de Resultados |

# Preguntas activadoras

* ¿Cuál es la diferencia entre la declaración e inicialización de una variable?
* ¿Cómo seleccionamos el tipo de dato adecuado al declarar una variable en Java?
* ¿Cuándo es apropiado utilizar operadores de asignación compuestos en lugar de operadores básicos?

# Actividad

**Descripción de la actividad**

En esta quinta semana, realizarás una actividad formativa individual con encargo, llamada “Aplicando variables en Java”, donde deberás proponer una solución al caso planteado con la programación de algoritmos y variables, considerando los distintos tipos y declarando e iniciando tales variables para resolver problemas específicos.

# Variables en Java

## Recordando sus características

El término “Variables” ya es conocido por ti, pero para poder continuar, haremos un pequeño repaso de lo que has aprendido sobre ellas.

### ¿Qué es una variable?

Se puede conceptualizar como un contenedor de información que almacena datos que pueden ser modificados durante la ejecución de un programa. En otras palabras, permiten que los programas sean dinámicos y flexibles. La capacidad de una variable para cambiar su valor es lo que las hace tan valiosas en programación.

Algunas de las características que tienen las variables son:

* **Flexibilidad y eficiencia:** las variables permiten escribir código que puede manejar datos que cambian o que no se conocen hasta que el programa se ejecuta. Esto hace que los programas sean más flexibles y eficientes.
* **Facilitación de la lectura y mantenimiento del código:** utilizar variables con nombres descriptivos hace que el código sea más fácil de leer y entender. Esto es crucial para el mantenimiento y la actualización del software.
* **Reutilización de código:** las variables facilitan la reutilización de código. Un mismo bloque de código puede ser utilizado con diferentes valores de variables, lo que reduce la necesidad de escribir código nuevo.
* **Control del flujo del programa:** las variables son clave para controlar el flujo de un programa. Pueden usarse en estructuras condicionales y bucles para determinar cómo y cuándo se ejecuta un bloque de código.

Importante

Las variables también facilitan el almacenamiento, acceso, procesamiento, transferencia y análisis de datos, lo que conduce a códigos más claros y mantenibles.

# Video

Antes de continuar, revisa el siguiente video, donde repasarás la creación de variables en Java:

<https://www.youtube.com/watch?v=Uz3_WmaC42o>

## Tipos de variables en Java

En Java existen varios tipos y se pueden clasificar principalmente en tres categorías:

* Variables locales
* Variables de clase o estáticas, y
* Variables de instancia o no estáticas.

Cada categoría tiene su propio propósito y ámbito de aplicación; profundizaremos sobre esto a continuación:

### Variables locales

Son aquellas que se declaran dentro de un método, constructor o bloque de código y solo son accesibles dentro de ese ámbito. Estas variables no son visibles fuera de su bloque de declaración, lo que significa que no pueden ser utilizadas por el resto del programa. Además, las variables locales no tienen un valor predeterminado, por lo que deben ser inicializadas antes de su uso. Su ciclo de vida está limitado al ámbito en el que se declaran, siendo destruidas una vez que el flujo de control sale de ese ámbito.

¿Cómo se demuestra esto? Imagina un método en Java que calcula el salario semanal de un empleado. En la siguiente figura, ‘horasTrabajadas’ y ‘tarifaPorHora’ son variables locales:

**Figura 1**

*Variable local*

Nota. Ejemplo de variables locales en Java. The Apache Software Foundation. (2023). *Apache NetBeans (19).* [Software]. Apache NetBeans. <https://netbeans.apache.org/front/main/>

Como puedes ver en este ejemplo, dentro del método ‘calcularSalario()’ de la clase Empleado, se definen dos variables locales: ‘horasTrabajadas’ y ‘tarifaPorHora’. ‘horasTrabajadas’ se inicializa con un valor de 40, asumiendo una semana laboral estándar. ‘tarifaPorHora’ se establece en 3500 CLP, representando la tarifa horaria en pesos chilenos. El salario se calcula multiplicando las horas trabajadas por la tarifa por hora, y el resultado (salario semanal) se muestra en consola expresado en pesos chilenos.

### Variables de clase (estáticas)

Son compartidas por todas las instancias de una clase. Se definen con la palabra clave ‘static’ y se colocan fuera de cualquier método o constructor, pero dentro de la clase. Hay una sola copia de la variable de clase, compartida por todas las instancias de la clase y se accede a ellas utilizando el nombre de la clase seguido por el operador de punto (.).

Aquí tienes un ejemplo de su aplicación:

**Figura 2**

*Variables de clase*

Nota. Ejemplo de variables de clase en Java. The Apache Software Foundation. (2023). *Apache NetBeans (19).* [Software]. Apache NetBeans. <https://netbeans.apache.org/front/main/>

En este ejemplo, puedes ver que la clase ‘ContadorDeAutos’ tiene una variable de clase ‘totalAutos’ que lleva la cuenta del número total de autos creados. Cada vez que se instancia un nuevo objeto ‘ContadorDeAutos’ (es decir, se crea un nuevo auto), el constructor incrementa ‘totalAutos’ en 1. Al ser una variable de clase (estática), ‘totalAutos’ no está ligada a una instancia específica, sino que se comparte entre todas las instancias de la clase. Se puede acceder a esta variable y obtener su valor actual mediante el método estático ‘getTotalAutos()’, que devuelve el valor de ‘totalAutos’. En el método ‘main’, se crean dos autos y luego se muestra el total usando ‘ContadorDeAutos.getTotalAutos()’, que en este caso será 2.

### Variables de instancia (no estáticas)

Son variables que pertenecen a una instancia específica de una clase donde cada objeto creado a partir de esa clase tiene su propia copia de estas variables. Son declaradas fuera de cualquier método, pero dentro de una clase.

Las variables de instancia representan el estado de un objeto y pueden tener valores diferentes en diferentes instancias de la misma clase. Son accesibles en todos los métodos no estáticos de la clase y su ciclo de vida es el mismo que el del objeto al que pertenecen.

En el siguiente ejemplo, te mostraremos cómo se utilizan las variables de instancia para representar las características específicas de un automóvil:

**Figura 3**

*Variables de instancia*

Nota. Ejemplo de variables de instancia en Java. The Apache Software Foundation. (2023). *Apache NetBeans (19).* [Software]. Apache NetBeans. <https://netbeans.apache.org/front/main/>

De acuerdo con el ejemplo, la clase ‘Automovil’ representa un automóvil con tres características: marca, modelo y año, que son variables de instancia. Cada objeto ‘Automovil’ creado tendrá su propia marca, modelo y año. Por ejemplo, cuando se crea el objeto ‘miAuto’ con los valores "Toyota", "Corolla" y 2020, estas características se asignan a las variables de instancia del objeto. Luego, el método ‘mostrarInfo()’ accede a estas variables de instancia para mostrar la información del automóvil.

Las variables en Java se clasifican en estas tres categorías para organizar el código y determinar su alcance y ciclo de vida, pero también se pueden clasificar según su tipo de datos, como enteros (int, long), decimales (float, double), caracteres (char), booleanos (boolean), entre otros. La declaración de una variable incluye su tipo y nombre, y opcionalmente puede incluir una inicialización con un valor.

**Figura 4**

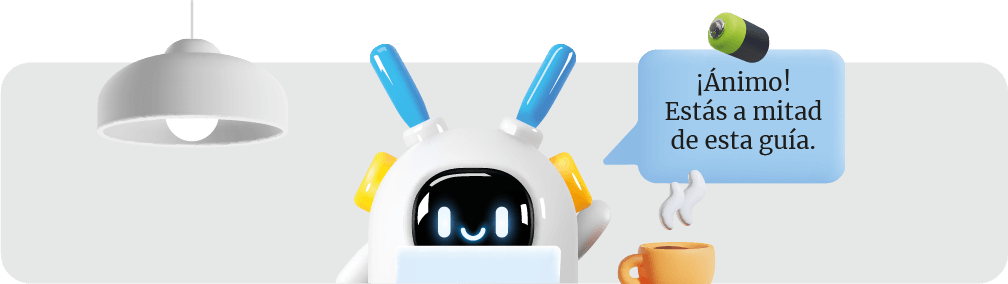
*Clasificación de variables en Java*

*Nota.* Muestra de clasificación de variables en Java*. The Apache Software Foundation. (2023). Apache NetBeans (19).* [Software]. Apache NetBeans. <https://netbeans.apache.org/front/main/>

# Video

Te invitamos a revisar el siguiente video, donde podrás observar el uso de los tipos de variables en Apache NetBeans:

<https://www.youtube.com/watch?v=fMXwqSZGkIY>

Ahora que ya profundizamos en las categorías de variables, te mostraremos cómo se declaran e inicializan en Java.

# Declaración e inicialización de variables en Java

En Java, la declaración e inicialización de variables son dos conceptos fundamentales de la programación. La declaración es el proceso por el cual se le dice al compilador que se reserve un espacio en memoria para una variable definiendo su tipo y nombre. La inicialización ocurre cuando se asigna un valor a esa variable por primera vez. En Java, es posible declarar una variable sin inicializarla de inmediato, especialmente en el caso de las variables de instancia y de clase, que tienen valores predeterminados. Sin embargo, las variables locales deben inicializarse antes de ser utilizadas. Aquí te dejamos algunos ejemplos de cómo declarar e inicializar variables en Java:

**Figura 5**

*Formas básicas de trabajo en Java*

*Nota*. Declaración e iniciación de variables en Java. *The Apache Software Foundation. (2023). Apache NetBeans (19).* [Software]. Apache NetBeans. <https://netbeans.apache.org/front/main/>

Importante

En Java, las variables deben declararse antes de ser utilizadas. Además, el tipo de datos de la variable se especifica al declararla, y este tipo no puede cambiarse una vez que se ha establecido.

## Declaración e inicialización de variables en el caso del Teatro Moro

*¿Recuerdas la solicitud realizada por el Teatro Moro?*

El cliente quería optimizar su servicio de venta de entradas, por lo que se necesitaba desarrollar un sistema de ventas en Java con la implementación de diversas funcionalidades entre las cuales se destacan la gestión de variables, menú de opciones, concatenación de cadenas, validación de datos y optimización de código.

Si bien ya brindaste esta solución en semanas anteriores, ahora nos enfocaremos en dar solución al requerimiento, pero con un enfoque en variables y optimización de código. Veamos a continuación el paso a paso de cómo abordar cada uno de estos aspectos:

**Paso 1 - Declaración e Inicialización de variables:** en Java, las variables se declaran especificando el tipo de dato seguido por el nombre de la variable. Utiliza nombres descriptivos para mejorar la comprensión del código e inicializa las variables antes de su uso para evitar errores. Por ejemplo:

**Figura 6**

*Declarando variables*

*Nota*. Declaración e iniciación de variables en Java. *The Apache Software Foundation. (2023). Apache NetBeans (19).* [Software]. Apache NetBeans. <https://netbeans.apache.org/front/main/>

La primera variable, ‘cantidadEntradasVendidas’, es un entero que se inicializa en 0 y está destinada a llevar la cuenta de la cantidad de entradas. La segunda, ‘totalIngresosPorVentas’, es un número de doble precisión inicializado en 0.0, utilizado para mantener un registro del total monetario de las ventas. La tercera, ‘nombreObraTeatro’, es una cadena de texto (‘String’) que se inicializa como una cadena vacía, y se usará para almacenar el nombre de la obra. Estas declaraciones establecen los valores iniciales para las variables que se utilizarán posteriormente en el programa.

**Paso 2 – Implementación de un menú de opciones:** implementa un menú para que los usuarios seleccionen diferentes acciones. Se utiliza la clase `Scanner` para obtener la entrada del usuario:

**Figura 7**

*Clase ‘Scanner’*

*Nota*. Menú de opciones en Java. *The Apache Software Foundation. (2023). Apache NetBeans (19).* [Software]. Apache NetBeans. <https://netbeans.apache.org/front/main/>

Utiliza un objeto ‘Scanner’ para leer la entrada del usuario, quien debe elegir entre tres opciones presentadas en la consola. Las opciones son "Registrar venta", "Consultar ventas" y "Salir". Según la elección del usuario, capturada en la variable ‘opcion’, el programa ejecuta un bloque de código correspondiente dentro de una estructura ‘switch’. Si se elige "Registrar venta" o "Consultar ventas", el programa ejecutaría la lógica específica para esas funciones. Si se elige "Salir", se muestra un mensaje y se termina la ejecución. Si se ingresa cualquier otro número, se informa al usuario que la opción no es válida.

**Paso 3 - Concatenación de cadenas:** utiliza tipos de datos adecuados para almacenar información (‘int’, ‘double’, ‘String’) y aplica la concatenación de cadenas para mostrar información de manera legible:

**Figura 8**

*Concatenación de cadenas*

Nota. Concatenación de cadenas en Java. The Apache Software Foundation. (2023). *Apache NetBeans (19).* [Software]. Apache NetBeans. <https://netbeans.apache.org/front/main/>

Este fragmento de código en Java imprimirá en consola los detalles de una venta. Mostrará el nombre de la obra de teatro, la cantidad de entradas vendidas y el total de ventas en pesos. Estos datos se mostrarán concatenando las variables ‘nombreObraTeatro’, ‘cantidadEntradasVendidas’ y ‘totalIngresosPorVentas’ con cadenas de texto que describen lo que se está imprimiendo.

**Paso 4 - Validación de resultados ingresados por el usuario:** utiliza estructuras de control como `if` para validar la entrada del usuario y proporciona mensajes claros para indicar posibles errores:

**Figura 9**

*Estructuras ‘if’ y ‘else’*

Nota. Estructuras de control en Java. The Apache Software Foundation. (2023). *Apache NetBeans (19).* [Software]. Apache NetBeans. <https://netbeans.apache.org/front/main/>

**Paso 5 - Optimización de código:** utiliza estructuras de control eficientes y evita redundancias, desarrolla funciones o métodos para modularizar el código y emplea buenas prácticas de programación para mejorar la legibilidad:

**Figura 10**

*Estructuras, funciones y métodos en Java*

*Nota*. Estructuras de control en Java. *The Apache Software Foundation. (2023).* *Apache NetBeans (19).* [Software]. Apache NetBeans. <https://netbeans.apache.org/front/main/>

Este código evalúa si el número de entradas vendidas es menor que cero, lo que no sería lógico en un contexto real, y en tal caso, imprime un mensaje de error. De lo contrario, pasa a procesar la cantidad de entradas vendidas, asumiendo que es un número válido.

Como puedes ver, al declarar e inicializar variables con precisión, se previenen errores y se facilita la comprensión del código. Además, implementar un menú interactivo mejora la experiencia del usuario, permitiendo una interacción clara y dirigida. En cuanto a la concatenación de cadenas y la validación de entradas garantizan que la información se maneje y muestre correctamente.

# Links de Interés

Te invitamos a seguir profundizando en las variables en Java a través del siguiente enlace: <https://www.manualweb.net/java/variables-java/>

# Cierre de la semana

En Java, las variables son fundamentales para almacenar y manipular datos dentro de un programa. Existen diversos tipos de variables, incluyendo locales, de instancia y de clase, cada una con su propio alcance y ciclo de vida.

La declaración de una variable la identifica con un tipo y un nombre, y la inicialización le asigna un valor inicial. Estos conceptos son esenciales para un código limpio y eficaz, y una correcta declaración e inicialización de variables contribuye a la claridad, la prevención de errores y la facilidad de mantenimiento del código en Java.

# Referencias

* Barragán, A. (2023). *Introducción a Java: Datos y variables.* OpenWebinars. <https://openwebinars.net/blog/introduccion-a-java-datos-y-variables/#:~:text=devuelven%20ning%C3%BAn%20valor.-,Qu%C3%A9%20son%20las%20variables%20en%20Java,a%20ella%20en%20el%20c%C3%B3digo>.
* G.TeC Facultad de Informática Universidad Complutense de Madrid (s.f.). Fundamentos de programación en Java [PDF].

<https://www.tesuva.edu.co/phocadownloadpap/Fundamentos%20de%20programcion%20en%20Java.pdf>

* Manualweb,net. (s.f.). Tutorial Java. <https://www.manualweb.net/java/variables-java/>
* ProgramarYa – By JuanDMeGon (2020). *Creación de variables en Java* [Video]: <https://www.youtube.com/watch?v=Uz3_WmaC42o>

# Lecturas de la semana

* Capítulo 4: Making the Most of Variables and Their Values

Burd, B. (2022). Java for Dummies. New Jersey: John Wiley & Sons. <https://webezproxy.duoc.cl/login?url=http://biblioteca.duoc.cl/bdigital/elibros/a50163-Java_fordummies/68/> Páginas 57 a 68

* Capítulo 1: Introducción a Java - Variables

Vegas Gertrudix, J. M. (2022). Java 17: Fundamentos prácticos de programación. Bogotá: Ediciones de la U. <https://webezproxy.duoc.cl/login?url=http://biblioteca.duoc.cl/bdigital/elibros/a50229-Java_17/33/> Páginas 37 a 40

# Apuntes

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



Reservados todos los derechos Fundación Instituto Profesional Duoc UC. No se permite copiar, reproducir, reeditar, descargar, publicar, emitir, difundir, de forma total o parcial la presente obra, ni su incorporación a un sistema informático, ni su transmisión en cualquier forma o por cualquier medio (electrónico, mecánico, fotocopia, grabación u otros) sin autorización previa y por escrito de Fundación Instituto Profesional Duoc UC La infracción de dichos derechos puede constituir un delito contra la propiedad intelectual.