



Java For Testers

Fecha: 01/09/2025



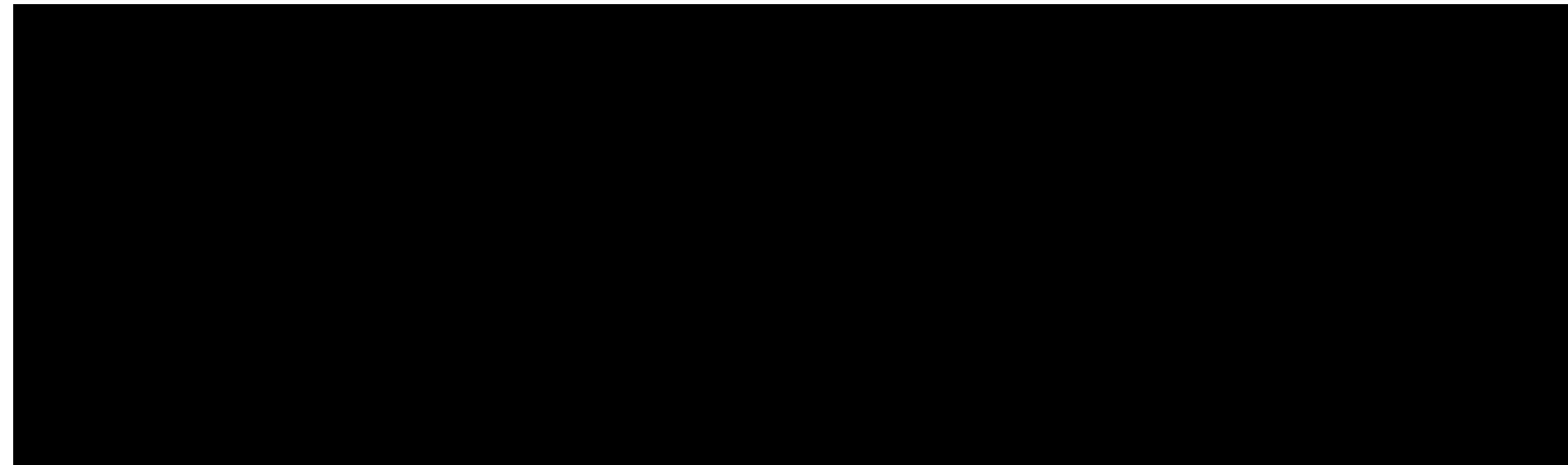
4

Condicionales y Loops

1. Condicionales
 - o IF
 - o IF/ELSE
 - o IF ANIDADOS
 - o Operador Ternario
2. For - Iteraciones
 - o For Clásico
 - o Enhanced For
3. While - Ciclos

4. Condicionales y Loops

4.1 Condicionales



Condicionales

- ★ Permiten que un programa **tome decisiones** según una condición (verdadero o falso)
(Se cumple o no se cumple).
- ★ Son la base para controlar el flujo del código.
- ★ **Sin condicionales, el código no decide: solo ejecuta en línea recta.**

¿Por qué un Tester debería saberlo?

- ★ Para validar reglas de negocio.
- ★ Para **automatizar** decisiones (si pasa, continúa / si falla, detiene).
- ★ Porque la mayoría de los flujos de una app dependen de condiciones.

Bloque IF

- ★ Ejecuta código **sólo si** la condición se cumple.
- ★ Nota las llaves del **IF**, todo lo que hay dentro se ejecuta solo si la condición se cumple
- ★ Analiza cómo “**equalsIgnoreCase**” es 1 método de un **String** que regresa Boolean,
“Qué puedes deducir de esto”

Recuerda puedes comparar los siguientes valores en una condicional

- ★ Boolean
- ★ Comparaciones numéricas (==, <, >, <=, >=)
- ★ Métodos que regresan boolean (Ej. equalsIgnoreCase)

```
//Ejemplo 1. If con booleano  
boolean esAdmin = true;  
if (esAdmin ) {  
    System.out.println("Acceso permitido");  
}  
  
//Ejemplo 2. If con String accediendo al método  
boolean equalsIgnoreCase  
String rolDeUsuario = "admin";  
if (rolDeUsuario.equalsIgnoreCase("admin")) {  
    System.out.println("Acceso permitido");  
}
```



Bloque IF / ELSE

- ★ Decide entre **DOS Caminos**.
- ★ Nota las llaves del **IF**, todo lo que hay dentro se ejecuta solo si la condición se cumple
- ★ Analiza las llaves del **ELSE**, todo lo que hay dentro se ejecuta si la condición de IF no se cumple

Recuerda puedes comparar los siguientes valores en una condicional

- ★ Boolean
- ★ Comparaciones numéricas (==, <, >, <=, >=)
- ★ Métodos que regresan boolean (Ej. equalsIgnoreCase)

```
if (edad >= 18) {  
    System.out.println("Es mayor de edad");  
} else {  
    System.out.println("No es mayor de edad");  
}
```



Bloque IF / ANIDADOS

- ★ Un **IF** dentro de otro **IF** para validar más de una condición.
- ★ Aplica con **ELSE**, es decir un **IF** dentro de un **ELSE** o viceversa
- ★ Analiza las llaves del código de ejemplo: **¿Qué puedes deducir?**

Recuerda puedes comparar los siguientes valores en una condicional

- ★ Boolean
- ★ Comparaciones numéricas (==, <, >, <=, >=)
- ★ Métodos que regresan boolean (Ej. equalsIgnoreCase)

```
boolean isAdmin = true;  
edad = 18;  
  
if (isAdmin) {  
    if (edad >= 18) {  
        System.out.println("Acceso permitido");  
    }  
}
```



Operador Ternario

- ★ Forma corta de un **IF / ELSE** en una sola línea.
- ★ Después de "?" es equivalente al bloque **IF**
- ★ Después de ":" es equivalente al bloque **ELSE**
- ★ No usa bloque por llaves "{}"
- ★ No puedes poner bloques grande de código
- ★ Analiza el código de ejemplo: ¿Qué puedes deducir?

Recuerda puedes comparar los siguientes valores en una condicional

- ★ Boolean
- ★ Comparaciones numéricas (==, <, >, <=, >=)
- ★ Métodos que regresan boolean (Ej. equalsIgnoreCase)

```
boolean esAdmin = true;  
// esAdmin == true  
String acceso = (esAdmin == true) ? "Permitido"  
: "Denegado";  
// esAdmin == true => Simplificando el comparador  
a true  
String acceso = (esAdmin) ? "Permitido" :  
"Denegado";  
  
int edad = 18;  
String mayorEdad = (edad >= 18) ? "Es mayor" :  
"Es menor";
```



Actividad

Realiza la siguiente actividad



1. Ve al código de ejemplo del repositorio
2. Analiza el código de condicionales en el paquete
“src/test/java/cap04/condicionales”
3. Importa el código a tu computador, ya hemos visto opciones de esto en capítulos anteriores
4. Entiende el código, ejecutalo

**NOTA

Los nombres de packages, directorios y archivos pueden cambiar en función de lo que el alumno elija no es forzoso mantener los nombre recomendados

Actividad

Realiza la siguiente actividad



1. Abre IntelliJ
2. Comienza a crear códigos de ejemplo relacionados a condicionales en el proyecto, package y archivo Java de tu elección

****NOTA**

Los nombres de packages, directorios y archivos pueden cambiar en función de lo que el alumno elija no es forzoso mantener los nombre recomendados

Actividad

Realiza la siguiente actividad



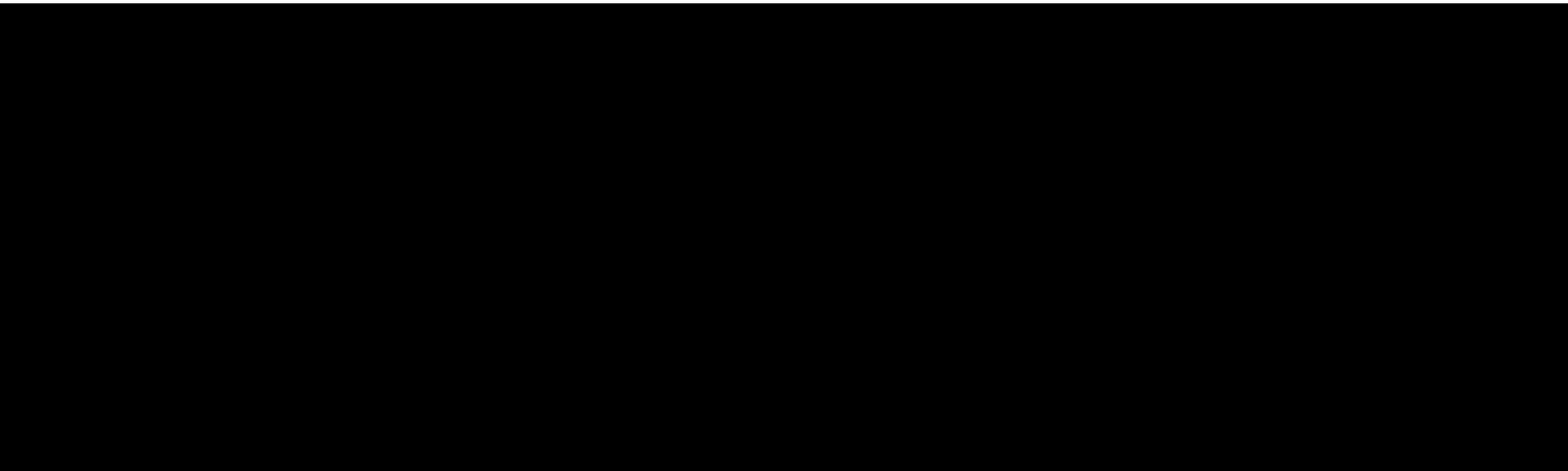
Te sugerimos esta actividad

1. Ve a la página del curso y resuelve el **Quiz 03**
2. Estos ejercicios te ayudarán a mejorar tu skill de Java
3. Te recomendamos intentar resolverlos por tu cuenta sin depender de una IA

****NOTA**

Los nombres de packages, directorios y archivos pueden cambiar en función de lo que el alumno elija no es forzoso mantener los nombre recomendados

4.2 For - Iteraciones



¿Qué es un FOR?

- ★ Es una estructura de control que permite **repetir código (Loop)**.
- ★ Cada repetición se llama iteración.
- ★ El número de iteraciones ya está definido desde el inicio.
- ★ Se ejecuta mientras se cumpla una condición clara.
- ★ No está pensado para ciclos infinitos (a diferencia de WHILE).
- ★ 2 formas de usar FOR
 - **For Clásico**
 - **Enhanced For (For Each)**

¿Por qué un Tester debería saber usar FOR?

- ★ Para recorrer arrays y listas de datos de prueba.
- ★ Para validar muchos escenarios con el mismo código.
- ★ Para automatizar acciones/validaciones repetitivas, evitando copiar el mismo código muchas veces.

FOR Clásico

- ★ Es un ciclo que **repite código** un número definido de veces usando tres partes:
 - Inicialización
 - Condición
 - Actualización
- ★ En una lista te va servir utilizar estos métodos:
 - **lista.size()** indica cuántos elementos hay en la lista.
 - **lista.get(i)** obtiene el elemento en esa posición.

Recuerda puedes ITERAR

- ★ Rangos numéricos (inicio → fin)
- ★ Datos en Arrays (length)
- ★ Datos en Listas (size)

```
//Empieza en 0 y va subiendo hasta 5.  
for (int i = 0; i <= 5; i++) {  
    System.out.println(i);  
}
```

```
//Empieza en 5 y va bajando hasta 1.  
for (int i = 5; i >= 1; i--) {  
    System.out.println(i);  
}
```



```
//Array (usando length)
String[] nombresArr = {"Bicho", "Zoe", "Josy"};

for (int i = 0; i < nombresArr.length; i++) {
    System.out.println(nombresArr[i]);
}

//Lista (usando size)
List<String> nombresList = new ArrayList<>();
nombresList.add("Bicho");
nombresList.add("Zoe");

for (int i = 0; i < nombresList.size(); i++) {
    System.out.println(nombresList.get(i));
}
```



Enhanced FOR (For-Each)

- ★ Es una versión más simple y limpia del for.
- ★ Se usa para recorrer arrays o listas.
- ★ No necesitas manejar índice (i).
- ★ Solo recorres los valores directamente.

FOR Clásico VS Enhanced

- for clásico → control total, usas índice.
- enhanced for → más limpio cuando solo necesitas leer datos.

Recuerda puedes ITERAR

- ★ Datos en Arrays (length)
- ★ Datos en Listas (size)

```
// Ejemplo con Array
String[] nombresArr = {"Bicho", "Zoe", "Josy"};

for (String nombre : nombresArr) {
    System.out.println(nombre);
}

// Ejemplo con Lista
List<String> nombresList = new ArrayList<>();
nombresList.add("Bicho");
nombresList.add("Josy");

for (String nombre : nombresList) {
    System.out.println(nombre);
}
```



Actividad

Realiza la siguiente actividad



1. Ve al código de ejemplo del repositorio
2. Analiza el código de condicionales en el paquete “src/test/java/cap04/loopsFor”
3. Importa el código a tu computador, ya hemos visto opciones de esto en capítulos anteriores
4. Entiende el código, ejecutalo

**NOTA

Los nombres de packages, directorios y archivos pueden cambiar en función de lo que el alumno elija no es forzoso mantener los nombre recomendados

Actividad

Realiza la siguiente actividad



1. Abre IntelliJ
2. Comienza a crear códigos de ejemplo relacionados a FOR en el proyecto, package y archivo Java de tu elección, cree nuevos de ser necesario

****NOTA**

Los nombres de packages, directorios y archivos pueden cambiar en función de lo que el alumno elija no es forzoso mantener los nombre recomendados

Actividad

Realiza la siguiente actividad



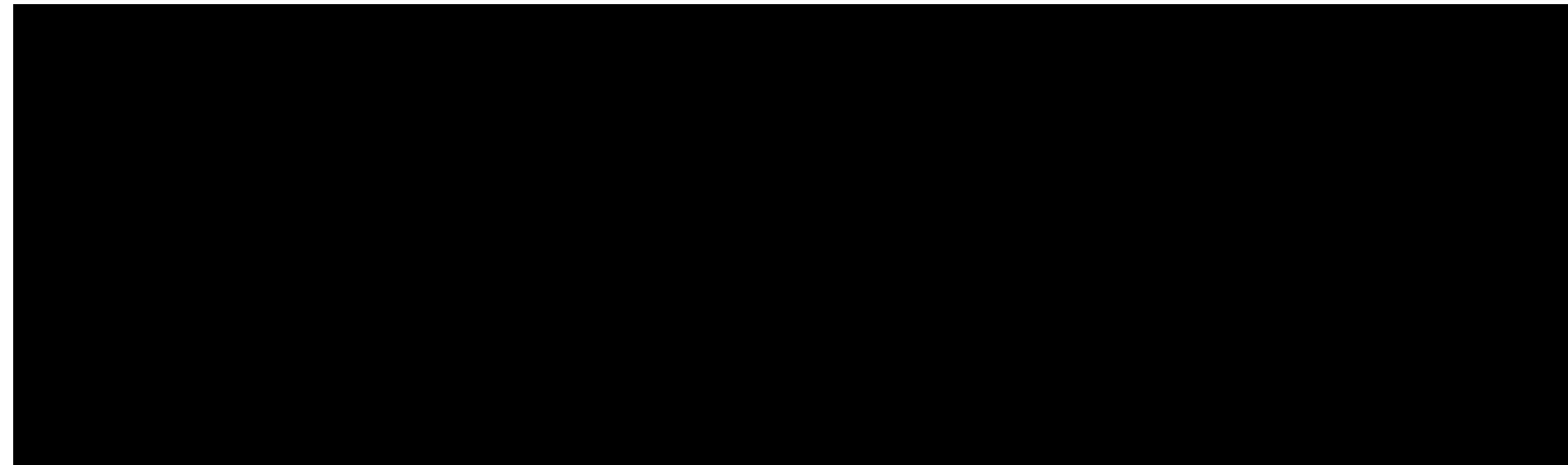
Te sugerimos esta actividad

1. Ve a la página del curso y resuelve el **Quiz 04**
2. Estos ejercicios te ayudarán a mejorar tu skill de Java
3. Te recomendamos intentar resolverlos por tu cuenta sin depender de una IA

****NOTA**

Los nombres de packages, directorios y archivos pueden cambiar en función de lo que el alumno elija no es forzoso mantener los nombre recomendados

4.3 While - Ciclos



¿Qué es un WHILE?

- ★ Es un **ciclo (Loop)** que repite código mientras una condición sea verdadera.
- ★ No sabes exactamente cuántas veces se va a ejecutar.
- ★ Se usa cuando la repetición depende de una condición dinámica.

¿Por qué un Tester debería saber usar FOR?

- ★ Para esperar condiciones dinámicas (ej. que un elemento aparezca).
- ★ Para validar procesos que dependen de estados cambiantes.
- ★ Porque muchas automatizaciones usan lógica repetitiva basada en condiciones.

For VS While

- ★ **for** → repeticiones controladas y definidas.
- ★ **while** → repite hasta que algo cambie.

```
// Ejemplo loop de números
int i = 1;

while (i <= 5) {
    System.out.println(i);
    i++;
}

// Ejemplo simulando esperar un objeto visible en pantalla
boolean elementoVisible = false;

while (!elementoVisible) {
    System.out.println("Elemento NO está visible...");
}
```



Actividad

Realiza la siguiente actividad



1. Ve al código de ejemplo del repositorio
2. Analiza el código de condicionales en el paquete
“src/test/java/cap04/loopsWhile”
3. Importa el código a tu computador, ya hemos visto opciones de esto en capítulos anteriores
4. Entiende el código, ejecutalo

**NOTA

Los nombres de packages, directorios y archivos pueden cambiar en función de lo que el alumno elija no es forzoso mantener los nombre recomendados

Actividad

Realiza la siguiente actividad



1. Abre IntelliJ
2. Comienza a crear códigos de ejemplo relacionados a FOR en el proyecto, package y archivo Java de tu elección, cree nuevos de ser necesario

****NOTA**

Los nombres de packages, directorios y archivos pueden cambiar en función de lo que el alumno elija no es forzoso mantener los nombre recomendados

Actividad

Realiza la siguiente actividad



Te sugerimos esta actividad

1. Ve a la página del curso y resuelve el [Quiz 05](#)
2. Estos ejercicios te ayudarán a mejorar tu skill de Java
3. Te recomendamos intentar resolverlos por tu cuenta sin depender de una IA

****NOTA**

Los nombres de packages, directorios y archivos pueden cambiar en función de lo que el alumno elija no es forzoso mantener los nombre recomendados

Quizz

Quizz

Realiza los siguientes

- ★ [Quiz 01](#)
- ★ [Quiz 02](#)
- ★ [Quiz 03](#)
- ★ [Quiz 04](#)
- ★ [Quiz 05](#)

Recuerda:

- ★ Trata de no memorizar orden de las respuestas, entiende, asimila y aprende
- ★ En la página del curso, todos los capítulos tienen su acceso directo a los quizzes



Recursos del curso

Recursos

En esta sección encontrarás recursos más importantes del curso

- ★ [Página del curso](#)
- ★ [Repositorio de Git](#)
- ★ [YouTube videos del curso](#)



Si deseas cursos personalizados u
otro servicio no dudes en
contactarnos

<https://testeracademy.com.mx/contacto>

Gracias

🧠 “No aprendemos para aprobar pruebas, aprendemos para transformar realidades.”



¡Hasta pronto!