**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное учреждение

высшего образования «Уральский федеральный университет

имени первого Президента России Б.Н. Ельцина»

Институт радиоэлектроники и информационных технологий - РТФ

Центр ускоренного обучения

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ 9**

по дисциплине «Разработка мобильных приложений»

**Тема: Unit 1 Начало работы “Модульные тесты”**

Студент группы РИЗ-300028у: А.С. Севостьянов

Преподаватель: С.И.Тимошенко,

доц., к.т.н.

**Екатеринбург 2022**

содержание

[1 Постановка задачи 3](#_Toc118041527)

[2 Результаты проделанной работы 4](#_Toc118041528)

[3 Анализ полученных результатов 5](#_Toc118041529)

## 1 Постановка задачи

Запустить начальные тесты в приложении SimpleCalc. Добавить дополнительные тесты в приложение SimpleCalc. Запустить модульные тесты, чтобы увидеть результаты.

.

## Результаты проделанной работы

На рисунке 1 показан результат запуска проекта.

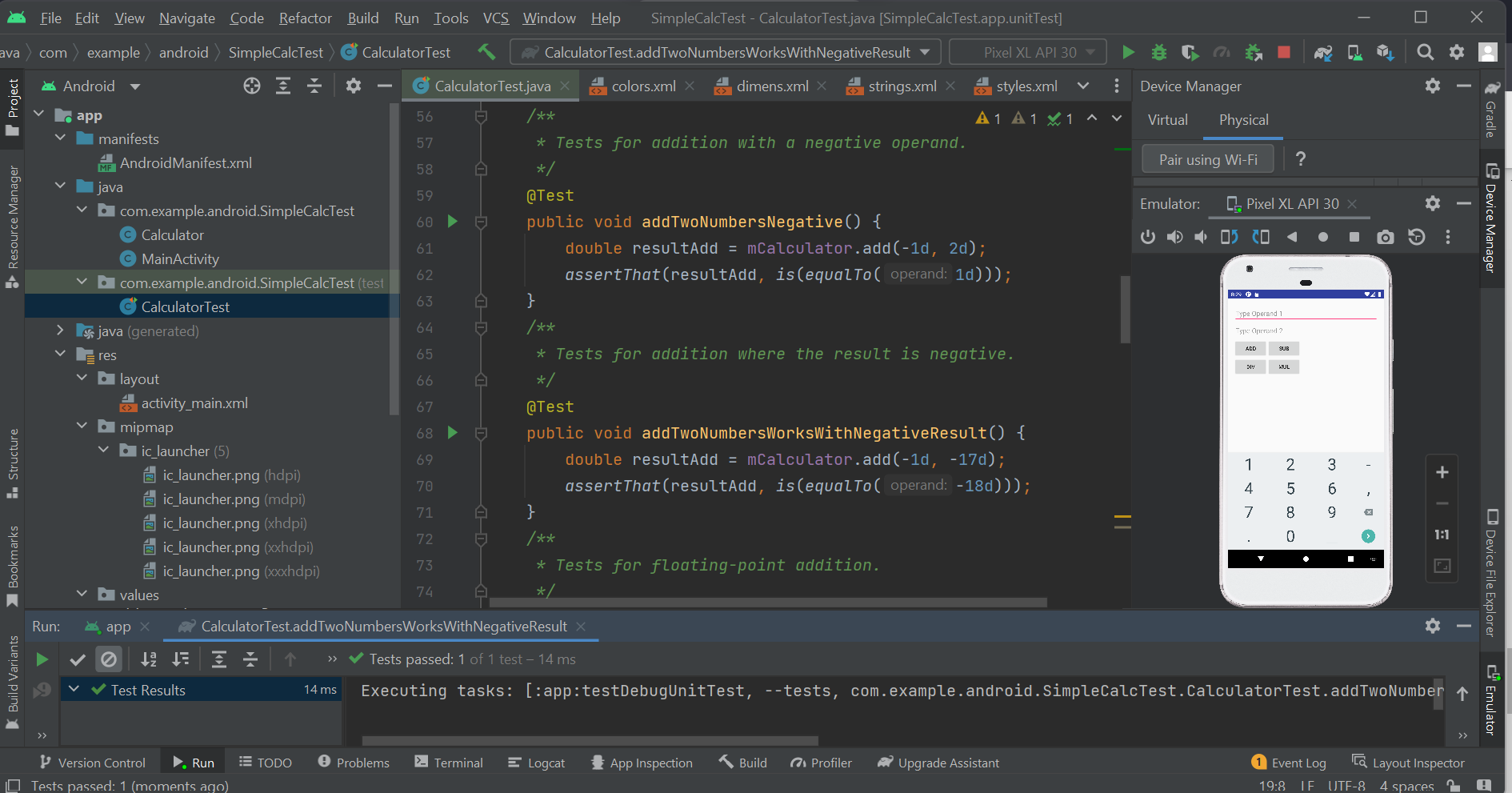


Рисунок 1 – исправно функционирующие приложения.

Добавили в проект вызываемый модульный тест subTwoNumbers(), который проверяет sub() метод. Добавили модульный тест, subWorksNegativeResurse() который проверяет sub() метод, в котором данное вычисление приводит к отрицательному числу. Добавили вызываемый модульный тест, mulTwoNumbers () который проверяет mul() метод. Добавили модульный тест, mulTwoNumbersZero() который проверяет mul() метод, по крайней мере, с одним аргументом, равным нулю. Добавили модульный тест, divTwoNumbers() который проверяет div() метод с двумя ненулевыми аргументами. Добавили вызываемый модульный тест divTwoNumbersZero(), который проверяет sub() метод с double делением и нулем в качестве делителя.

Программный код проекта, файл CalculatorTest.

package com.example.android.SimpleCalcTest;  
  
import android.test.suitebuilder.annotation.SmallTest;  
import org.junit.Before;  
import org.junit.Test;  
import org.junit.runner.RunWith;  
import org.junit.runners.JUnit4;  
import static org.hamcrest.CoreMatchers.*equalTo*;  
import static org.hamcrest.CoreMatchers.*is*;  
import static org.junit.Assert.*assertThat*;  
  
@RunWith(JUnit4.class)  
@SmallTest  
public class CalculatorTest {  
  
 private Calculator mCalculator;  
  
 */\*  
 Настраивает среду для тестирования.  
 \*/* @Before  
 public void setUp() {  
 mCalculator = new Calculator();  
 }  
  
 */\*  
 Тесты для простого сложения..  
 \*/* @Test  
 public void addTwoNumbers() {  
 double resultAdd = mCalculator.add(1d, 1d);  
 *assertThat*(resultAdd, *is*(*equalTo*(2d)));  
 }  
 */\*  
 Тесты на сложение с отрицательным операндом  
 \*/* @Test  
 public void addTwoNumbersNegative() {  
 double resultAdd = mCalculator.add(-1d, 2d);  
 *assertThat*(resultAdd, *is*(*equalTo*(1d)));  
 }  
 */\*  
 Тесты на сложение, где результат отрицательный.  
 \*/* @Test  
 public void addTwoNumbersWorksWithNegativeResult() {  
 double resultAdd = mCalculator.add(-1d, -17d);  
 *assertThat*(resultAdd, *is*(*equalTo*(-18d)));  
 }  
 */\**

*Тесты для сложения с плавающей запятой.  
 \*/* @Test  
 public void addTwoNumbersFloats() {  
 double resultAdd = mCalculator.add(1.111d, 1.111d);  
 *assertThat*(resultAdd, *is*(*equalTo*(2.222)));  
 }  
  
 */\*\*  
 \* Тесты для особо больших чисел..  
 \*/* @Test  
 public void addTwoNumbersBignums() {  
 double resultAdd = mCalculator.add(123456781d, 111111111d);  
 *assertThat*(resultAdd, *is*(*equalTo*(234567892d)));  
 }  
 */\*  
 Тесты для простого вычитания.  
 \*/* @Test  
 public void subTwoNumbers() {  
 double resultSub = mCalculator.sub(1d, 1d);  
 *assertThat*(resultSub, *is*(*equalTo*(0d)));  
 }  
  
 */\*  
 Тесты на простое вычитание с отрицательным результатом.  
 \*/* @Test  
 public void subWorksWithNegativeResult() {  
 double resultSub = mCalculator.sub(1d, 17d);  
 *assertThat*(resultSub, *is*(*equalTo*(-16d)));  
 }  
  
 */\*  
 Тесты на простое деление..  
 \*/* @Test  
 public void divTwoNumbers() {  
 double resultDiv = mCalculator.div(32d,2d);  
 *assertThat*(resultDiv, *is*(*equalTo*(16d)));  
 }  
  
 */\*  
 Тесты для деления на ноль; должны вызывать исключение IllegalArgumentException.  
 \*/* @Test(expected = IllegalArgumentException.class)  
 public void divByZeroThrows() {  
 mCalculator.div(32d,0d);  
 }  
  
 */\*  
 Тесты на деление на ноль; всегда терпит неудачу, поэтому удален.*

*\*/* @Test  
 public void divTwoNumbersZero() {  
 double resultDiv = mCalculator.div(32d,0);  
 assertThat(resultDiv, is(equalTo(Double.POSITIVE\_INFINITY)));  
 }\*/  
  
  
 */\*  
 Тесты на простое умножение.  
 \*/* @Test  
 public void mulTwoNumbers() {  
 double resultMul = mCalculator.mul(32d, 2d);  
 *assertThat*(resultMul, *is*(*equalTo*(64d)));  
 }  
  
}

## **Анализ полученных результатов**

В результате выполнения данной лабораторной работы мной был разработан проект, в котором я научился организовывать и запускать тесты в Android Studio. Научился использовать модульные тесты.