



*Universidad Tecnológica de Tijuana*

**TEMA:**

Sentencias de Block

**PRESENTADO POR:**

Sanchez Zamudio Guadalupe

**GRUPO:**

9B BIS

**MATERIA:**

Wearables

**PROFESOR:**

Ray Brunett Parra Galaviz

Tijuana, Baja California, 2 de octubre del 2024

## 1. Sentencias de Interacción

Estas se refieren a la forma en que el usuario o el sistema interactúa con el programa. En Kotlin, puedes usar las siguientes para obtener o mostrar datos:

- `print()`: Muestra texto o datos en la consola sin un salto de línea al final.

```
```kotlin  
  
print("Hola")  
...`
```

- `println()`: Muestra texto o datos en la consola con un salto de línea al final.

```
```kotlin  
  
println("Hola, Kotlin!")  
...`
```

- `readLine()`: Lee una entrada del usuario desde la consola. Devuelve un valor de tipo `String` o `null`.

```
```kotlin  
  
val nombre = readLine()  
  
println("Tu nombre es: $nombre")  
...`
```

## 2. Sentencias de Control

Controlan el flujo del programa, permitiendo que se realicen diferentes acciones en función de condiciones o repeticiones.

a. Condicionales

- ``if/else``: Evalúa una condición y ejecuta un bloque de código si es verdadera, o un bloque alternativo si es falsa.

```
```kotlin

val edad = 18

if (edad >= 18) {

    println("Eres mayor de edad")

} else {

    println("Eres menor de edad")

}

```
```

- ``when``: Similar a un ``switch`` en otros lenguajes. Evalúa un valor y ejecuta un bloque dependiendo del caso.

```
```kotlin

val dia = 3

when (dia) {

    1 -> println("Lunes")

    2 -> println("Martes")

    3 -> println("Miércoles")

    else -> println("Día no válido")

}

```
```

b. Bucles o ciclos

- `for`: Recorre rangos, arrays o listas.

```
```kotlin  
  
for (i in 1..5) {  
    println(i)  
}  
...
```

- `while`: Ejecuta el bloque de código mientras la condición sea verdadera.

```
```kotlin  
  
var contador = 0  
  
while (contador < 5) {  
    println(contador)  
    contador++  
}  
...
```

- `do/while`: Similar a `while`, pero ejecuta el bloque al menos una vez antes de verificar la condición.

```
```kotlin  
  
var contador = 0  
  
do {  
    println(contador)  
    contador++  
}
```

```
} while (contador < 5)
...

```

### 3. Funciones

Son bloques de código que pueden recibir parámetros, realizar una tarea y devolver un resultado. Te ayudan a reutilizar código y mantener tu programa organizado.

- Declarar una función:

```
```kotlin
fun saludar(nombre: String) {
    println("Hola, $nombre!")
}

```

```
saludar("Juan")

```

```
...

```

- Funciones con retorno:

```
```kotlin
fun sumar(a: Int, b: Int): Int {
    return a + b
}

```

```
val resultado = sumar(5, 3)

```

```
println("La suma es: $resultado")
```

```
...
```

- Funciones de una sola línea: En Kotlin, puedes simplificar funciones cortas.

```
```kotlin
```

```
fun multiplicar(a: Int, b: Int) = a * b
```

```
...
```