Task Core 1 – Spike: Taking Chances

Link to GitHub repository: https://github.com/SoftDevMobDevJan2024/core1-104177995

Goals:

- Showcase the individual's proficiency via an individual activity app.
- Recognize the many stages of an activity.
- For testing and debugging, use the IDE and logs
- Create layouts that have numerous constraints.
- Effectively employ listeners as needed.
- Develop an application with multilingual support.

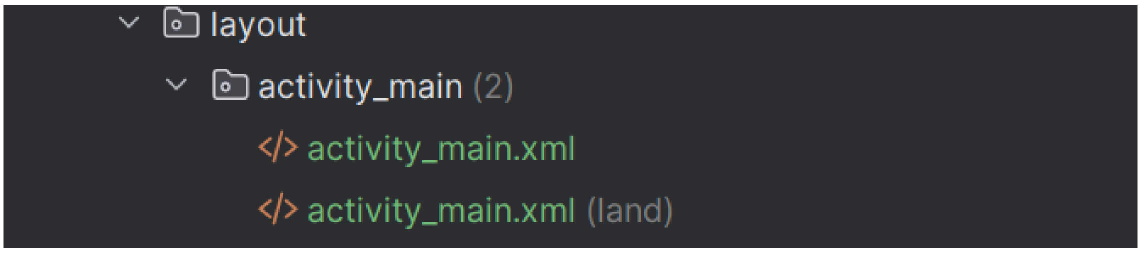
Tools and Resources Used

- The course's modules
- GitHub and Git
- Kotlin programming language and XML files
- Android Studio IDE
- Android Developer: https://developer.android.com/guide/

Knowledge Gaps and Solutions

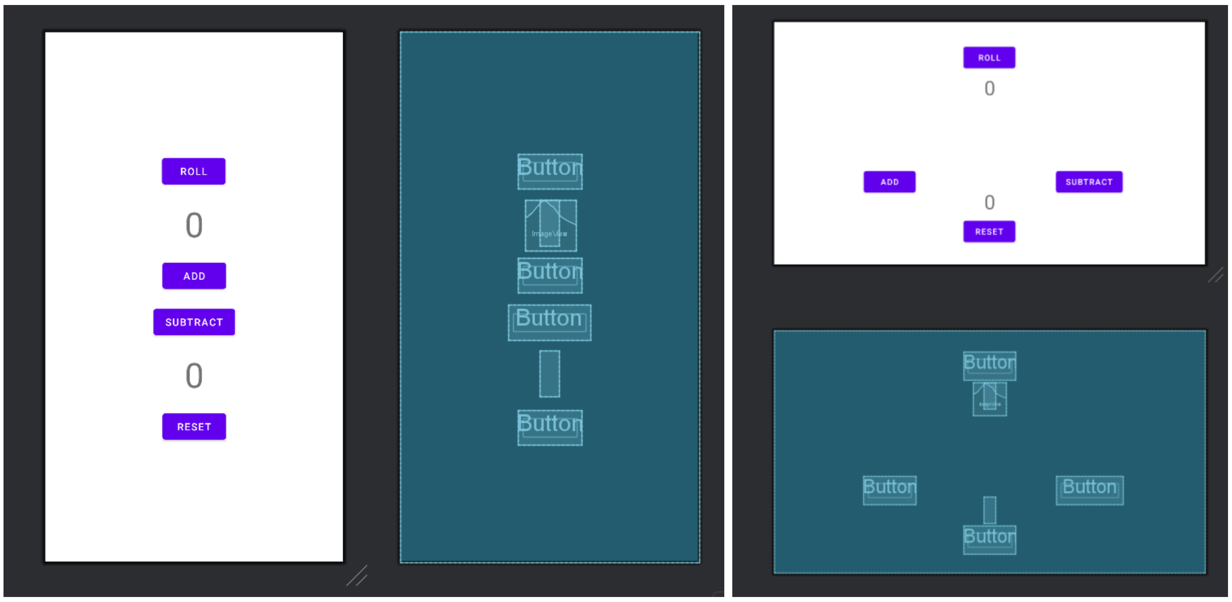
Gap 1: 2 layouts for landscape and portrait screen

By default, the app is oriented in portrait mode. An additional layout resource file must be generated in the `activity_main` directory in order to generate a landscape layout.

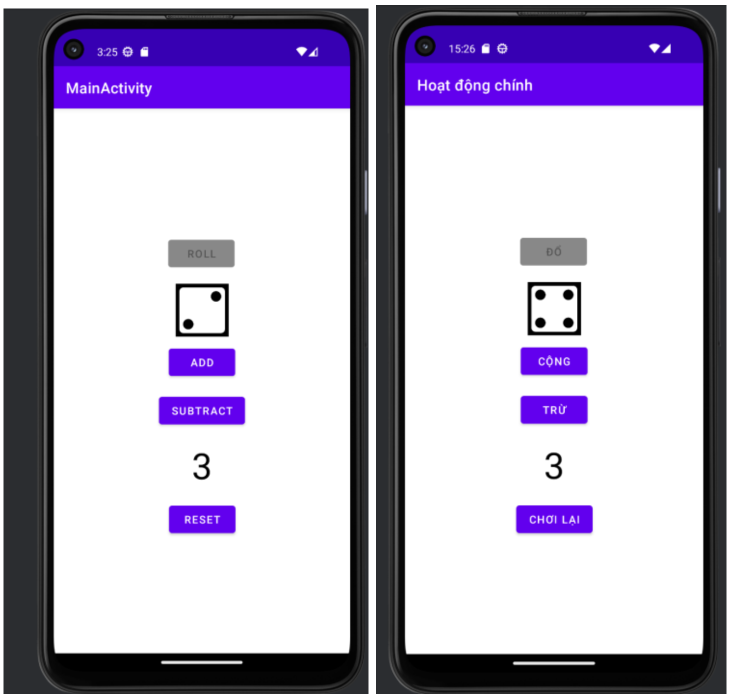


Gap 2: Constraint Layout deployment

I made my mind to utilize a constraint layout for both the landscape and portrait orientations after observing the demonstration footage and carefully reading the assignment's description, which stated that at least 1 layout had to be a constraint layout.

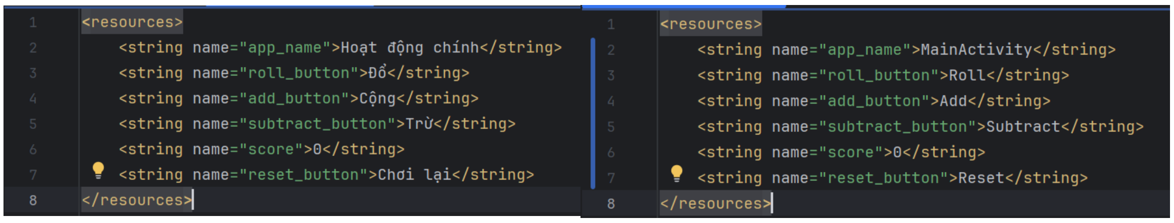


Gap 3: Incoporating localisation



There are two languages versions of the app: English and Vietnamese. In order to accomplish this, I make an additional resource directory for values, pick the Vietnamese language, and select the locale qualifier. The default language is currently set to English.

Except for the string values between the string tags, the resource files must be exactly the same. The IDs kept in both of the strings.xml files are used by string externalization to provide this functionality. An application's development and usability are improved when the device's language setting is modified because the matching resource file is utilized.



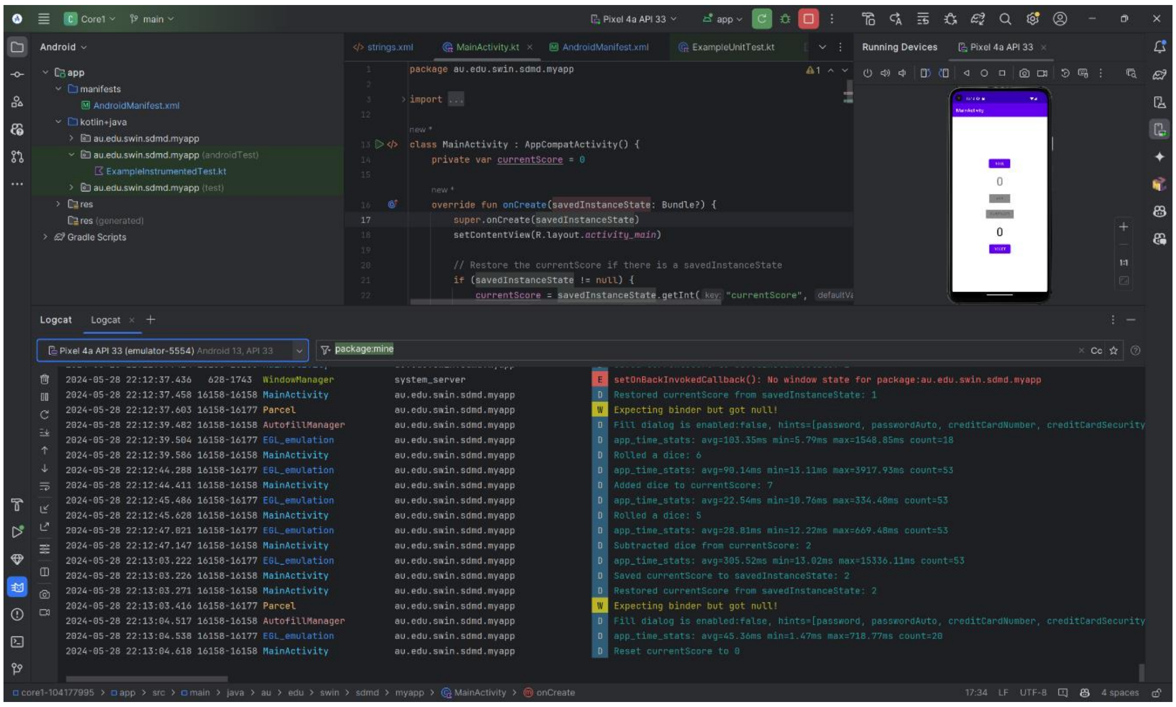
```xml
<resources>
    <string name="app_name">Hoạt động chính</string>
    <string name="roll_button">Đổ</string>
    <string name="add_button">Cộng</string>
    <string name="subtract_button">Trừ</string>
    <string name="score">0</string>
    <string name="reset_button">Chơi lại</string>
</resources>
```

```xml
<resources>
    <string name="app_name">MainActivity</string>
    <string name="roll_button">Roll</string>
    <string name="add_button">Add</string>
    <string name="subtract_button">Subtract</string>
    <string name="score">0</string>
    <string name="reset_button">Reset</string>
</resources>
```

Gap 4: Employing Logs

At different stages of the activity's life cycle and interactions, log messages are appended in order to inspect the logs in this instance. Log messages are inserted, for example, when the activity is created, restored, or when buttons are touched, to offer comprehensive debugging information about these occurrences. We might inspect these logs by opening Logcat once the program has executed. This is the code's implementation of logging:

```
roll.setOnClickListener {
    dice = diceRoller.roll()
    Log.d("MainActivity", "Rolled a dice: $dice")
   }
```



```kotlin
package au.edu.swin.sdmd.myapp

import ...

class MainActivity : AppCompatActivity() {
    private var currentScore = 0

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

        // Restore the currentScore if there is a savedInstanceState
        if (savedInstanceState != null) {
            currentScore = savedInstanceState.getInt( key: "currentScore",  defaultVa
```

```
Pixel 4a API 33 (emulator-5554) Android 13, API 33    package:mine
2024-05-28 22:12:37.436   628-1743  WindowManager     system_server          E  setOnBackInvokedCallback(): No window state for package:au.edu.swin.sdmd.myapp
2024-05-28 22:12:37.458 16158-16158 MainActivity      au.edu.swin.sdmd.myapp D  Restored currentScore from savedInstanceState: 1
2024-05-28 22:12:37.603 16158-16177 Parcel            au.edu.swin.sdmd.myapp W  Expecting binder but got null!
2024-05-28 22:12:39.482 16158-16158 AutofillManager   au.edu.swin.sdmd.myapp D  Fill dialog is enabled:false, hints=[password, passwordAuto, creditCardNumber, creditCardSecurity
2024-05-28 22:12:39.504 16158-16177 EGL_emulation     au.edu.swin.sdmd.myapp D  app_time_stats: avg=103.35ms min=5.79ms max=1548.85ms count=18
2024-05-28 22:12:39.586 16158-16158 MainActivity      au.edu.swin.sdmd.myapp D  Rolled a dice: 6
2024-05-28 22:12:44.288 16158-16177 EGL_emulation     au.edu.swin.sdmd.myapp D  app_time_stats: avg=90.14ms min=13.11ms max=3917.93ms count=53
2024-05-28 22:12:44.411 16158-16158 MainActivity      au.edu.swin.sdmd.myapp D  Added dice to currentScore: 7
2024-05-28 22:12:45.486 16158-16177 EGL_emulation     au.edu.swin.sdmd.myapp D  app_time_stats: avg=22.54ms min=10.76ms max=334.48ms count=53
2024-05-28 22:12:45.628 16158-16158 MainActivity      au.edu.swin.sdmd.myapp D  Rolled a dice: 5
2024-05-28 22:12:47.021 16158-16177 EGL_emulation     au.edu.swin.sdmd.myapp D  app_time_stats: avg=28.81ms min=12.22ms max=669.48ms count=53
2024-05-28 22:12:47.147 16158-16158 MainActivity      au.edu.swin.sdmd.myapp D  Subtracted dice from currentScore: 2
2024-05-28 22:13:03.222 16158-16177 EGL_emulation     au.edu.swin.sdmd.myapp D  app_time_stats: avg=305.52ms min=13.02ms max=15336.11ms count=53
2024-05-28 22:13:03.226 16158-16158 MainActivity      au.edu.swin.sdmd.myapp D  Saved currentScore to savedInstanceState: 2
2024-05-28 22:13:03.271 16158-16158 MainActivity      au.edu.swin.sdmd.myapp D  Restored currentScore from savedInstanceState: 2
2024-05-28 22:13:03.416 16158-16177 Parcel            au.edu.swin.sdmd.myapp W  Expecting binder but got null!
2024-05-28 22:13:04.517 16158-16158 AutofillManager   au.edu.swin.sdmd.myapp D  Fill dialog is enabled:false, hints=[password, passwordAuto, creditCardNumber, creditCardSecurity
2024-05-28 22:13:04.538 16158-16177 EGL_emulation     au.edu.swin.sdmd.myapp D  app_time_stats: avg=45.36ms min=1.47ms max=718.77ms count=20
2024-05-28 22:13:04.618 16158-16158 MainActivity      au.edu.swin.sdmd.myapp D  Reset currentScore to 0
```

Logging also helps track the state when the activity's state changes, such as during screen rotations. To achieve this, we override the `onSaveInstanceState()` function:

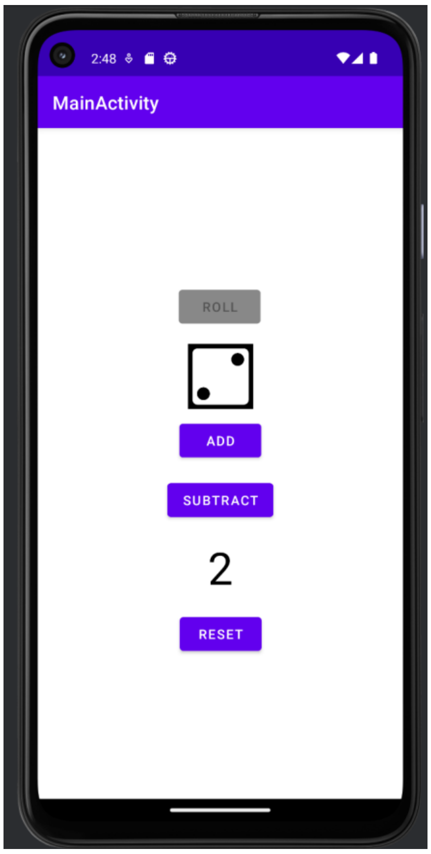
```
override fun onSaveInstanceState(outState: Bundle) {
    super.onSaveInstanceState(outState)
    outState.putInt("currentScore", currentScore)
    Log.d("MainActivity", "Saved currentScore to
    savedInstanceState: $currentScore")
}x
```

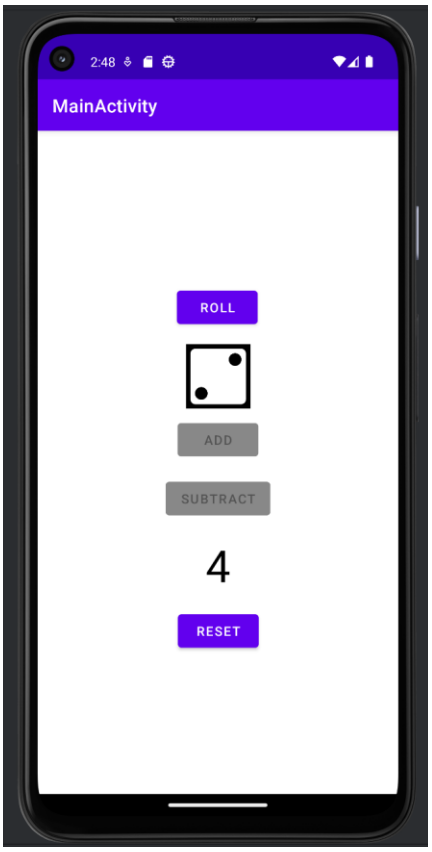
This setup ensures that the current score is logged and can be restored correctly after state changes, making it easier to debug and maintain the app.

Gap 5: Utilizing listeners

The listeners were used as the backbone of the program's logic and functions. I have used the listeners to manage crucial interactions, such as rolling the dice, incrementing/decrementing the score, and resetting the game. For example, the listener for the roll button rolls the dice and updates the UI with the new dice value. Similarly, listeners for the add, sub, and reset buttons handle score modifications and UI updates. The listener for the add button is shown in the code snippet below:

```
add.setOnClickListener {
     currentScore = score.text.toString().toInt()
     currentScore += dice
     score.text = currentScore.toString()
     Log.d("MainActivity", "Added dice to currentScore: $currentScore")
}
```

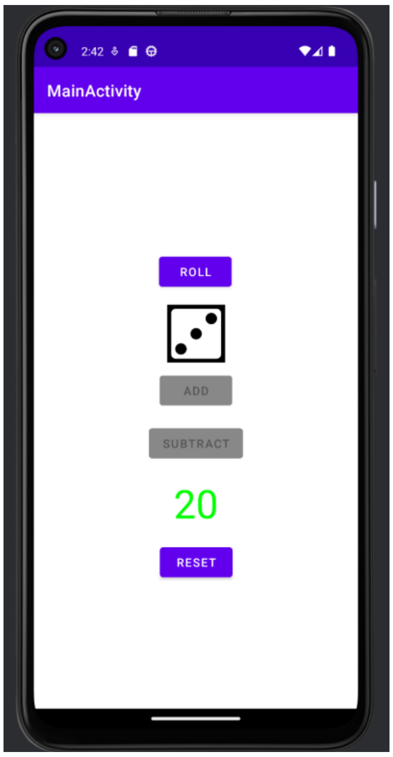


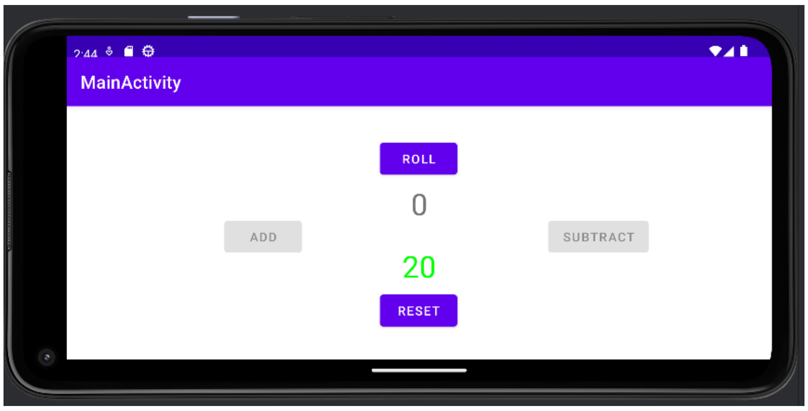


Gap 6: Preserving the state after rotating applications

For this part, the onSaveInstanceState method is utilized to save the current score, ensuring its preservation upon screen rotation. This is demonstrated in the code snippet below. The score TextView is the element that requires saving upon screen rotation.

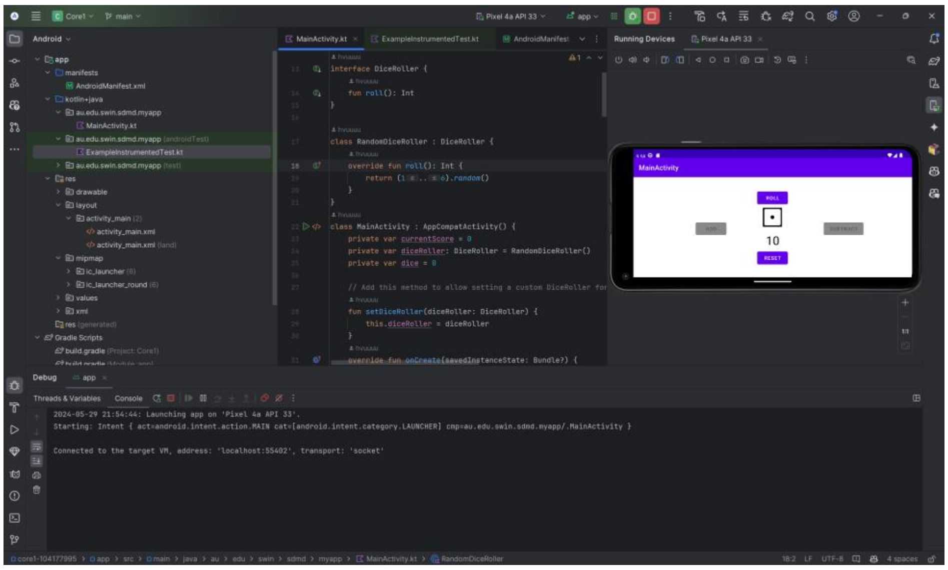
```
override fun onSaveInstanceState(outState: Bundle) {

        super.onSaveInstanceState(outState)

        val score = findViewById<TextView>(R.id.score)
        // Save the currentScore and text color into the outState Bundle

        outState.putInt("currentScore", currentScore)

        outState.putInt("textColor", score.currentTextColor)

        Log.d("MainActivity", "Saved currentScore to savedInstanceState: $currentScore")

}
```





Gap 7: IDE application

I found and fixed bugs during the working process by using the aid of the IDE's run, debug, test, and log capabilities.



```
interface DiceRoller {
    fun roll(): Int
}

class RandomDiceRoller : DiceRoller {
    override fun roll(): Int {
        return (1 .. 6).random()
    }
}
class MainActivity : AppCompatActivity() {
    private var currentScore = 0
    private var diceRoller: DiceRoller = RandomDiceRoller()
    private var dice = 0

    // Add this method to allow setting a custom DiceRoller for
    fun setDiceRoller(diceRoller: DiceRoller) {
        this.diceRoller = diceRoller
    }
    override fun onCreate(savedInstanceState: Bundle?) {
```

```
2024-05-29 21:54:44: Launching app on 'Pixel 4a API 33'.
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=au.edu.swin.sdmd.myapp/.MainActivity }

Connected to the target VM, address: 'localhost:55402', transport: 'socket'
```

Open Issues and Recommendations

During the development of the app, I encountered an issue during the testing phase. Despite the app functioning correctly during manual testing and not introducing any new errors, the automated tests failed again and again. Upon investigation, I discovered that the randomness of the dice rolls, which produce values between 1 and 6, made it impossible to consistently produce the exact results needed for comparison in some tests. To solve this, I set the dice roller to use a predefined list of fixed dice rolls. The code I used is shown below:

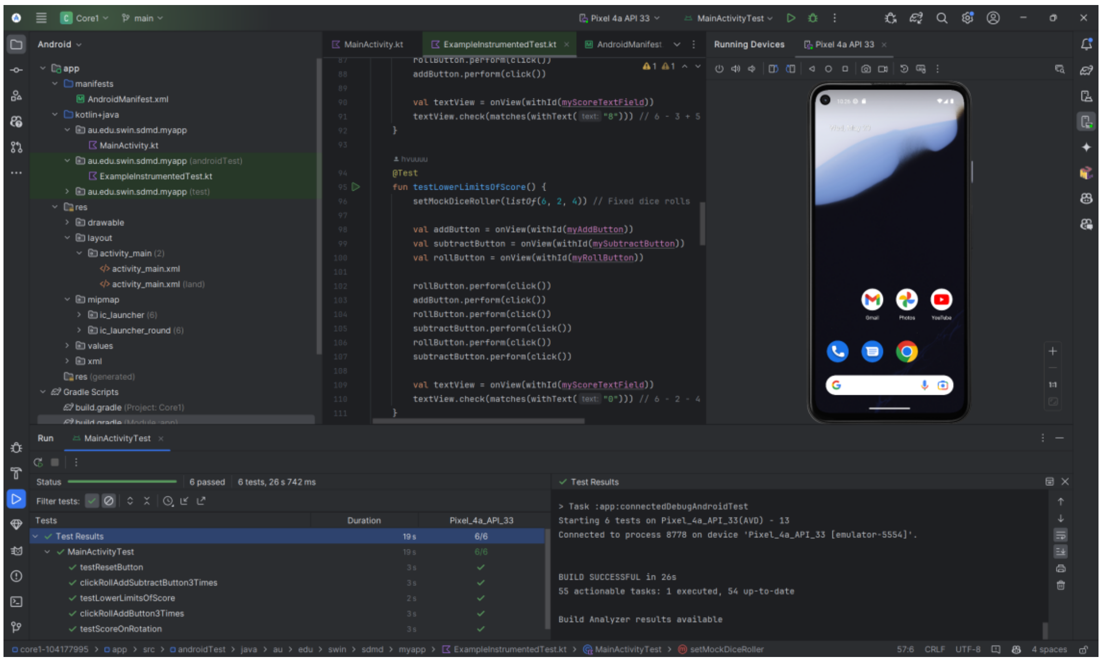
- Added code:

```
class MockDiceRoller(private val fixedRolls: List<Int>) : DiceRoller {
    private var index = 0
    override fun roll(): Int {
        val result = fixedRolls[index % fixedRolls.size]
        index++
        return result
    }
}
```

Furthermore, I still need to modify the rule in the class MainActivityTest to replace the random dice roller with a mock dice roller that returns predefined values.

```
private fun setMockDiceRoller(fixedRolls: List<Int>) {
        mActivityScenarioRule.scenario.onActivity { activity ->
            activity.setDiceRoller(MockDiceRoller(fixedRolls))
        }
}
```

With the adjusted code, the test ensures that every call to roll the dice will return values from the `fixedRolls` list in order, allowing for precise control over the dice roll outcomes during testing. The test returned passed as the screenshot below.



```
        addButton.perform(click())

        val textView = onView(withId(myScoreTextField))
        textView.check(matches(withText( text: "8"))) // 6 - 3 + 5
    }

    @Test
    fun testLowerLimitsOfScore() {
        setMockDiceRoller(listOf(6, 2, 4)) // Fixed dice rolls

        val addButton = onView(withId(myAddButton))
        val subtractButton = onView(withId(mySubtractButton))
        val rollButton = onView(withId(myRollButton))

        rollButton.perform(click())
        addButton.perform(click())
        rollButton.perform(click())
        subtractButton.perform(click())
        rollButton.perform(click())
        subtractButton.perform(click())

        val textView = onView(withId(myScoreTextField))
        textView.check(matches(withText( text: "0"))) // 6 - 2 - 4
    }
```

Status 6 passed 6 tests, 26 s 742 ms

```
Tests                               Duration   Pixel_4a_API_33
✓ Test Results                      19 s       6/6
  ✓ MainActivityTest                19 s       6/6
    ✓ testResetButton               3 s        ✓
    ✓ clickRollAddSubtractButton3Times  3 s    ✓
    ✓ testLowerLimitsOfScore        2 s        ✓
    ✓ clickRollAddButton3Times      3 s        ✓
    ✓ testScoreOnRotation           3 s        ✓
```

```
> Task :app:connectedDebugAndroidTest
Starting 6 tests on Pixel_4a_API_33(AVD) - 13
Connected to process 8778 on device 'Pixel_4a_API_33 [emulator-5554]'.


BUILD SUCCESSFUL in 26s
55 actionable tasks: 1 executed, 54 up-to-date

Build Analyzer results available
```

Core1
main
Pixel 4a API 33
MainActivityTest
MainActivity.kt
ExampleInstrumentedTest.kt
AndroidManifest
Running Devices
Run
Test Results
core1-104177995 > app > src > androidTest > java > au > edu > swin > sdmd > myapp > ExampleInstrumentedTest.kt > MainActivityTest > setMockDiceRoller