El uso de Kotlin Ranges nos permite trabajar con conceptos fundamentales como las sentencias de control pero utilizando un lenguaje más moderno como Kotlin a la hora de simplificar la sintaxis y compactar el código. Vamos a ver un ejemplo sencillo partiendo de un caso de if/else clásico de manejo de notas . Para ello vamos a construir en un primer lugar el código en Java .

```
package com.arquitecturajava;

public class Principal5 {

	public static void main(String[] args) {

		int nota = 3;

		if (nota >= 0 && nota < 2) {
			System.out.println("muy mal");

		} else if (nota > 2 && nota <= 4) {

			System.out.println("has suspendido");
		} else if (nota >= 5 && nota < 8) {
			System.out.println("buen examen");

		} else {
			System.out.println("genial");
		}

	}
}
```

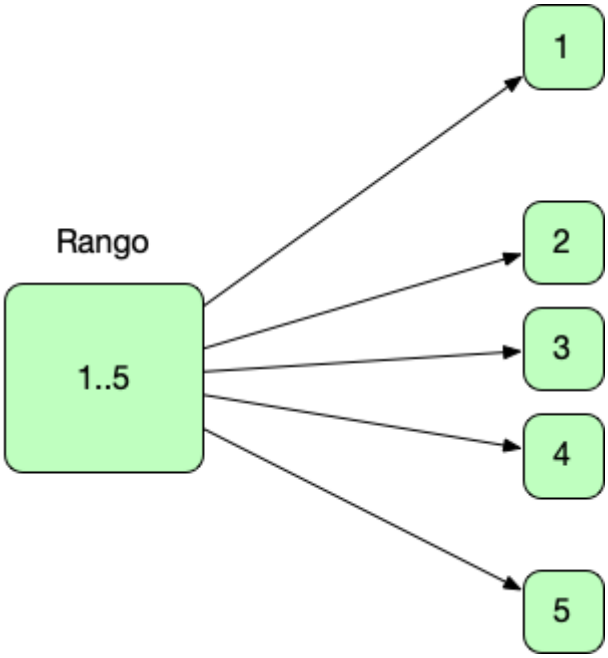
En este caso estamos ante un bloque if/else que se encarga de mostrarnos por la consola una nota dependiendo del valor inicial en este caso imprime el valor de has suspendido:

```
<terminated> Principal5 [Java Application] /Library/Java/
has suspendido
```

Kotlin Ranges y estructura de control

Las estructras de control en kotlin se pueden generar de una forma más compacta a través del uso de rangos. Un rango no es ni más ni menos que un conjunto de valores soportados en este caso números enteros.

```
val numeros:Int:Range=1..5
```



Este concepto de Rango se puede encajar perfectamente dentro de nuestra estructura if/else pero usando Kotlin como lenguaje.

```
var nota = 3

var mensaje=if (nota in 1..2) {
    "muy mal"
} else if (nota in 3..4) {
    "mal"
```

```
} else if (nota in 5..7) {
    "ok"
} else {
    "genial"
}
print(mensaje)
```

Acabamos de construir una estructura que nos permite generar de una forma más compacta nuestro código y sobre todo más sencilla de entender ya que usamos if/else combinado con rangos. Esta sintaxis la podemos mejorar y compactar todavía más usando la clausula when de Kotlin.

```
var nota = 9

var mensaje=when (nota) {
  in 1..2->"muy mal"
  in 3..4->"mal"
  in 5..7->"ok"
 else->"genial"
}
print(mensaje)
```

Es aquí donde la sencillez y elegancia de Kotlin destaca y un código que en Java son unas 23 lineas y que cuesta entender de un primer vistazo. Kotlin lo reduce a 9 y lo clarifica mucho más:

Otros artículos relacionados

- Kotlin Destructuring
- Kotlin Delegación buenas prácticas.
- Curso Kotlin y buenas prácticas