

A blurred background image of a computer screen displaying code. The code is in a light blue/cyan color on a dark background. It includes comments and variable assignments for a 'mirror' object.

```
mirror_mod = modifier_ob.  
# Set mirror object to mirror_  
mirror_mod.mirror_object =  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False
```

Módulo 6

Strings y Condiciones

¿Qué es un String?

String

Un String es un conjunto de caracteres. No es una variable primitiva, pues está compuesta de por lo menos dos caracteres.

Veamos el siguiente ejemplo:

```
public static void main(String[] args) {  
    String game = "Super Mario World";  
}
```

```
public static void main(String[] args) {  
    String game = "Super Mario World";  
}
```

DATA

S	u	p	e	r		M	a	r	i	o		W	o	r	l	d	\0
---	---	---	---	---	--	---	---	---	---	---	--	---	---	---	---	---	----

INDEX

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
---	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----

Importante dos cosas:

- El conteo de los índices comienza en 0. El caracter 'S' es el caracter en el índice 0.
- Para indicar que el String termina, Java almacena el caracter NULL (\0) al final de la cadena.



Un `String` se puede declarar de tres formas distintas:

```
01 String g1;  
02 g1 = "Hello!";  
03  
04 String g2 = "Hello!";  
05  
06 String g3 = new String("Hello!");
```

Las variables de tipo `String` pueden almacenar cualquier caracter de Unicode utilizando [representación UTF-16 \[16 bits\]](#).

Concatenar Strings

Dos strings pueden concatenarse utilizando el operador `+`.

Podemos concatenar valores y variables de diferentes tipos, como `Strings`, `int`, `double`, `char`, `boolean`, etc.

```
01 String greeting;  
02 greeting = "Hello ";  
03  
04 String sentence;  
05 sentence = greeting + "officer";  
06 System.out.println(sentence); //Prints "Hello officer"
```

Inicializar un String

Hay veces que es necesario inicializar un String con un valor vacío. Esto se puede hacer de la siguiente forma:

```
01 String s1 = "";
```

String length()

El método `length()` nos permite calcular el tamaño de un String. Este método devolverá un entero con la cantidad de caracteres que un String contenga.

```
01 String s1;  
02 int n;  
03  
04 s1 = "12345";  
05 n = s1.length();  
06 System.out.println(n); //Prints 5  
07  
08 s1 = "aabc12345";  
09 n = s1.length();  
10 System.out.println(n); //Prints 9  
11  
12 s1 = "";  
13 n = s1.length();  
14 System.out.println(n); //Prints 0
```


String substring()

El método `substring(int beginIndex, int endIndex)` regresa un recorte del String original. El substring comienza en el índice especificado como `beginIndex` y se extiende hasta `endIndex - 1`. El tamaño del String resultante será: `endIndex - beginIndex`.

NOTA: El conteo de posiciones comienza desde 0.

Ejemplo:

```
01 String s1 = "Monterrey, Nuevo León";
02 String ciudad = s1.substring(0,9);
03 String estado = s1.substring(11,21);
04
05 System.out.println(ciudad); //Prints "Monterrey"
06 System.out.println(estado); //Prints "Nuevo León"
07
08 String s2 = "smiles".substring(1, 5);
09 System.out.println(s2); //prints "mile"
```

String charAt()

El método `charAt(int index)` permite recuperar el caracter que se encuentra en la posición especificada, siendo 0 el índice del primer caracter, y `length()-1` índice del último.

```
01 String s1 = "The Jungle Book";
02 char c1 = s1.charAt(1);
03 System.out.println(c1); //prints 'h'
04
05 char c2 = s1.charAt(4);
06 System.out.println(c2); //prints 'J'
07
08 char c3 = s1.charAt(s1.length()-1);
09 System.out.println(c3); //prints 'k'
```

String trim()

El método `trim()` crea un nuevo string eliminado los espacios vacíos al inicio y al final.

```
01 String s1 = "   :)   ";  
02 String s2 = s1.trim();  
03 System.out.println(s2); //prints ":)"
```

String toLowerCase() / toUpperCase()

Los métodos `toLowerCase()` y `toUpperCase()` nos permiten procesar un texto para convertir una cadena a minúscula o mayúscula respectivamente. Estos métodos generan cadenas nuevas, por lo que debemos asignar el valor resultante a una variable de tipo `String`.

```
01 String s1 = "abCD";  
02 String s2 = "abCD";  
03  
04 String lowerCase = s1.toLowerCase();  
05 String upperCase = s2.toUpperCase();  
06  
07 System.out.println(lowerCase); //Prints "abcd"  
08 System.out.println(upperCase); //Prints "ABCD"
```

String replace()

El método `replace(char oldChar, char newChar)` sirve para reemplazar un caracter dentro del texto especificado por otro.

```
01 String s1 = "Bienvenido a la ciudad!";
02 String s2 = s1.replace('e','x');
03 System.out.println(s2); //Prints "Bixnvxnido a la ciudad!"
04
05 s2 = s2.replace('a','x');
06 System.out.println(s2); //Prints "Bixnvxnido x lx ciudxd!"
```

String trim()

El método `trim()` crea un nuevo string eliminado los espacios vacíos al inicio y al final.

```
01 String s1 = "   :)   ";  
02 String s2 = s1.trim();  
03 System.out.println(s2); //prints ":)"
```

Condiciones

IF

Existen instrucciones de control de flujo que nos permiten ejecutar selectivamente ciertos bloques de código. La primera instrucción es el **if**.

```
01 int result;  
02 if (denominator != 0) {  
03     result = numerator / denominator;  
04 }
```

Cuando la condición que está dentro de los paréntesis se cumple, entonces las instrucciones dentro del bloque **{ }** se ejecutarán.

IF-ELSE

Podemos incluir un segundo bloque, el **ELSE**, que se ejecuta cuando la condición NO se cumple.

Un bloque **ELSE** siempre debe estar ligado a un bloque **IF**.

```
01  int result;  
02  
03  if (denominator == 0) {  
04      System.out.println("Error! You cannot divide by zero.");  
05  } else {  
06      result = numerator / denominator;  
07  }
```

Expresiones booleanas

Dentro de un IF, debemos poner una expresión booleana. Estas pueden ser expresiones booleanas simples:

```
if (minutes > 60){  
    hour++;  
}  
  
if (letter == 'c') {  
    System.out.println("You selected option c");  
}
```

O expresiones booleanas complejas:

```
if ((hour >= 8) && (hour <= 22)){  
    System.out.println("The store is open!");  
}  
  
if (letter == 'a' || letter == 'e' || letter == 'i' || letter == 'o' || letter == 'u' ){  
    System.out.println("Letter is a vocal!");  
}
```

Name	Java Notation	Java Examples
Logical <i>and</i>	&&	(sum > min) && (sum < max)
Logical <i>or</i>		(answer == 'y') (answer == 'Y')
Logical <i>not</i>	!	!(number < 0)

Value of A	Value of B	Value of $A \ \&\& \ B$	Value of $A \ \ B$	Value of $! \ (A)$
true	true	true	true	false
true	false	false	true	false
false	true	false	true	true
false	false	false	false	true

Comparación de variables primitivas

Para comparar la **igualdad** de dos valores primitivos, podemos utilizar el operador de igualdad (**==**).

Por ejemplo:

```
01  if (a == 3) {  
02      System.out.println("a equals 3");  
03  }
```

Comparación de Strings

Para comparar dos variables de tipo String, NO podemos utilizar el operador (`==`), pues **no son variables primitivas**.

Para esto, debemos utilizar el método `equals()`.

```
01 String s1 = "hola";
02 String s2 = "adios";
03
04 if (s1.equals(s2)){
05     System.out.println("Iguales!");
06 } else {
07     System.out.println("Diferentes!");
08 }
```

Comparación de Strings

También podemos utilizar el método `equalsIgnoreCase()` si queremos verificar comparar sin considerar mayúsculas o minúsculas.

```
01 String s1 = "hola";
02 String s2 = "HOLA";
03
04 if (s1.equalsIgnoreCase(s2)){
05     System.out.println("Iguales!");
06 } else {
07     System.out.println("Diferentes!");
08 }
```

Ejercicio!

Ejemplo

Monkey Trouble!

Tenemos dos changos:

- Si los dos changos están sonriendo, estamos en problemas
- Si los dos changos están serios, estamos en problemas.

Diseña un programa de Java que permita modelar este ejercicio.



```
01 import java.util.Scanner;
02
03 public class MonkeyTrouble {
04     public static void main(String[] args) {
05         Scanner teclado = new Scanner(System.in);
06
07         System.out.print("Is the first monkey smiling? (true/false): ");
08         boolean monkey1 = teclado.nextBoolean();
09
10         System.out.print("Is the second monkey smiling? (true/false): ");
11         boolean monkey2 = teclado.nextBoolean();
12
13         if ((monkey1 == true && monkey2 == true) || (monkey1 == false && monkey2 == false)) {
14             System.out.println("Look out! The monkeys are planning something!");
15         }
16         else {
17             System.out.println("Don't worry, everything is OK");
18         }
19         teclado.close();
20     }
21 }
```

```
01 import java.util.Scanner;
02
03 public class MonkeyTrouble {
04     public static void main(String[] args) {
05         Scanner teclado = new Scanner(System.in);
06
07         System.out.print("Is the first monkey smiling? (true/false): ");
08         boolean monkey1 = teclado.nextBoolean();
09
10         System.out.print("Is the second monkey smiling? (true/false): ");
11         boolean monkey2 = teclado.nextBoolean();
12
13         if ((monkey1 == true && monkey2 == true) || (monkey1 == false && monkey2 == false)) {
14             System.out.println("Look out! The monkeys are planning something!");
15         }
16         else {
17             System.out.println("Don't worry, everything is OK");
18         }
19         teclado.close();
20     }
21 }
```

Imports, definición de la clase y main

```

01 import java.util.Scanner;
02
03 public class MonkeyTrouble {
04     public static void main(String[] args) {
05         Scanner teclado = new Scanner(System.in);
06
07         System.out.print("Is the first monkey smiling? (true/false): ");
08         boolean monkey1 = teclado.nextBoolean();
09
10         System.out.print("Is the second monkey smiling? (true/false): ");
11         boolean monkey2 = teclado.nextBoolean();
12
13         if ((monkey1 == true && monkey2 == true) || (monkey1 == false && monkey2 == false)) {
14             System.out.println("Look out! The monkeys are planning something!");
15         }
16         else {
17             System.out.println("Don't worry, everything is OK");
18         }
19         teclado.close();
20     }
21 }

```

Preguntar y capturar el estado de los dos monos. Al ser variables boolean, hay que capturar TRUE / FALSE

```

01 import java.util.Scanner;
02
03 public class MonkeyTrouble {
04     public static void main(String[] args) {
05         Scanner teclado = new Scanner(System.in);
06
07         System.out.print("Is the first monkey smiling? (true/false): ");
08         boolean monkey1 = teclado.nextBoolean();
09
10         System.out.print("Is the second monkey smiling? (true/false): ");
11         boolean monkey2 = teclado.nextBoolean();
12
13         if ((monkey1 == true && monkey2 == true) || (monkey1 == false && monkey2 == false)) {
14             System.out.println("Look out! The monkeys are planning something!");
15         }
16         else {
17             System.out.println("Don't worry, everything is OK");
18         }
19         teclado.close();
20     }
21 }

```

Si ambos monos, o ninguno, están sonriendo, entonces imprimimos en pantalla un mensaje de alerta.

```
01 import java.util.Scanner;
02
03 public class MonkeyTrouble {
04     public static void main(String[] args) {
05         Scanner teclado = new Scanner(System.in);
06
07         System.out.print("Is the first monkey smiling? (true/false): ");
08         boolean monkey1 = teclado.nextBoolean();
09
10         System.out.print("Is the second monkey smiling? (true/false): ");
11         boolean monkey2 = teclado.nextBoolean();
12
13         if ((monkey1 == true && monkey2 == true) || (monkey1 == false && monkey2 == false)) {
14             System.out.println("Look out! The monkeys are planning something!");
15         }
16         else {
17             System.out.println("Don't worry, everything is OK");
18         }
19         teclado.close();
20     }
21 }
```

De lo contrario, imprimimos que todo está OK

```
01 import java.util.Scanner;
02
03 public class MonkeyTrouble {
04     public static void main(String[] args) {
05         Scanner teclado = new Scanner(System.in);
06
07         System.out.print("Is the first monkey smiling? (true/false): ");
08         boolean monkey1 = teclado.nextBoolean();
09
10         System.out.print("Is the second monkey smiling? (true/false): ");
11         boolean monkey2 = teclado.nextBoolean();
12
13         if ((monkey1 == true && monkey2 == true) || (monkey1 == false && monkey2 == false)) {
14             System.out.println("Look out! The monkeys are planning something!");
15         }
16         else {
17             System.out.println("Don't worry, everything is OK");
18         }
19         teclado.close();
20     }
21 }
```

Se cierra el objeto teclado.

Como la variable monkey es una variable de tipo boolean que ya almacena un valor TRUE/FALSE, podemos ahorrarnos la comparación.

```
01 if ((monkey1 == true && monkey2 == true) || (monkey1 == false && monkey2 == false)) {
02     System.out.println("Look out! The monkeys are planning something!");
03 }
04 else {
05     System.out.println("Don't worry, everything is OK");
06 }
```

Y utilizar la siguiente expresión:

```
01 if ((monkey1 && monkey2) || (!monkey1 && !monkey2)) {
02     System.out.println("Look out! The monkeys are planning something!");
03 }
04 else {
05     System.out.println("Don't worry, everything is OK");
06 }
```

Ambas expresiones son equivalentes.