



UNIVERSIDAD
IBEROAMERICANA
CIUDAD DE MÉXICO ®

**P5. Fundamentos de programación en
Kotlin**

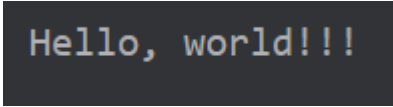
Alumna: Daniela Mendez Ramirez

Número de Cuenta: 258331-9

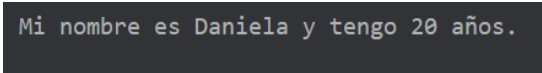
Aplicaciones de Aplicaciones

Profesor: Omar Vázquez González

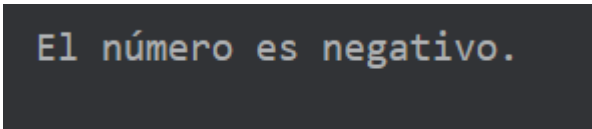
Fecha de Entrega: 16 de febrero de 2025

Código	Resultado
<pre>/** * You can edit, run, and share this code. * play.kotlinlang.org */ fun main() { println("Hello, world!!!") }</pre>	

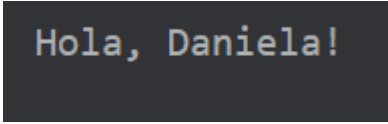
2. Variables

Código	Resultado
<pre>fun main() { val nombre: String = "Daniela" var edad: Int = 20 println("Mi nombre es \$nombre y tengo \$edad años.") }</pre>	

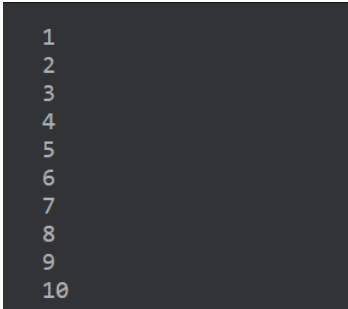
3. Condiciones con IF

Código	Resultado
<pre>fun main() { val numero = -5 if (numero > 0) { println("El número es positivo.") } else if (numero < 0) { println("El número es negativo.") } else { println("El número es cero.") } }</pre>	

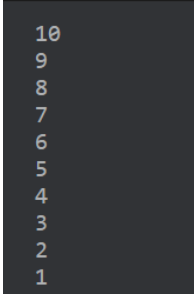
4. Funciones

Código	Resultado
<pre>fun saludar(nombre: String) { println("Hola, \$nombre!") } fun main() { saludar("Daniela") }</pre>	

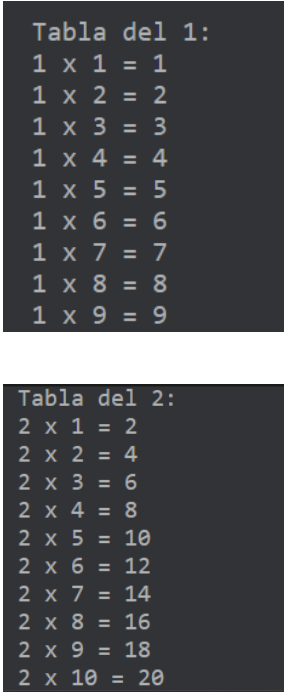
5. Ciclo For

Código	Resultado
<pre>fun main() { for (i in 1..10) { println(i) } }</pre>	

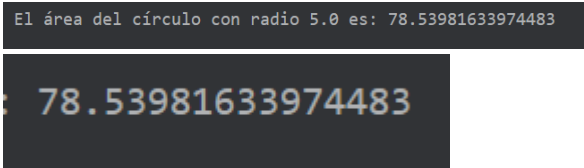
6. Ciclo While

Código	Resultado
<pre>fun main() { for (i in 1..10) { println(i) } }</pre>	

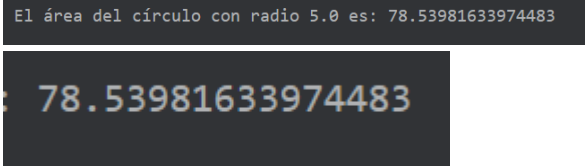
7. Tablas de multiplicar

Código	Resultado
<pre>fun imprimirTablas() { for (i in 1..10) { println("Tabla del \$i:") for (j in 1..10) { println("\$i x \$j = \${i * j}") } println() // Línea en blanco para separar las tablas } } fun main() { imprimirTablas() }</pre>	 <p>The screenshot displays the output of the Kotlin program. It shows two multiplication tables. The first table is for the number 1, with rows from 1 x 1 to 1 x 9. The second table is for the number 2, with rows from 2 x 1 to 2 x 10. The output is formatted with a title for each table and the results of the multiplication.</p> <pre>Tabla del 1: 1 x 1 = 1 1 x 2 = 2 1 x 3 = 3 1 x 4 = 4 1 x 5 = 5 1 x 6 = 6 1 x 7 = 7 1 x 8 = 8 1 x 9 = 9 Tabla del 2: 2 x 1 = 2 2 x 2 = 4 2 x 3 = 6 2 x 4 = 8 2 x 5 = 10 2 x 6 = 12 2 x 7 = 14 2 x 8 = 16 2 x 9 = 18 2 x 10 = 20</pre>

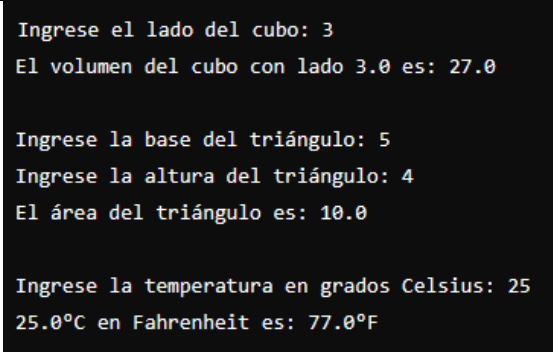
8. Área de un Círculo Por Valor

Código	Resultado
<pre>fun calcularAreaPorValor(radio: Double): Double { return Math.PI * radio * radio } fun main() { val radio = 5.0 val area = calcularAreaPorValor(radio) println("El área del círculo con radio \$radio es: \$area") }</pre>	 <p>The screenshot shows the output of the Kotlin program. It displays the calculated area of a circle with a radius of 5.0. The output is formatted with a title and the result of the calculation.</p> <pre>El área del círculo con radio 5.0 es: 78.53981633974483 78.53981633974483</pre>

9. Área de un Círculo Por Referencia

Código	Resultado
<pre> class Circulo(var radio: Double, var area: Double = 0.0) fun calcularAreaPorReferencia(circulo: Circulo) { circulo.area = Math.PI * circulo.radio * circulo.radio } fun main() { val miCirculo = Circulo(5.0) calcularAreaPorReferencia(miCirculo) println("El área del círculo con radio \${miCirculo.radio} es: \${miCirculo.area}") } </pre>	 <p>El área del círculo con radio 5.0 es: 78.53981633974483</p>

10. Area, Celcius

Código	Resultado
<pre> fun volumenCubo(lado: Double): Double { return lado * lado * lado } fun areaTriangulo(base: Double, altura: Double): Double { return (base * altura) / 2 } fun celsiusAFahrenheit(celsius: Double): Double { return (celsius * 9 / 5) + 32 } fun main() { val ladoCubo = 3.0 </pre>	 <p>Ingrese el lado del cubo: 3 El volumen del cubo con lado 3.0 es: 27.0</p> <p>Ingrese la base del triángulo: 5 Ingrese la altura del triángulo: 4 El área del triángulo es: 10.0</p> <p>Ingrese la temperatura en grados Celsius: 25 25.0°C en Fahrenheit es: 77.0°F</p>

<pre>println("El volumen del cubo con lado \$ladoCubo es: \${volumenCubo(ladoCubo)}") val baseTriangulo = 5.0 val alturaTriangulo = 4.0 println("El área del triángulo con base \$baseTriangulo y altura \$alturaTriangulo es: \${areaTriangulo(baseTriangulo, alturaTriangulo)}") val temperaturaCelsius = 25.0 println("\$temperaturaCelsius°C en Fahrenheit es: \${celsiusAFahrenheit(temperaturaCelsius) }°F") }</pre>	
--	--

11. Una función que compute la sumatoria de los primeros n números, dado n.

Código	Resultado
<pre>fun sumatoria(n: Int): Int { return (n * (n + 1)) / 2 } fun main() { print("Ingrese un número n: ") val n = readLine()!!.toInt() println("La sumatoria de los primeros \$n números es: \${sumatoria(n)}") }</pre>	<pre>Ingrese un número n: 5 La sumatoria de los primeros 5 números es: 15</pre>

12. Fibonacci

Código	Resultado
<pre>fun fibonacci(n: Int): Int { if (n == 0) return 0 if (n == 1) return 1 return fibonacci(n - 1) + fibonacci(n - 2) } fun main() { print("Ingrese un número n: ") val n = readLine()!!.toInt() println("El término \$n de la serie Fibonacci es: \${fibonacci(n)}") }</pre>	<pre>Ingrese un número n: 5 El término 5 de la serie Fibonacci es: 5</pre>

13. Auto

Código	Resultado
<pre>// Definimos la clase Auto con dos propiedades: marca y velocidad class Auto(val marca: String, val velocidad: Int) { // Método para calcular cuántos kilómetros avanza en un tiempo dado fun avanzar(tiempo: Int) { val distancia = velocidad * tiempo println("El auto de marca \$marca recorrió \$distancia km en \$tiempo horas.") } } fun main() { // Creamos autos con diferentes marcas y velocidades val auto1 = Auto("Toyota", 100) val auto2 = Auto("Ferrari", 200)</pre>	<pre>El auto de marca Toyota recorrió 200 km en 2 horas. El auto de marca Ferrari recorrió 400 km en 2 horas.</pre>

<pre>// Mostramos cuántos kilómetros avanzan en 2 horas auto1.avanzar(2) auto2.avanzar(2) }</pre>	
---	--

14. Alumnos

Código	Resultado
<pre>// Clase Materia que representa el nombre de la materia class Materia(val nombre: String) // Clase Calificación que asocia una materia con una calificación class Calificacion(val materia: Materia, val nota: Double) // Clase Alumno con nombre y lista de calificaciones class Alumno(val nombre: String) { private val calificaciones = mutableListOf<Calificacion>() // Método para agregar una calificación fun agregarCalificacion(materia: Materia, nota: Double) { calificaciones.add(Calificacion(materia, nota)) } // Método para calcular el promedio del alumno fun calcularPromedio(): Double { if (calificaciones.isEmpty()) return 0.0</pre>	<pre>Materia: Matemáticas - Nota: 8.5 Materia: Español - Nota: 9.0 Materia: Ciencias - Nota: 7.5 Promedio: 8.333333333333334</pre>


```

        return calificaciones.map
{ it.nota }.average()
    }

    // Método para mostrar las calificaciones
del alumno
    fun mostrarCalificaciones() {
        println("Calificaciones de $nombre:")
        calificaciones.forEach {
            println("Materia: ${it.materia.nombre}
- Nota: ${it.nota}")
        }
        println("Promedio:
${calcularPromedio()}\n")
    }
}

fun main() {
    // Creamos materias
    val matematicas =
Materia("Matemáticas")
    val espanol = Materia("Español")
    val ciencias = Materia("Ciencias")

    // Creamos un alumno
    val alumno1 = Alumno("Lorenzo Grebe")

    // Agregamos calificaciones

    alumno1.agregarCalificacion(matematicas
, 8.5)
    alumno1.agregarCalificacion(espanol,
9.0)
    alumno1.agregarCalificacion(ciencias,
7.5)

```

<pre>// Mostramos las calificaciones y el promedio del alumno alumno1.mostrarCalificaciones() }</pre>	
---	--

Conclusión

A través de estos ejercicios en Kotlin, pude reforzar conceptos clave como estructuras de control, funciones y programación orientada a objetos. Implementé ciclos for y while para resolver problemas matemáticos, además de utilizar funciones con paso por valor y referencia para cálculos como el área de un círculo y la conversión de grados. También trabajé con clases y objetos, lo que me permitió organizar mejor la información en programas como el control de calificaciones y la simulación de autos. Además, usé listas mutables para almacenar y manipular datos de manera dinámica. En general, estos ejercicios me ayudaron a entender la flexibilidad y legibilidad de Kotlin, destacando su eficiencia para desarrollar código limpio y estructurado.