

Assignment 1, Mobile Programming

Exercise 1: Kotlin Syntax Basics

1. Variables and Data Types:

```
1 fun main(args: Array<String>) {
2     var salary: Int = 600000
3     val weight: Double = 55.9
4     val city: String = "Astana"
5     val isSunny: Boolean = true
6
7     println("Salary: $salary")
8     println("Weight: $weight")
9     println("City: $city")
10    println("Is sunny: $isSunny")
11 }
```

Output:

Salary: 600000
Weight: 55.9
City: Astana
Is sunny: true

- Create variables of different data types: Int, Double, String, Boolean.
- Print the variables using println.

Conditional Statements:

```
1 fun main(args: Array<String>) {
2     print("Enter a number: ")
3     val number = readLine()?.toIntOrNull()
4
5     if (number > 0) {
6         println("Positive")
7     } else if (number < 0) {
8         println("Negative")
9     } else {
10        println("Zero")
11    }
12 }
```

STDIN

4

Output:

Enter a number: Positive

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

Loops:

```
1 fun main(args: Array<String>) {
2     println("For loop:")
3     for (i in 1..10) {
4         println(i)
5     }
6
7     println("While loop:")
8     var i = 1
9     while (i <= 10) {
10        println(i)
11        i++
12    }
13 }
```

Output:

For loop:

1
2
3
4
5
6
7
8
9
10

While loop:

1
2
3
4
5
6
7
8
9
10

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

Collections:

```
1 fun main(args: Array<String>) {
2     val numbers = listOf(100, 85, 91, 100)
3     var sum = 0
4
5     for (num in numbers) {
6         sum += num
7     }
8     println("Sum = $sum")
9 }
10
```

Output:
Sum = 376

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

Exercise 2: Kotlin OOP (Object-Oriented Programming)

```
1 //Create a Person class:
2 open class Person(val name: String, val age: Int, val email: String) {
3     open fun displayInfo() {
4         println("Name: $name")
5         println("Age: $age")
6         println("Email: $email")
7     }
8 }
9 //Inheritance:
10 class Employee(name: String, age: Int, email: String, val salary: Double) : Person(name, age, email) {
11     override fun displayInfo() {
12         super.displayInfo()
13         println("Salary: $$salary")
14     }
15 }
16 //Encapsulation:
17 class BankAccount(private var balance: Double) {
18     fun deposit(amount: Double) {
19         if (amount > 0) {
20             balance += amount
21             println("Inserted: $$amount. Current balance: $$balance")
22         } else {
23             println("Error!")
24         }
25     }
26     fun withdraw(amount: Double) {
27         if (amount > 0 && amount <= balance) {
28             balance -= amount
29             println("Withdrawn: $$amount. Current balance: $$balance")
30         } else {
31             println("Error!")
32         }
33     }
34 }
35
36 fun main() {
37     val person = Person("Biba", 23, "b_saugabayeva@kbtu.kz")
38     person.displayInfo()
39     val employee = Employee("Ulzhana", 24, "u_kylyshbek@kbtu.kz", 50000.0)
40     employee.displayInfo()
41     val bankAccount = BankAccount(1000.0)
42     bankAccount.deposit(200.0)
43     bankAccount.withdraw(500.0)
44     bankAccount.withdraw(800.0)
45 }
```

STDIN

Output:

```
Name: Biba
Age: 23
Email: b_saugabayeva@kbtu.kz
Name: Ulzhana
Age: 24
Email: u_kylyshbek@kbtu.kz
Salary: $50000.0
Inserted: $200.0. Current balance: $1200.0
Withdrawn: $500.0. Current balance: $700.0
Error!
```

1. Create a Person class:

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

Inheritance:

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

Encapsulation:

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

Exercise 3: Kotlin Functions

1. Basic Function:

<pre>1 // Basic function: 2 fun sum(a: Int, b: Int): Int { 3 return a + b 4 } 5 6 fun main() { 7 val result = sum(100, 94) 8 println("\$result") 9 } 10</pre>	STDIN
	Input for the p
	Output:
	194

- Write a function that takes two integers as arguments and returns their sum

Lambda Functions:

<pre>1 // Basic function: 2 fun sum(a: Int, b: Int): Int { 3 return a + b 4 } 5 6 val multiply: (Int, Int) -> Int = { a, b -> a * b } 7 8 fun main() { 9 val result = sum(100, 94) 10 println("\$result") 11 12 val result2 = multiply(4, 12) 13 println("\$result2") 14 } 15</pre>	STDIN
	Input for
	Output:
	194
	48

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

Higher-Order Functions:

<pre>1 // Basic function: 2 fun sum(a: Int, b: Int): Int { 3 return a + b 4 } 5 //Lambda Functions: 6 val multiply: (Int, Int) -> Int = { a, b -> a * b } 7 //Higher-Order Functions: 8 fun applyOperation(a: Int, b: Int, operation: (Int, Int) -> Int): Int { 9 return operation(a, b) 10 } 11 12 fun main() { 13 val result = sum(100, 94) 14 println("\$result") 15 16 val result2 = multiply(4, 12) 17 println("\$result2") 18 19 //Using Higher-Order Functions: 20 val result_2 = applyOperation(100, 94) { x, y -> x + y } 21 println("\$result_2") 22 23 val result2_2 = applyOperation(4, 12, multiply) 24 println("\$result2_2") 25 }</pre>	STDIN
	Input for ti
	Output:
	194
	48
	194
	48

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

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

In this part of the assignment, I faced a lot of challenges. Initially, I encountered issues with the versions of the JDK, Gradle, and SDK. Despite trying to update and downgrade them, nothing seemed to work.

This might be because I had downloaded Android Studio almost a year ago. Eventually, I decided to uninstall and reinstall Android Studio completely.

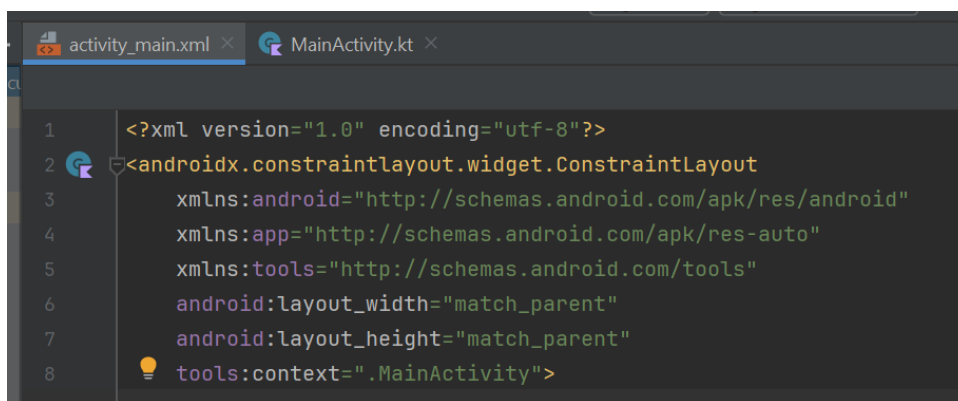
After reinstalling, I ran into new issues with the MainActivity and the virtual Android simulator. It turned out that I had opened the project incorrectly at the start. Then the simulator kept terminating, reporting a lack of storage space. Even though there had been 12GB of free space out of 205GB in the morning, the storage quickly dropped to 0MB. I had to delete a lot of files and restart my computer to free up space.

Despite all these difficulties, it was a great learning experience, though it took up more time than I expected, which is why I wasn't able to submit it on time (

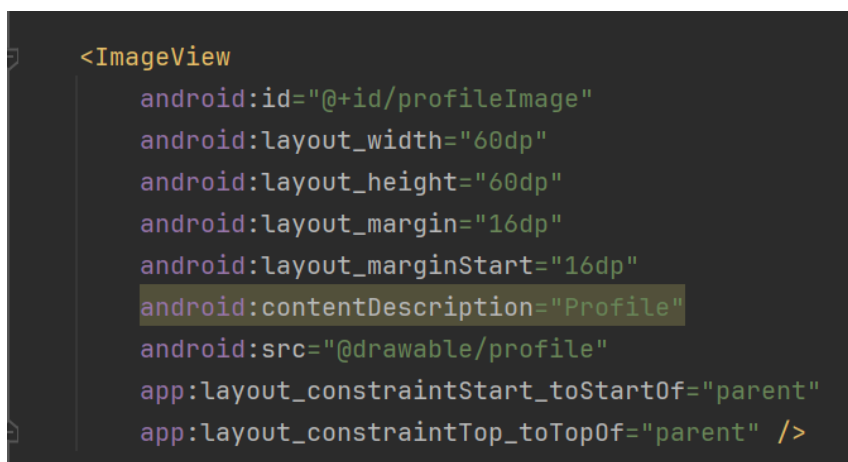
1. Set Up the Android Project:

- Create a new Android project in Android Studio.
- Ensure you have a Kotlin-based project.

This is the activity_main.xml file.



ImageView is used to show the picture in the feed in an app. Some specifications like width, height, margin are shown.



TextView is used to display text under each image in the feed.

```

<TextView
    android:id="@+id/username"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="28dp"
    android:text="ulzhana_kub"
    android:textSize="17sp"
    android:textStyle="bold"
    app:layout_constraintStart_toEndOf="@id/profileImage"
    app:layout_constraintTop_toTopOf="parent" />

```

RecyclerView is a more advanced version of ListView. It is used to show a list of posts in a scrollable layout, like Instagram feed.

```

<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerView"
    android:layout_width="0dp"
    android:layout_height="0dp"
    app:layout_constraintTop_toBottomOf="@id/profileImage"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintBottom_toBottomOf="parent" />

<androidx.constraintlayout.widget.ConstraintLayout>

```

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:

```

// MainActivity.kt
data class Post(val imageResId: Int, val caption: String)

class PostAdapter(private val posts: List<Post>) :
    RecyclerView.Adapter<PostAdapter.PostViewHolder>() {

    // ViewHolder class
    class PostViewHolder(private val binding: PostIndBinding) : RecyclerView.ViewHolder(binding.root) {
        fun bind(post: Post) {
            binding.postImage.setImageResource(post.imageResId)
            binding.postCaption.text = post.caption
        }
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): PostViewHolder {
        val inflater = LayoutInflater.from(parent.context)
        val binding = PostIndBinding.inflate(inflater, parent, attachToParent: false)
        return PostViewHolder(binding)
    }

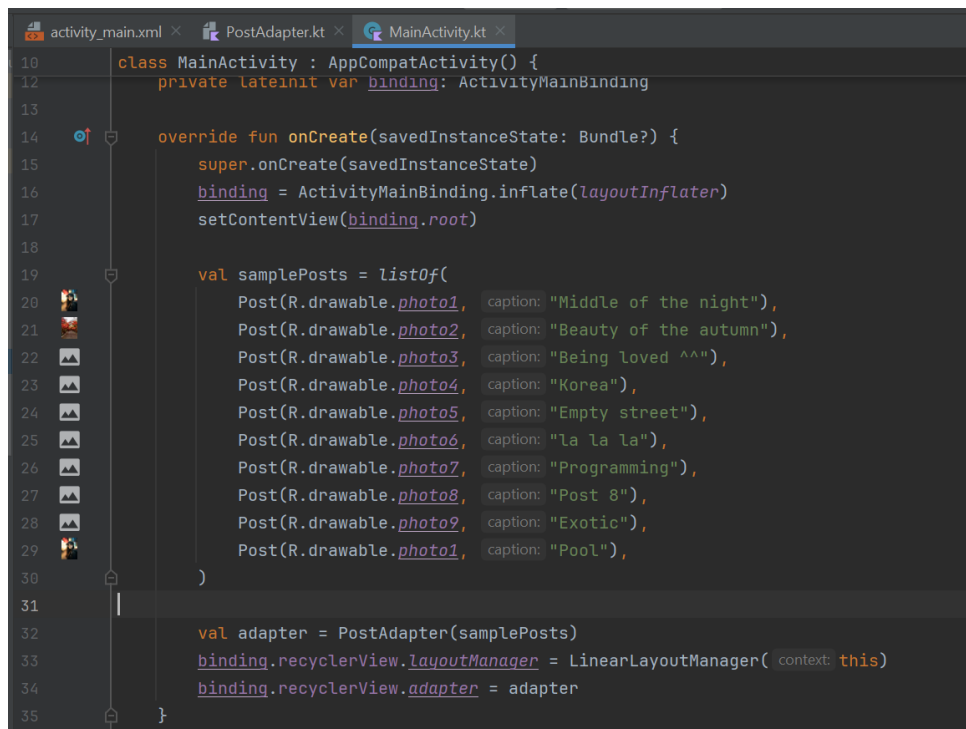
    override fun onBindViewHolder(holder: PostViewHolder, position: Int) {
        holder.bind(posts[position])
    }

    override fun getItemCount(): Int = posts.size
}

```

This is the code of the RecyclerView Adapter. It manages the data and provides a binding between data and the views displayed in the RecyclerView.

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

A screenshot of an IDE showing the MainActivity.kt file. The code defines a MainActivity class that inherits from AppCompatActivity. In the onCreate method, it inflates the ActivityMainBinding, sets the content view, and creates a list of sample posts. Each post consists of a drawable resource (photo1 through photo9, and photo1 again) and a text caption. A PostAdapter is created with the sample posts, and the RecyclerView's layoutManager is set to LinearLayoutManager(this). The adapter is then set on the RecyclerView.

```
10 class MainActivity : AppCompatActivity() {
11     private lateinit var binding: ActivityMainBinding
12
13
14     override fun onCreate(savedInstanceState: Bundle?) {
15         super.onCreate(savedInstanceState)
16         binding = ActivityMainBinding.inflate(layoutInflater)
17         setContentView(binding.root)
18
19         val samplePosts = listOf(
20             Post(R.drawable.photo1, caption: "Middle of the night"),
21             Post(R.drawable.photo2, caption: "Beauty of the autumn"),
22             Post(R.drawable.photo3, caption: "Being loved ^^"),
23             Post(R.drawable.photo4, caption: "Korea"),
24             Post(R.drawable.photo5, caption: "Empty street"),
25             Post(R.drawable.photo6, caption: "la la la"),
26             Post(R.drawable.photo7, caption: "Programming"),
27             Post(R.drawable.photo8, caption: "Post 8"),
28             Post(R.drawable.photo9, caption: "Exotic"),
29             Post(R.drawable.photo1, caption: "Pool"),
30         )
31
32         val adapter = PostAdapter(samplePosts)
33         binding.recyclerView.layoutManager = LinearLayoutManager(context = this)
34         binding.recyclerView.adapter = adapter
35     }
}
```

This is MainActivity.kt file. I added some photos and text captions to show in the feed.

```
val adapter = PostAdapter(samplePosts)
binding.recyclerView.layoutManager = LinearLayoutManager(this)
binding.recyclerView.adapter = adapter
```

MainActivity Setup:

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

Assignment 1



Middle of the night



Beauty of the autumn



Being loved ^{AA}



4 spac