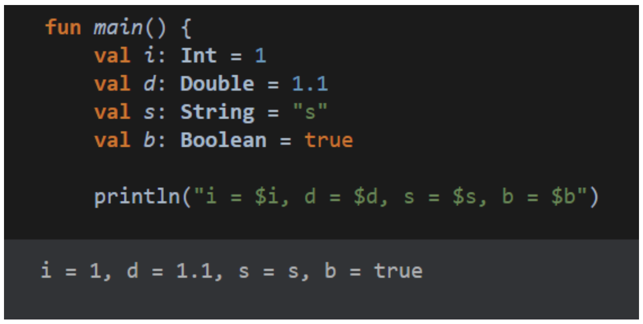
Assignment 1, Mobile Programming
Atentayev Abdulla

Exercise 1: Kotlin Syntax Basics

1. **Variables and Data Types:**
   - Create variables of different data types: Int, Double, String, Boolean.
   - Print the variables using println.



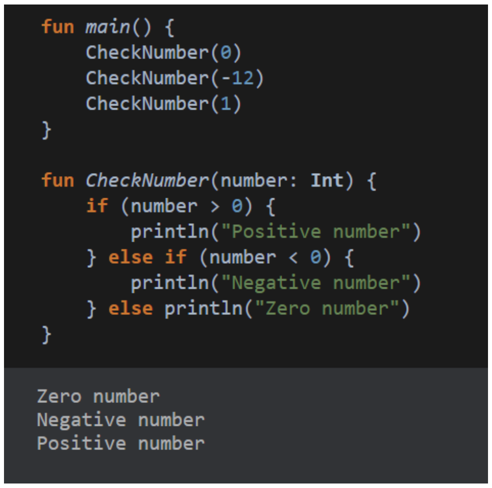
```kotlin
fun main() {
    val i: Int = 1
    val d: Double = 1.1
    val s: String = "s"
    val b: Boolean = true

    println("i = $i, d = $d, s = $s, b = $b")
```

```
i = 1, d = 1.1, s = s, b = true
```

Screenshot #1 – Print data types in console

Conditional Statements:

- Create a simple program that checks if a number is positive, negative, or zero.



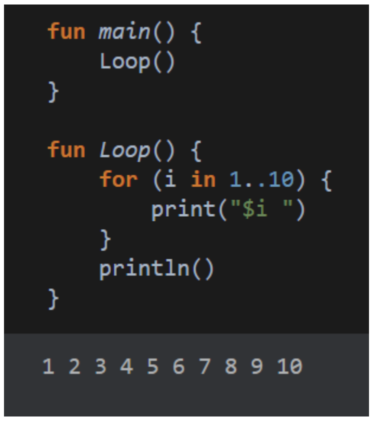
```kotlin
fun main() {
    CheckNumber(0)
    CheckNumber(-12)
    CheckNumber(1)
}

fun CheckNumber(number: Int) {
    if (number > 0) {
        println("Positive number")
    } else if (number < 0) {
        println("Negative number")
    } else println("Zero number")
}
```

```
Zero number
Negative number
Positive number
```

Screenshot #2 – Function “Print type of number”

Loops:

- Write a program that prints numbers from 1 to 10 using for and while loops



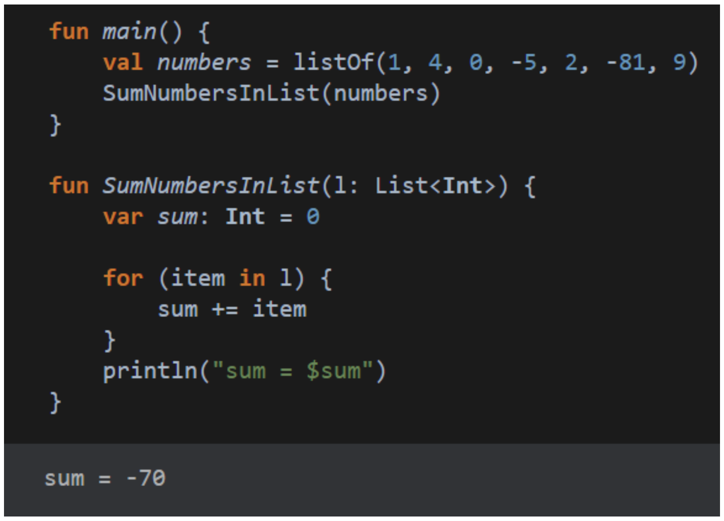
```kotlin
fun main() {
    Loop()
}

fun Loop() {
    for (i in 1..10) {
        print("$i ")
    }
    println()
}
```

```
1 2 3 4 5 6 7 8 9 10
```

Screenshot #3 – Function “Print numbers from 1 to 10”

Collections:

- Create a list of numbers, iterate through the list, and print the sum of all numbers.



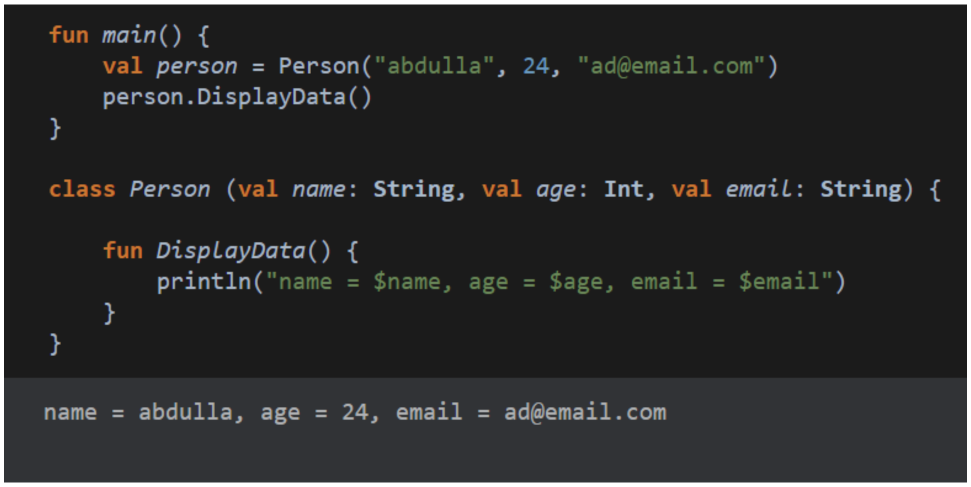
```kotlin
fun main() {
    val numbers = listOf(1, 4, 0, -5, 2, -81, 9)
    SumNumbersInList(numbers)
}

fun SumNumbersInList(l: List<Int>) {
    var sum: Int = 0

    for (item in l) {
        sum += item
    }
    println("sum = $sum")
}
```

```
sum = -70
```

Screenshot #4 – Function “Print sum of numbers in list”

Exercise 2: Kotlin OOP (Object-Oriented Programming)

1. **Create a Person class:**
   - Define properties for name, age, and email.
   - Create a method to display the person's details.



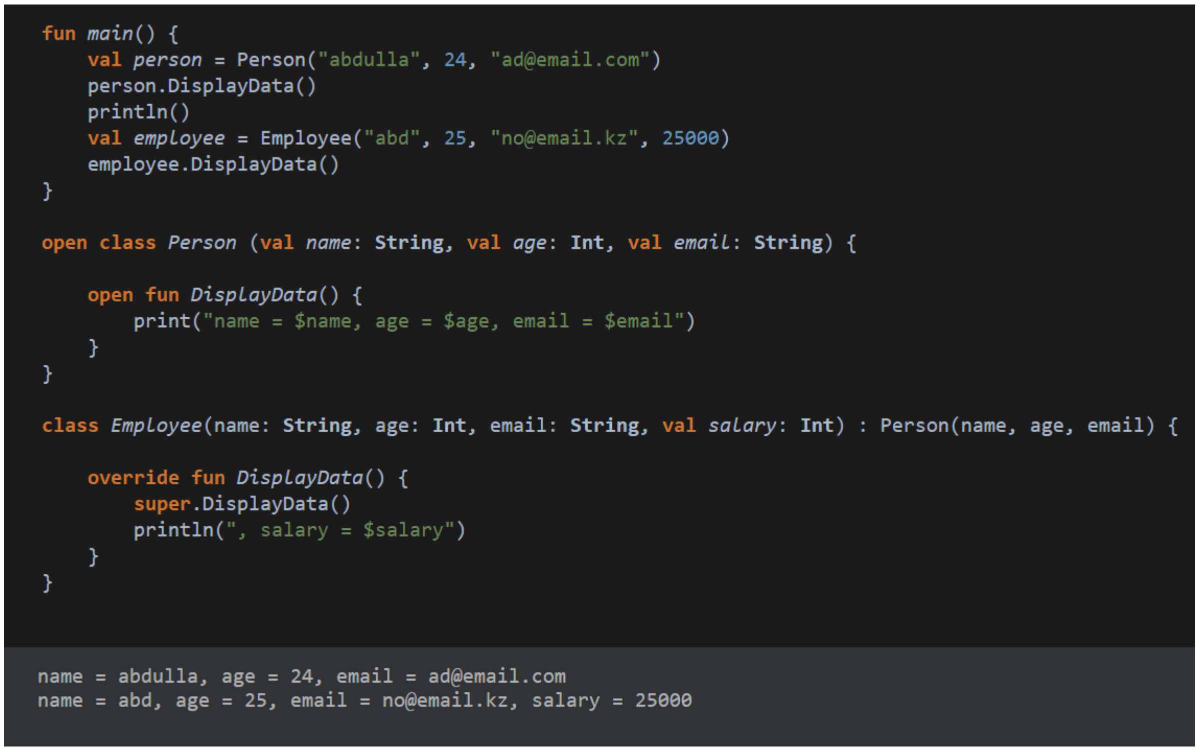
```kotlin
fun main() {
    val person = Person("abdulla", 24, "ad@email.com")
    person.DisplayData()
}

class Person (val name: String, val age: Int, val email: String) {

    fun DisplayData() {
        println("name = $name, age = $age, email = $email")
    }
}
```

```
name = abdulla, age = 24, email = ad@email.com
```

Screenshot #5 – Class Person

Inheritance:

- Create a class Employee that inherits from the Person class.
- Add a property for salary.
- Override the displayInfo method to include the salary.



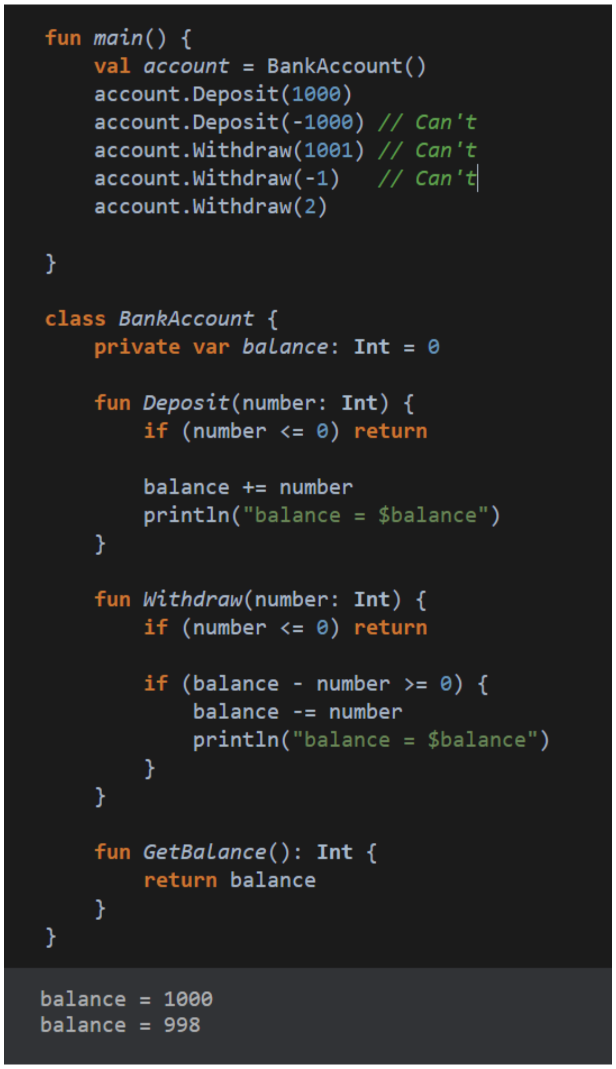
```
fun main() {
    val person = Person("abdulla", 24, "ad@email.com")
    person.DisplayData()
    println()
    val employee = Employee("abd", 25, "no@email.kz", 25000)
    employee.DisplayData()
}

open class Person (val name: String, val age: Int, val email: String) {

    open fun DisplayData() {
        print("name = $name, age = $age, email = $email")
    }
}

class Employee(name: String, age: Int, email: String, val salary: Int) : Person(name, age, email) {

    override fun DisplayData() {
        super.DisplayData()
        println(", salary = $salary")
    }
}
```

```
name = abdulla, age = 24, email = ad@email.com
name = abd, age = 25, email = no@email.kz, salary = 25000
```

Screenshot #6 – Class Employee

Encapsulation:

- Create a BankAccount class with a private property balance.
- Provide methods to deposit and withdraw money, ensuring the balance never goes negative.



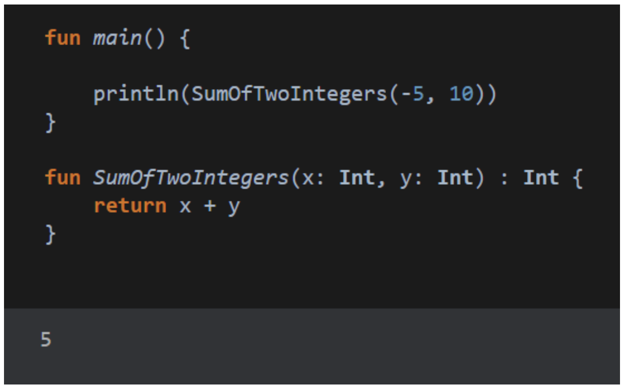
```kotlin
fun main() {
    val account = BankAccount()
    account.Deposit(1000)
    account.Deposit(-1000) // Can't
    account.Withdraw(1001) // Can't
    account.Withdraw(-1)   // Can't
    account.Withdraw(2)

}

class BankAccount {
    private var balance: Int = 0

    fun Deposit(number: Int) {
        if (number <= 0) return

        balance += number
        println("balance = $balance")
    }

    fun Withdraw(number: Int) {
        if (number <= 0) return

        if (balance - number >= 0) {
            balance -= number
            println("balance = $balance")
        }
    }

    fun GetBalance(): Int {
        return balance
    }
}
```

```
balance = 1000
balance = 998
```

Screenshot #7 - Class BankAccount

Exercise 3: Kotlin Functions

1. **Basic Function:**
   - Write a function that takes two integers as arguments and returns their sum



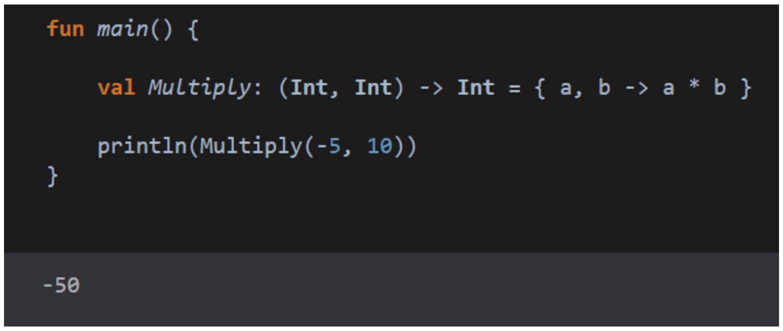
```kotlin
fun main() {

    println(SumOfTwoIntegers(-5, 10))
}

fun SumOfTwoIntegers(x: Int, y: Int) : Int {
    return x + y
}
```

```
5
```

Screenshot #8 – Function sum of two integers

Lambda Functions:

- Create a lambda function that multiplies two numbers and returns the result



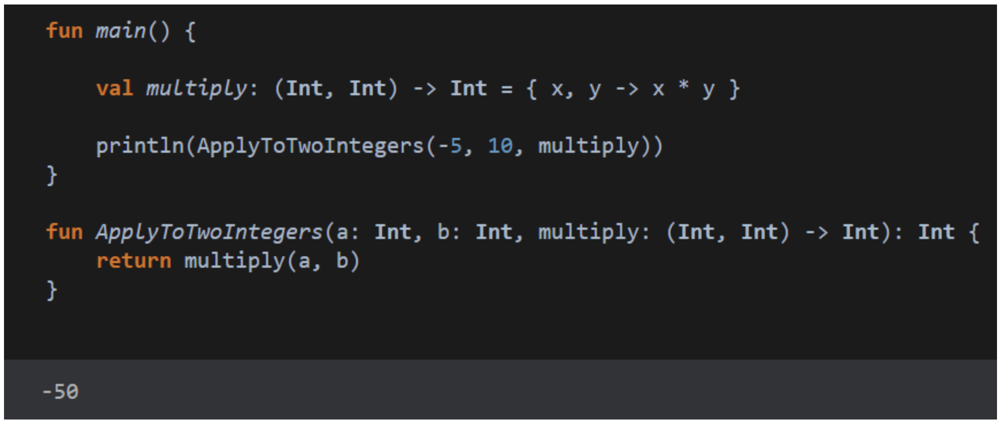
```kotlin
fun main() {

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

    println(Multiply(-5, 10))
}
```

```
-50
```

Screenshot #9 – Lambda multiply function

Higher-Order Functions:

- Write a function that takes a lambda function as a parameter and applies it to two integers.



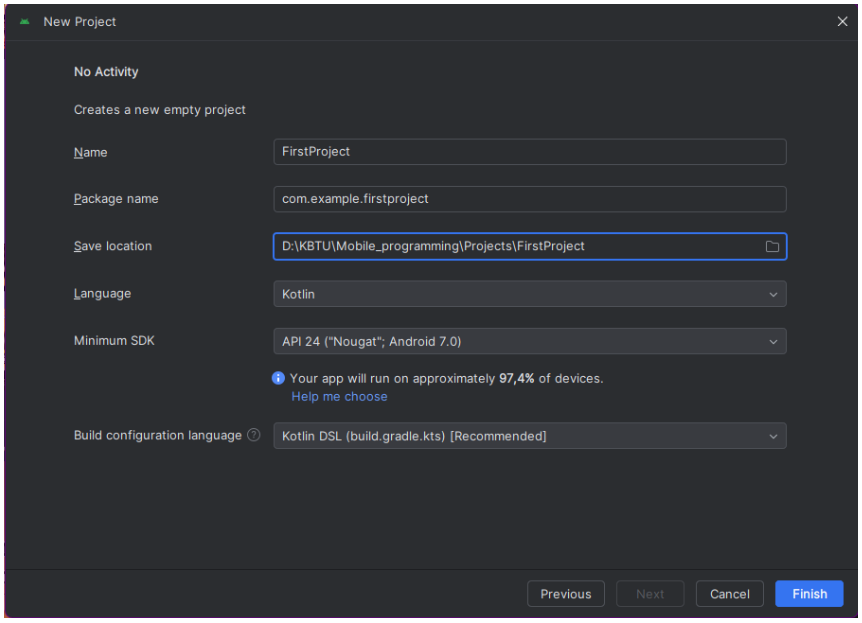
```
fun main() {

    val multiply: (Int, Int) -> Int = { x, y -> x * y }

    println(ApplyToTwoIntegers(-5, 10, multiply))
}

fun ApplyToTwoIntegers(a: Int, b: Int, multiply: (Int, Int) -> Int): Int {
    return multiply(a, b)
}
```

```
-50
```

Screenshot #10 – Lambda function in function

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

1. **Set Up the Android Project:**
   - Create a new Android project in Android Studio.
   - Ensure you have a Kotlin-based project.



New Project

No Activity

Creates a new empty project

Name: FirstProject

Package name: com.example.firstproject

Save location: D:\KBTU\Mobile_programming\Projects\FirstProject

Language: Kotlin

Minimum SDK: API 24 ("Nougat"; Android 7.0)

Your app will run on approximately **97,4%** of devices.
Help me choose

Build configuration language: Kotlin DSL (build.gradle.kts) [Recommended]

Screenshot #11 – Create new project

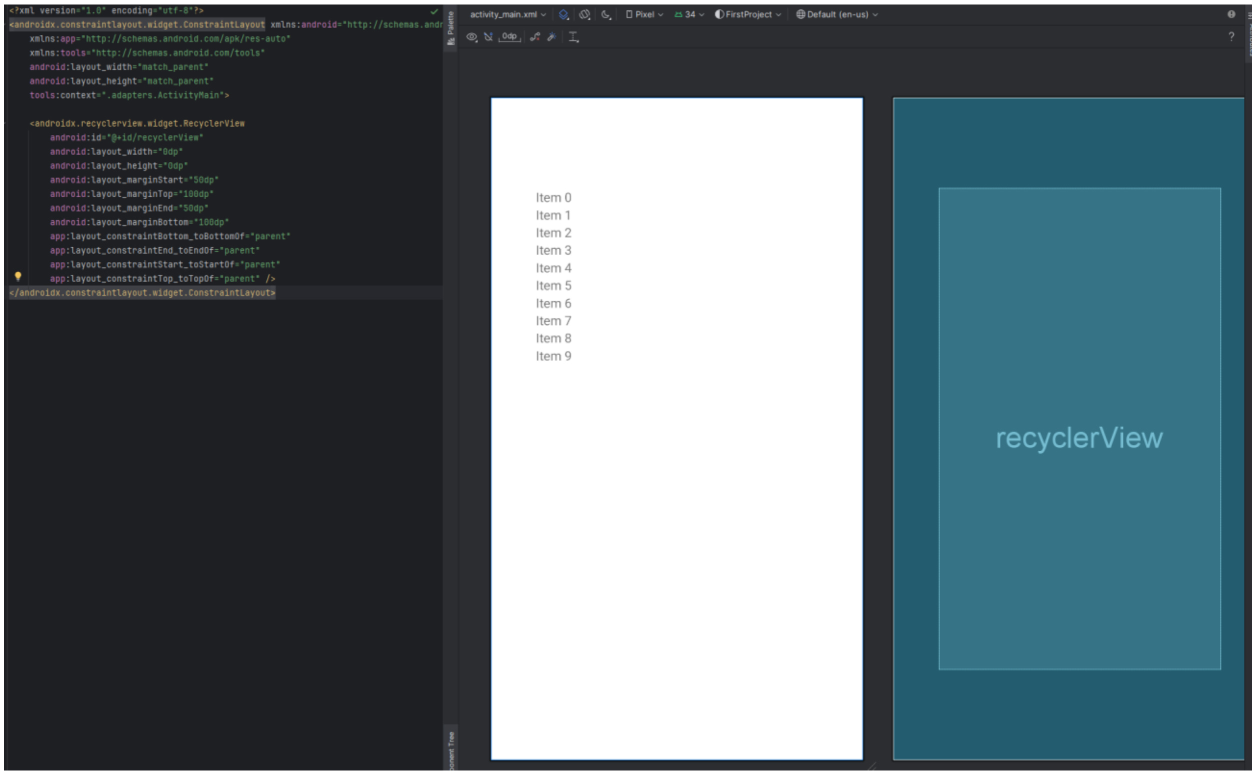
2. **Design the Layout:**
   - Create a new XML layout file (activity_main.xml) for a simple Instagram-like user interface.
   - Include elements like ImageView, TextView, and RecyclerView for the feed

   Create the RecyclerView Adapter:

   - Set up the RecyclerView to display a feed of posts with ImageView for the picture and TextView for the caption.

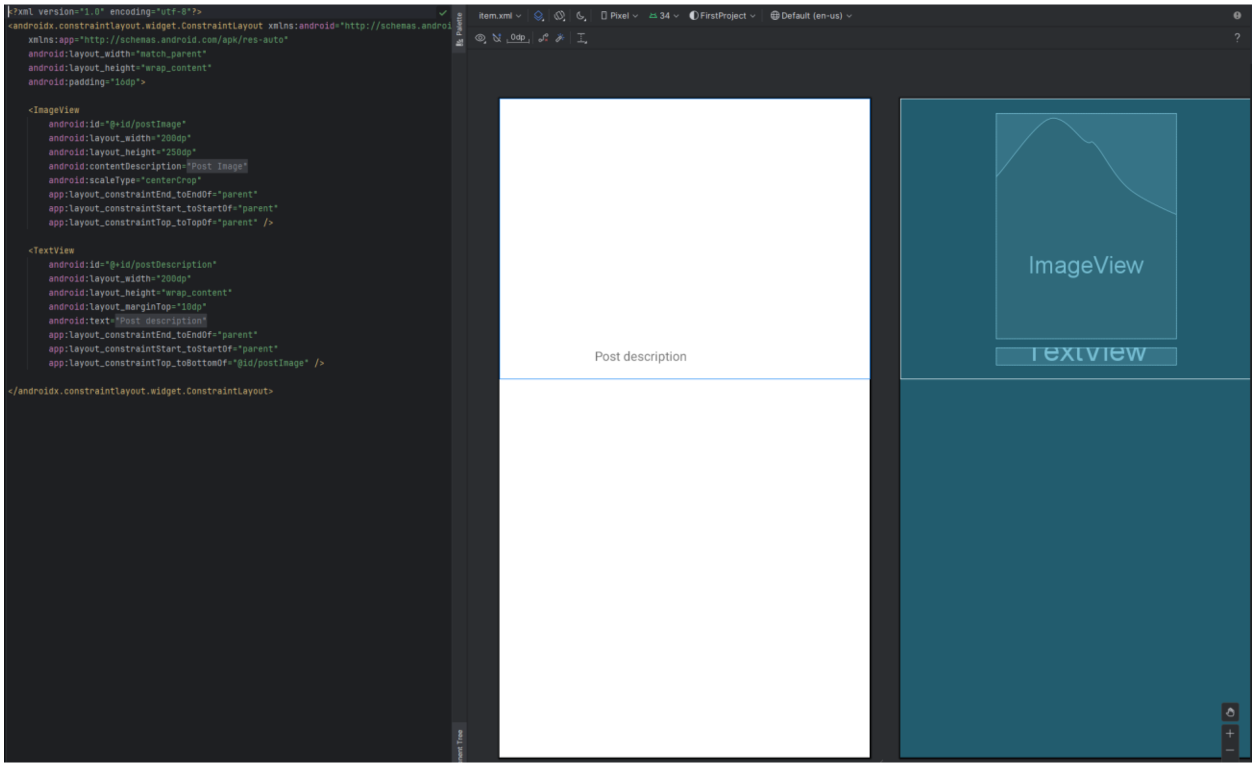
   MainActivity Setup:

   - Initialize the RecyclerView in MainActivity and populate it with sample data



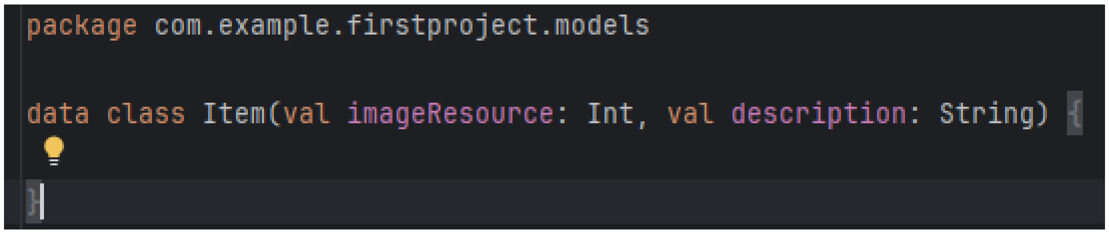
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.andr
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".adapters.ActivityMain">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginStart="50dp"
        android:layout_marginTop="100dp"
        android:layout_marginEnd="50dp"
        android:layout_marginBottom="100dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Screenshot #12 – Make main activity page



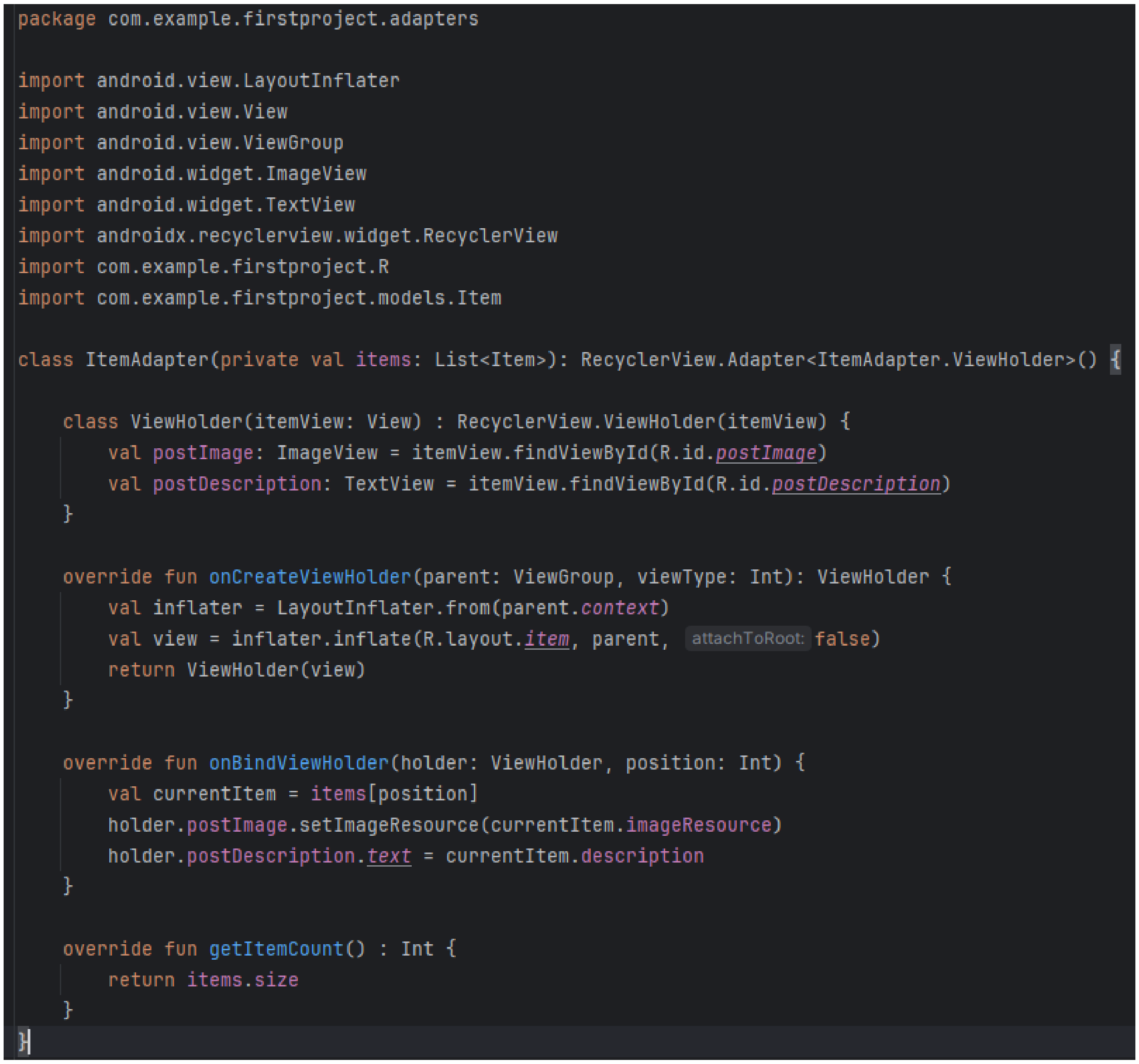
```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.androi
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="16dp">

    <ImageView
        android:id="@+id/postImage"
        android:layout_width="200dp"
        android:layout_height="250dp"
        android:contentDescription="Post Image"
        android:scaleType="centerCrop"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/postDescription"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:text="Post description"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/postImage" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Screenshot #13 – Make post item



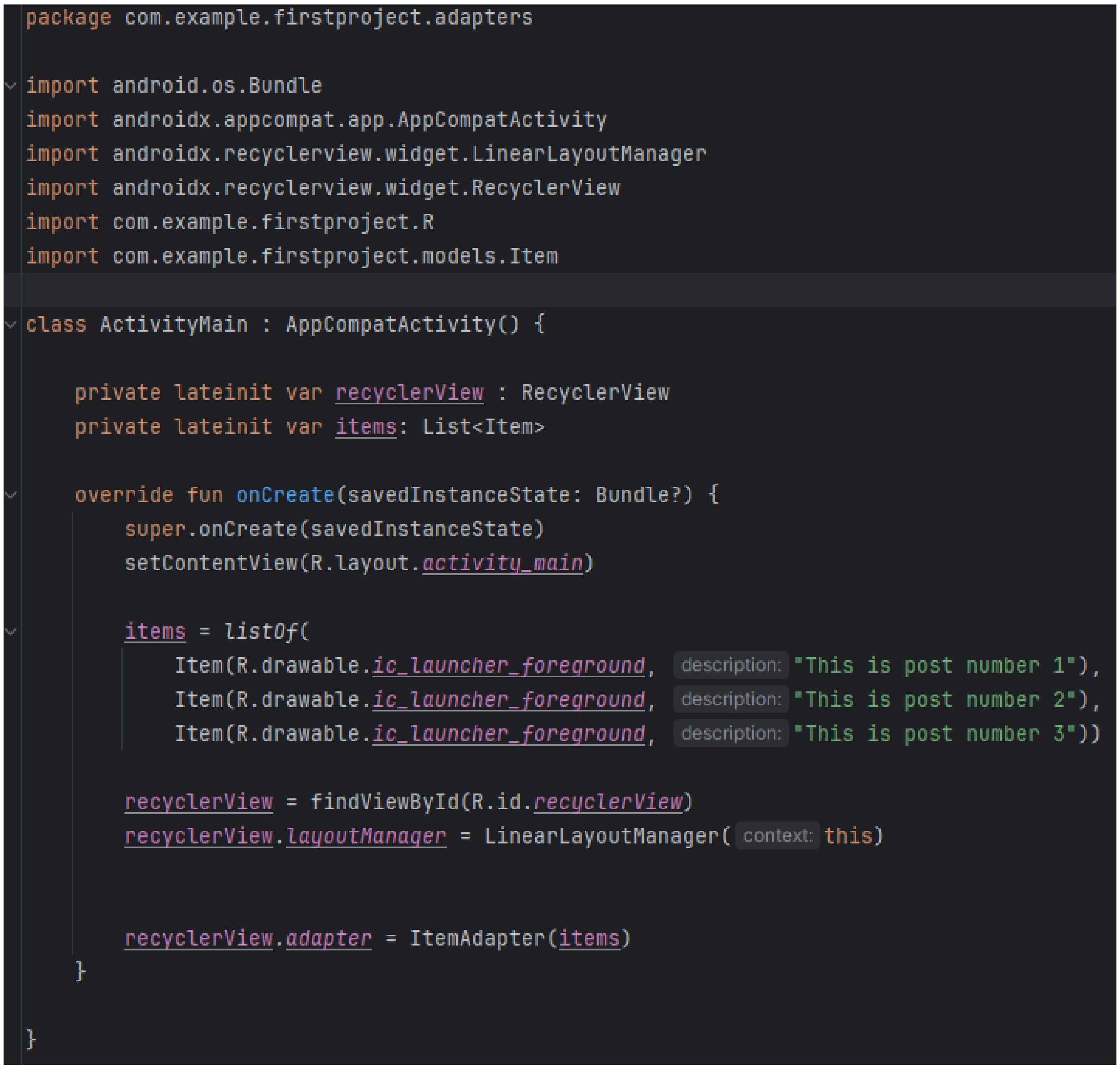
```kotlin
package com.example.firstproject.models

data class Item(val imageResource: Int, val description: String) {

}
```

Screenshot #14 – Make item model



```kotlin
package com.example.firstproject.adapters

import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ImageView
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView
import com.example.firstproject.R
import com.example.firstproject.models.Item

class ItemAdapter(private val items: List<Item>): RecyclerView.Adapter<ItemAdapter.ViewHolder>() {

    class ViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
        val postImage: ImageView = itemView.findViewById(R.id.postImage)
        val postDescription: TextView = itemView.findViewById(R.id.postDescription)
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
        val inflater = LayoutInflater.from(parent.context)
        val view = inflater.inflate(R.layout.item, parent,  attachToRoot: false)
        return ViewHolder(view)
    }

    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
        val currentItem = items[position]
        holder.postImage.setImageResource(currentItem.imageResource)
        holder.postDescription.text = currentItem.description
    }

    override fun getItemCount() : Int {
        return items.size
    }
}
```

Screenshot #15 – Make item adapter



```kotlin
package com.example.firstproject.adapters

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import com.example.firstproject.R
import com.example.firstproject.models.Item

class ActivityMain : AppCompatActivity() {

    private lateinit var recyclerView : RecyclerView
    private lateinit var items: List<Item>

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        items = listOf(
            Item(R.drawable.ic_launcher_foreground,  description: "This is post number 1"),
            Item(R.drawable.ic_launcher_foreground,  description: "This is post number 2"),
            Item(R.drawable.ic_launcher_foreground,  description: "This is post number 3"))

        recyclerView = findViewById(R.id.recyclerView)
        recyclerView.layoutManager = LinearLayoutManager( context: this)


        recyclerView.adapter = ItemAdapter(items)
    }

}
```

Screenshot #16 – Make main activity adapter