Assignment 1, Mobile Development

Madi Khassenov, 23MD0450

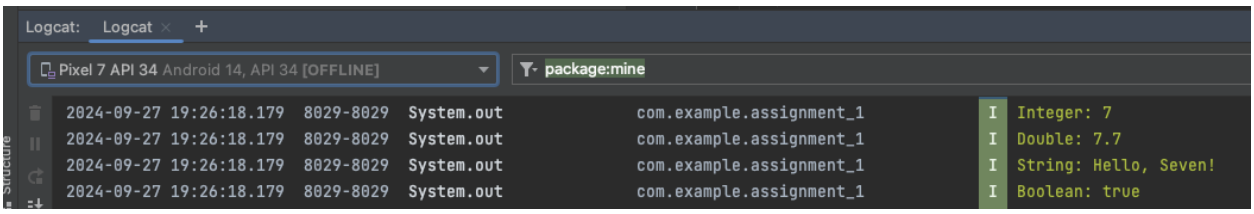
Exercise 1: Kotlin Syntax Basics

1. Variables and Data Types:

```kotlin
fun main() {
    val myInt: Int = 7
    val myDouble: Double = 7.7
    val myString: String = "Hello, Seven!"
    val myBoolean: Boolean = true

    println("Integer: $myInt")
    println("Double: $myDouble")
    println("String: $myString")
    println("Boolean: $myBoolean")
}
```

```kotlin
// Ex1
val ex1 = Ex1()
ex1.main()
```



2. Conditional Statements:

```
fun checkNumber(number: Int) {
    if (number > 0) {
        println("$number is positive")
    } else if (number < 0) {
        println("$number is negative")
    } else {
        println("$number is zero")
    }
}
```

```
ex1.checkNumber( number: 7)
```

```
Boolean: true
7 is positive
```

3. Loops:

```
fun loop() {
    println("Using for loop:")
    for (i in 1 ≤ .. ≤ 10) {
        println(i)
    }

    println("Using while loop:")
    var i = 1
    while (i <= 10) {
        println(i)
        i++
    }
}
```

```
ex1.loop()
```

```
com.example.assignment_1 I Using for loop:
com.example.assignment_1 I 1
com.example.assignment_1 I 2
com.example.assignment_1 I 3
com.example.assignment_1 I 4
com.example.assignment_1 I 5
com.example.assignment_1 I 6
com.example.assignment_1 I 7
com.example.assignment_1 I 8
com.example.assignment_1 I 9
com.example.assignment_1 I 10
```

```
com.example.assignment_1 I Using while loop:
com.example.assignment_1 I 1
com.example.assignment_1 I 2
com.example.assignment_1 I 3
com.example.assignment_1 I 4
com.example.assignment_1 I 5
com.example.assignment_1 I 6
com.example.assignment_1 I 7
com.example.assignment_1 I 8
com.example.assignment_1 I 9
com.example.assignment_1 I 10
```

4. **Collections:**

```kotlin
fun collections() {
    val numbers = listOf(1, 2, 3, 4, 5)
    var sum = 0

    for (number in numbers) {
        sum += number
    }

    println("Sum of all numbers: $sum")
}
```

```kotlin
ex1.collections()
```

```
Sum of all numbers: 15
```

Exercise 2: Kotlin OOP (Object-Oriented Programming)

1. Create a Person class:

```kotlin
open class Person(
    private val name: String,
    private val age: Int,
    private val email: String
) {
    open fun details() {
        println("Name: $name")
        println("Age: $age")
        println("Email: $email")
    }
}
```

```
val person = Person( name: "Alibek",  age: 30,  email: "alibek@gmail.com")
person.details()
```

```
Name: Alibek
Age: 30
Email: alibek@gmail.com
```

2. Inheritance:

```
class Employee(
    name: String,
    age: Int,
    email: String,
    private val salary: Double
) : Person(name, age, email) {
    override fun details() {
        super.details()
        println("Salary: $salary")
    }
}
```

```
val employee = Employee( name: "Asel",  age: 25,  email: "asel@gmail.com",  salary: 500000.0)
employee.details()
```

```
Name: Asel
Age: 25
Email: asel@gmail.com
Salary: 500000.0
```

3. Encapsulation:

```
class BankAccount(private var balance: Double) {
    fun deposit(amount: Double) {
        if (amount > 0) {
            balance += amount
            println("Deposited: $amount, New Balance: $balance")
        } else {
            println("Deposit amount must be positive.")
        }
    }

    fun withdraw(amount: Double) {
        if (amount > 0 && amount <= balance) {
            balance -= amount
            println("Withdrew: $amount, New Balance: $balance")
        } else {
            println("Insufficient balance or invalid amount.")
        }
    }

    fun displayBalance() {
        println("Current Balance: $balance")
    }
}
```

```
val bankAccount = BankAccount( balance: 1000.0)
bankAccount.deposit( amount: 500.0)
bankAccount.withdraw( amount: 200.0)
bankAccount.displayBalance()
```

```
Deposited: 500.0, New Balance: 1500.0
Withdrew: 200.0, New Balance: 1300.0
Current Balance: 1300.0
```

Exercise 3: Kotlin Functions

1. Basic Function:

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

```
val ex3 = Ex3()
println("Sum of 5 and 3: ${ex3.sum( a: 5,  b: 3)}")
```

```
Sum of 5 and 3: 8
```

2. Lambda Functions:

```
val multiply: (Int, Int) -> Int = { a, b -> a * b }
```

```
println("Product of 5 and 3: ${ex3.multiply(5, 3)}")
```

```
I  Product of 5 and 3: 15
```

3. Higher-Order Functions:

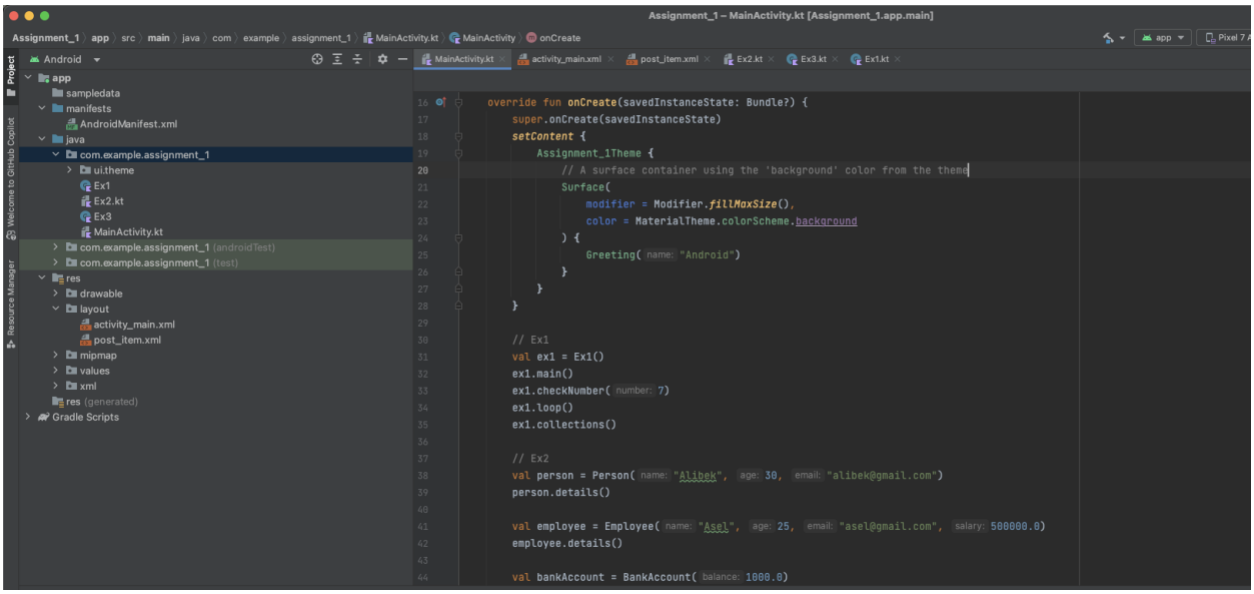
```
fun higherOrder(a: Int, b: Int, operation: (Int, Int) -> Int): Int {
    return operation(a, b)
}
```

```
println("Product of 5 and 3 using higher order function: ${ex3.higherOrder( a: 5,  b: 3, ex3.multiply)}")
```

```
I  Product of 5 and 3 using higher order function: 15
```

Exercise 4: Android Layout in Kotlin (Instagram-like Layout)

1. Set Up the Android Project:



2. Design the Layout:

MainActivity.kt | activity_main.xml | post_item.xml | Ex2.kt | Ex3.kt | Ex1.kt

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerViewFeed"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:padding="8dp"
        app:layoutManager="androidx.recyclerview.widget.GridLayoutManager"
        app:spanCount="3"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />
</LinearLayout>
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    card_view:cardCornerRadius="8dp"
    android:layout_margin="4dp"
    card_view:cardElevation="4dp">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <ImageView
            android:id="@+id/imageViewPost"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="1"
            android:scaleType="centerCrop"
            android:contentDescription="@string/post_image" />

        <TextView
            android:id="@+id/textViewCaption"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:padding="8dp"
            android:text="Caption"
            android:textSize="14sp" />

    </LinearLayout>

</LinearLayout>
```