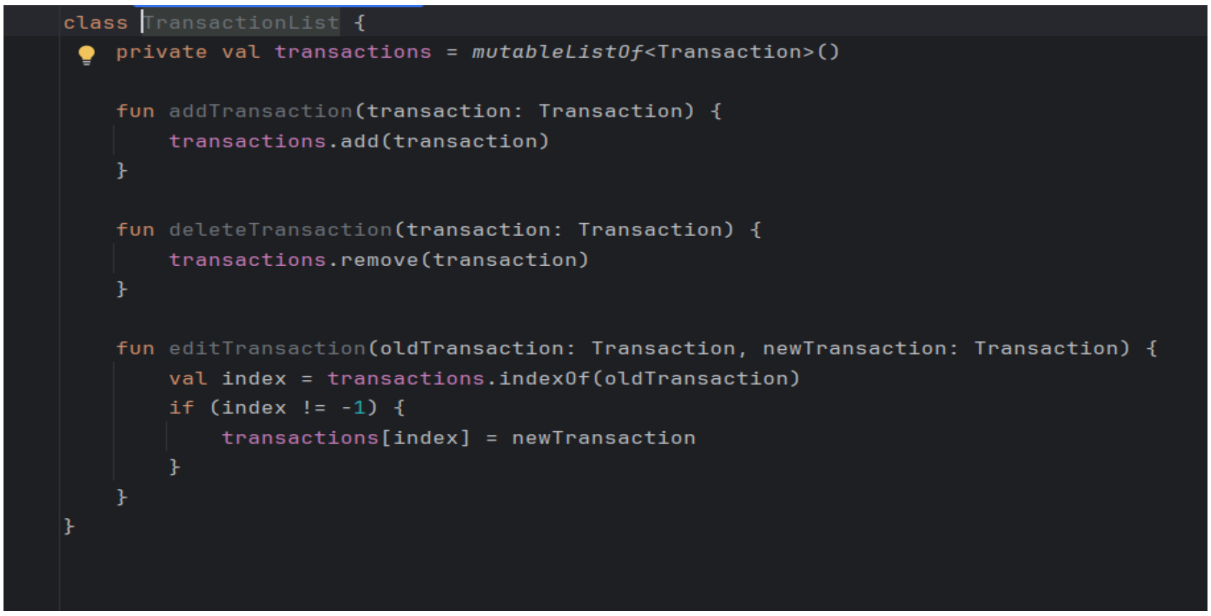
Sneha Walikar(Assignment 1_Day2)

Day 2: Functions and OOP Basics

Task 5: Write functions to add, delete, and edit transactions in a TransactionList class.



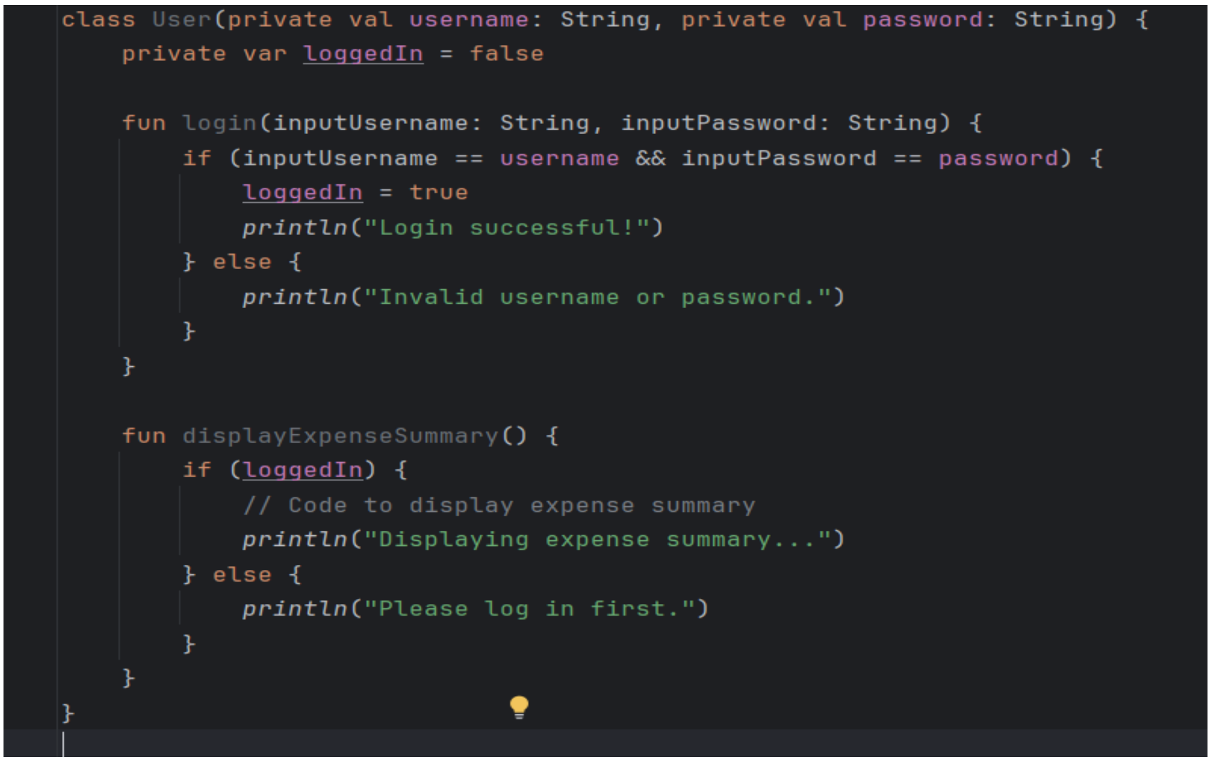
```
class TransactionList {
    private val transactions = mutableListOf<Transaction>()

    fun addTransaction(transaction: Transaction) {
        transactions.add(transaction)
    }

    fun deleteTransaction(transaction: Transaction) {
        transactions.remove(transaction)
    }

    fun editTransaction(oldTransaction: Transaction, newTransaction: Transaction) {
        val index = transactions.indexOf(oldTransaction)
        if (index != -1) {
            transactions[index] = newTransaction
        }
    }
}
```

Task 6: Develop a simple User class with methods to login and display a summary of expenses.

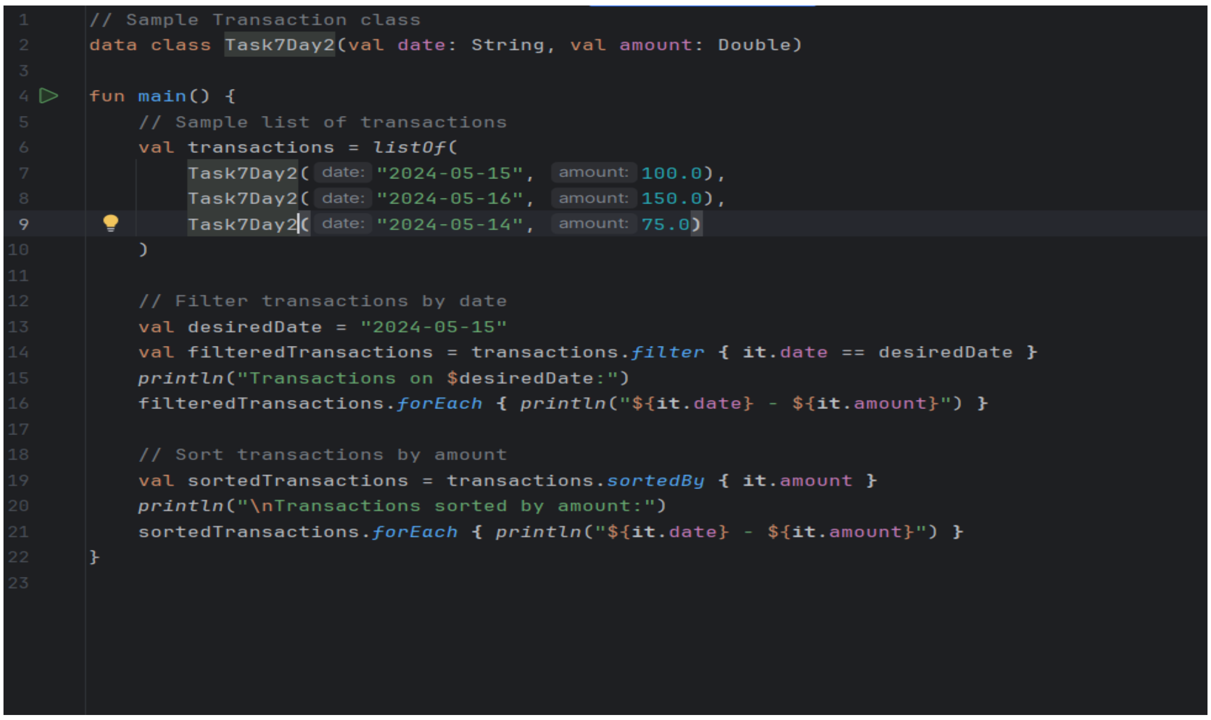


```
class User(private val username: String, private val password: String) {
    private var loggedIn = false

    fun login(inputUsername: String, inputPassword: String) {
        if (inputUsername == username && inputPassword == password) {
            loggedIn = true
            println("Login successful!")
        } else {
            println("Invalid username or password.")
        }
    }

    fun displayExpenseSummary() {
        if (loggedIn) {
            // Code to display expense summary
            println("Displaying expense summary...")
        } else {
            println("Please log in first.")
        }
    }
}
```

Task 7: Use lambdas and higher-order functions to filter and sort transactions by date or amount.



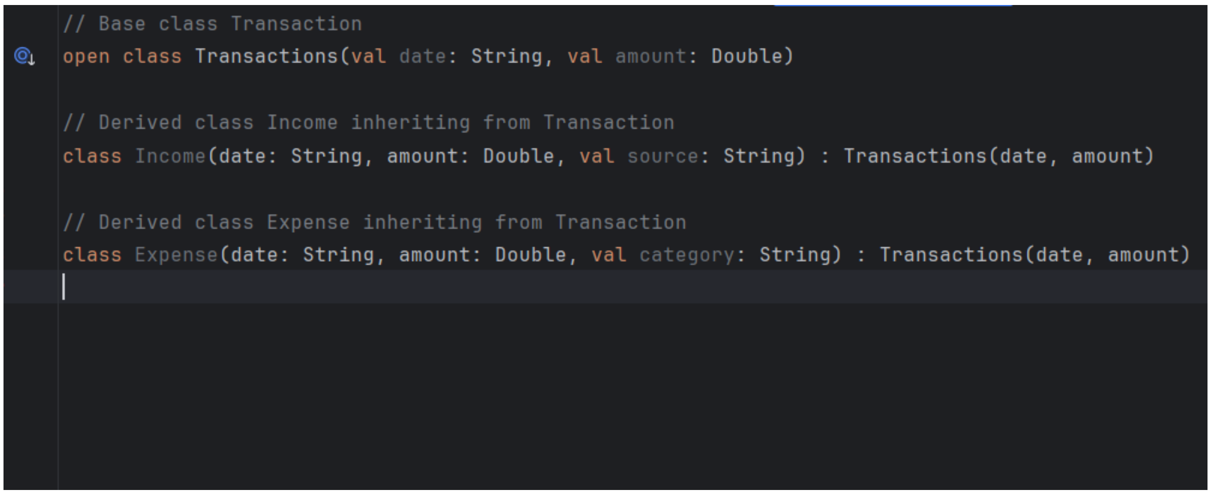
```kotlin
// Sample Transaction class
data class Task7Day2(val date: String, val amount: Double)

fun main() {
    // Sample list of transactions
    val transactions = listOf(
        Task7Day2( date: "2024-05-15",  amount: 100.0),
        Task7Day2( date: "2024-05-16",  amount: 150.0),
        Task7Day2( date: "2024-05-14",  amount: 75.0)
    )

    // Filter transactions by date
    val desiredDate = "2024-05-15"
    val filteredTransactions = transactions.filter { it.date == desiredDate }
    println("Transactions on $desiredDate:")
    filteredTransactions.forEach { println("${it.date} - ${it.amount}") }

    // Sort transactions by amount
    val sortedTransactions = transactions.sortedBy { it.amount }
    println("\nTransactions sorted by amount:")
    sortedTransactions.forEach { println("${it.date} - ${it.amount}") }
}
```

Task 8: Implement inheritance by creating specific transaction classes like Income and Expense that inherit from Transaction.



```kotlin
// Base class Transaction
open class Transactions(val date: String, val amount: Double)

// Derived class Income inheriting from Transaction
class Income(date: String, amount: Double, val source: String) : Transactions(date, amount)

// Derived class Expense inheriting from Transaction
class Expense(date: String, amount: Double, val category: String) : Transactions(date, amount)
```