INSTITUCIÓN:

Universidad Don Bosco.

FACULTAD:

Ingeniería.

ASIGNATURA:

Desarrollo de Software para Móviles.

DOCENTE:

Ing. Alexander Alberto Sigüenza Campos.

ACTIVIDAD:

Trabajo de investigación 10%.

INTEGRANTES:

García Aparicio Sara Daniela	GA190843.
Molina Jovel Daniel Adonay	MJ150737.
Mejía Olmedo Edwin Adonay	MO172006.
Nolasco Umanzor Ronald Alexander	NU172013.

FECHA:

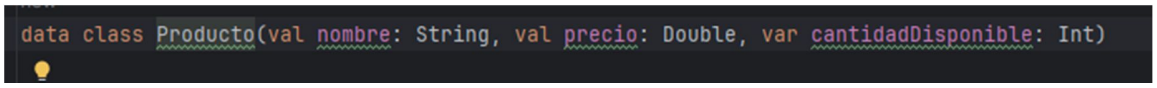
San Salvador, 03/03/2024.

Documentación de la Aplicación de Carrito de Compras en Kotlin.

Estructura del Código.

La aplicación está estructurada en tres archivos principales: **Carrito.kt**, **Producto.kt** y **Main.kt**, los cuales interactúan entre sí para permitir al usuario gestionar un carrito de compras.

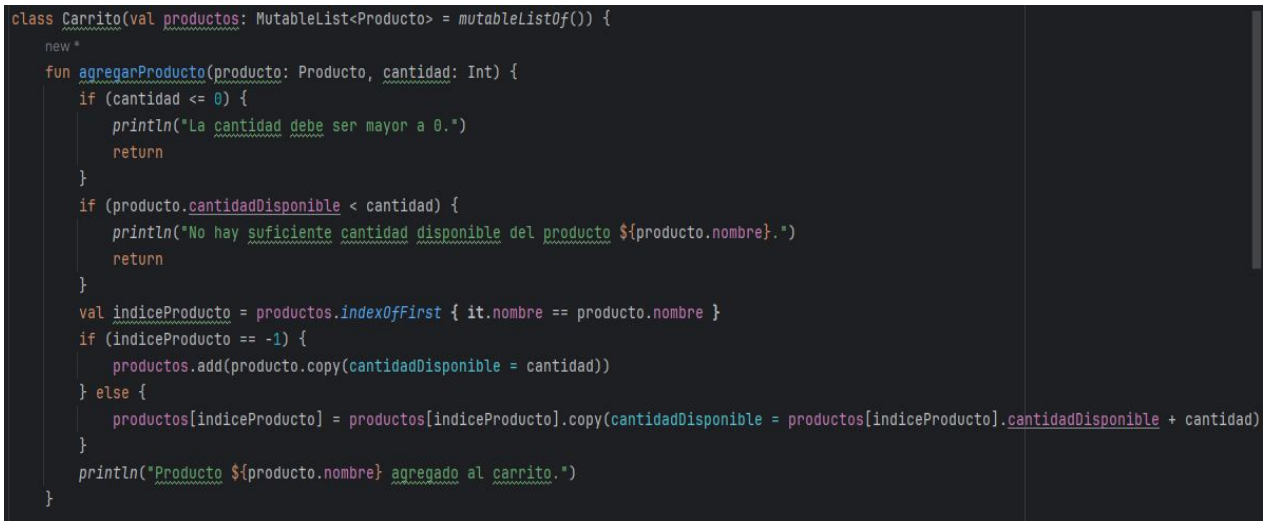
1. **Producto.kt**: Define la clase **Producto**, la cual modela los productos que se pueden añadir al carrito. Cada producto tiene un **nombre**, un **precio** y una **cantidad Disponible**.



```
data class Producto(val nombre: String, val precio: Double, var cantidadDisponible: Int)
```

2. **Carrito.kt**: Contiene la clase **Carrito**, que administra las operaciones del carrito de compras, como agregar y eliminar productos, mostrar el contenido del carrito y generar una factura.

    - **Agregar Producto**: Agrega un producto al carrito. Si el producto ya está en el carrito, aumenta su cantidad.



```
class Carrito(val productos: MutableList<Producto> = mutableListOf()) {
    new *
    fun agregarProducto(producto: Producto, cantidad: Int) {
        if (cantidad <= 0) {
            println("La cantidad debe ser mayor a 0.")
            return
        }
        if (producto.cantidadDisponible < cantidad) {
            println("No hay suficiente cantidad disponible del producto ${producto.nombre}.")
            return
        }
        val indiceProducto = productos.indexOfFirst { it.nombre == producto.nombre }
        if (indiceProducto == -1) {
            productos.add(producto.copy(cantidadDisponible = cantidad))
        } else {
            productos[indiceProducto] = productos[indiceProducto].copy(cantidadDisponible = productos[indiceProducto].cantidadDisponible + cantidad)
        }
        println("Producto ${producto.nombre} agregado al carrito.")
    }
```

- **Eliminar Producto**: Elimina un producto del carrito por su nombre.

```
fun eliminarProducto(nombreProducto: String) {
    val indiceProducto = productos.indexOfFirst { it.nombre == nombreProducto }
    if (indiceProducto == -1) {
        println("No se encontró el producto $nombreProducto en el carrito.")
        return
    }
    productos.removeAt(indiceProducto)
    println("Producto $nombreProducto eliminado del carrito.")
}
```

- **Mostrar Carrito**: Muestra los productos en el carrito y el total a pagar.

```
fun mostrarCarrito() {
    if (productos.isEmpty()) {
        println("El carrito está vacío.")
        return
    }
    println("Productos en el carrito:")
    var totalGeneral = 0.0
    productos.forEach { producto ->
        val precioTotal = producto.precio * producto.cantidadDisponible
        totalGeneral += precioTotal
        println("${producto.nombre}, Cantidad: ${producto.cantidadDisponible}, Precio unitario: ${producto.precio}, Precio total: $precioTotal")
    }
    println("Total general del carrito: $totalGeneral")
}
```

- **Generar Factura**: Genera un resumen detallado de la compra.

```
    fun generarFactura(): String {
        if (productos.isEmpty()) {
            return "El carrito está vacío."
        }
        var factura = "Factura:\n"
        var totalGeneral = 0.0
        productos.forEach { producto ->
            val precioTotal = producto.precio * producto.cantidadDisponible
            totalGeneral += precioTotal
            factura += "Producto: ${producto.nombre}, Cantidad: ${producto.cantidadDisponible}, Precio unitario: ${producto.precio}, " +
                    "Precio total: $precioTotal\n"
        }
        factura += "Total general de la compra: $totalGeneral"
        return factura
    }
}
```

3. **Main.kt**: Es el punto de entrada de la aplicación. Define la función **main** que interactúa con el usuario a través de la consola, permitiendo realizar diversas acciones:

```kotlin
fun main() {
    val productos = listOf(
        Producto( nombre: "Producto 1",  precio: 10.0,  cantidadDisponible: 5),
        Producto( nombre: "Producto 2",  precio: 20.0,  cantidadDisponible: 3),
        Producto( nombre: "Producto 3",  precio: 30.0,  cantidadDisponible: 1)
    )
    val carrito = Carrito()
    val scanner = Scanner(System.`in`)
    var continuar = true
    while (continuar) {
        println("\nOpciones principales:")
        println("1. Mostrar productos")
        println("2. Mostrar carrito")
        println("3. Generar factura")
        println("4. Salir")
        print("Seleccione una opción: ")
        when (scanner.nextInt()) {
            1 -> mostrarProductos(productos, carrito, scanner)
            2 -> mostrarCarrito(carrito, scanner)
            3 -> println(carrito.generarFactura())
            4 -> {
                println("Gracias por usar la aplicación.")
                continuar = false
            }
            else -> println("Opción no válida. Intente de nuevo.")
        }
    }
}
```

- Mostrar los productos disponibles.

```
fun mostrarProductos(productos: List<Producto>, carrito: Carrito, scanner: Scanner) {
    var regresar = false
    while (!regresar) {
        println("\nProductos disponibles:")
        listarProducto(productos)
        println("\n**************************************")
        println("**** 1. Agregar producto al carrito ****")
        println("**** 2. Regresar al menú principal  ****")
        println("**** 3. Salir                       ****")
        println("**************************************")
        print("Seleccione una opción: ")
        when (scanner.nextInt()) {
            1 -> {
                listarProducto(productos)
                print("Ingrese el número del producto: ")
                val numProducto = scanner.nextInt()
                if (numProducto in 1 ≤ .. ≤ productos.size) {
                    val producto = productos[numProducto - 1]
                    print("Ingrese la cantidad a comprar: ")
                    var cantidad = scanner.nextInt()
                    while (cantidad > producto.cantidadDisponible) {
                        println("No hay suficiente stock para el producto ${producto.nombre}. Cantidad disponible: " +
                                "${producto.cantidadDisponible}. " + "Ingrese una cantidad válida: ")
                        cantidad = scanner.nextInt()
                    }
```

- Agregar un producto al carrito.

```
                    carrito.agregarProducto(producto.copy(), cantidad)
                    producto.cantidadDisponible -= cantidad // Disminuir la cantidad disponible del producto
                } else {
                    println("Número de producto no válido.")
                }
            }
            2 -> regresar = true
            3 -> {
                println("Gracias por usar la aplicación.")
                System.exit( status: 0)
            }
            else -> println("Opción no válida. Intente de nuevo.")
        }
    }
}
```

- Mostrar los productos en el carrito.

```
fun mostrarCarrito(carrito: Carrito, scanner: Scanner) {
    var regresar = false
    while (!regresar) {
        println("\nProductos seleccionados:")
        carrito.mostrarCarrito()
        println("\n**************************************")
        println("****1. Editar cantidad de producto ****")
        println("****2. Eliminar producto             ****")
        println("****3. Regresar al menú principal    ****")
        println("*****4. Salir                        ****")
        println("**************************************")
        print("Seleccione una opción: ")
```

- Editar la cantidad de un producto en el carrito.

```
when (scanner.nextInt()) {
    1 -> {
        carrito.mostrarCarrito()
        println("Ingrese el número del producto para editar:")

        val numProducto = scanner.nextInt()
        if (numProducto in 1 ≤ .. ≤ carrito.productos.size) {
            val producto = carrito.productos[numProducto - 1]
            print("Ingrese la nueva cantidad para \"${producto.nombre}\": ")
            val cantidad = scanner.nextInt()
```

- Eliminar un producto del carrito.

```
            if (cantidad > 0) {
                carrito.eliminarProducto(producto.nombre)
                carrito.agregarProducto(producto, cantidad)
            } else if (cantidad <= 0) {
                carrito.eliminarProducto(producto.nombre)
                println("Producto eliminado del carrito.")
            }
        } else {
            println("Número de producto no válido.")
        }
    }
    2 -> {
        println("Ingrese el número del producto para eliminar:")
        carrito.mostrarCarrito()
        val numProducto = scanner.nextInt()
        if (numProducto in 1 ≤ .. ≤ carrito.productos.size) {
            val producto = carrito.productos[numProducto - 1]
            carrito.eliminarProducto(producto.nombre)

            println("Producto \"${producto.nombre}\" eliminado del carrito.")
        } else {
            println("Número de producto no válido.")
        }
    }
```

```
                3 -> regresar = true
                4 -> {
                    println("Gracias por usar la aplicación.")
                    System.exit( status: 0)
                }
                else -> println("Opción no válida. Intente de nuevo.")
            }
        }
    }
```

- Generar y mostrar una factura.

```
fun listarProducto(productos: List<Producto>){
    productos.forEachIndexed { index, producto ->
        println("${index + 1}. \"${producto.nombre}\", ${producto.precio}, ${producto.cantidadDisponible}")
    }
}
```

- Salir de la aplicación.

Cómo Ejecutar la Aplicación

Para ejecutar la aplicación, necesitas tener Kotlin instalado en tu máquina o utilizar un entorno de desarrollo que soporte Kotlin, como IntelliJ IDEA. Aquí están los pasos básicos para ejecutar la aplicación:

1. **Configuración del Entorno**: Asegúrate de que Kotlin esté correctamente instalado en tu sistema o que tu IDE esté configurado para proyectos Kotlin.
2. **Clonar/Descargar el Código**: Obtén el código fuente de los tres archivos: **Producto.kt**, **Carrito.kt** y **Main.kt**.
3. **Ejecutar la Aplicación**:
    - Si estás utilizando un IDE como IntelliJ IDEA, puedes simplemente abrir el proyecto y ejecutar el archivo **Main.kt**.
    - Si prefieres la línea de comandos, navega al directorio del proyecto y ejecuta **Kotlin Main.kt -include-runtime -d main.jar** seguido de **java -jar main.jar**.
4. **Interactuar con la Aplicación**: Una vez ejecutada, la aplicación mostrará un menú de opciones en la consola. Puedes seleccionar la acción que deseas realizar siguiendo las instrucciones y respondiendo a las solicitudes de entrada según sea necesario.

Notas Adicionales

- La aplicación es interactiva y se basa completamente en la entrada del usuario a través de la consola.
- La aplicación no persiste los datos, por lo que cualquier producto añadido o modificado se perderá al reiniciar la aplicación.