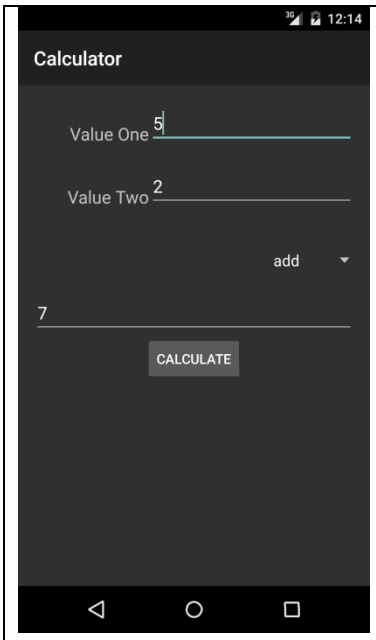
Mini-Project: Calculator App

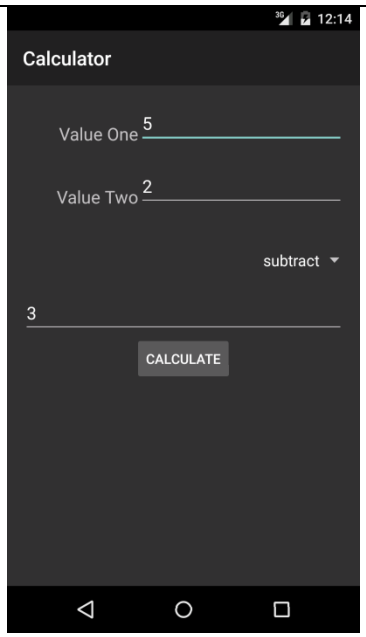
Mini-Project Overview and Requirements

In this mini-project assignment you'll write the program logic needed to complete a simple calculator app in Java. This app will perform integer arithmetic on values entered via components defined using Android's user interface (or UI). We'll supply you with skeleton code that implements the calculator's UI in Android. The app you implement should meet the following requirements:

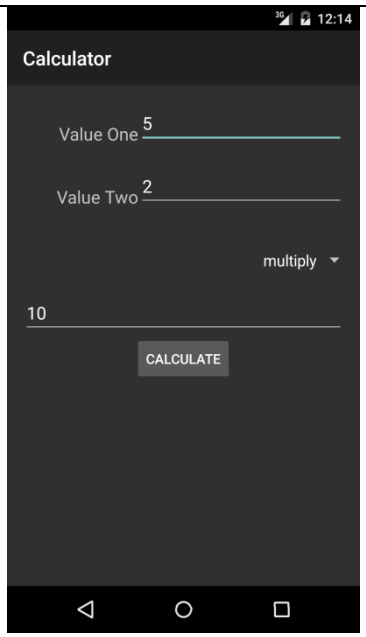
- The UI we provide allows the user to enter two integer values and to select one of four operations: addition, subtraction, multiplication, or division.
- After supplying the two integer values and pressing the "CALCULATE" button on the UI, three entities will be provided to the Java code you write: the two integer values entered by the user and the operation they selected to perform on these values.
- Your code must then perform the necessary computation and print a string that contains the final answer in the expected form. out.print(String) is how you will display the results to the App's screen.
- The final results printed for integer addition, subtraction, and multiplication are what you'd expect them to be, as shown in these screenshots:



Performing addition

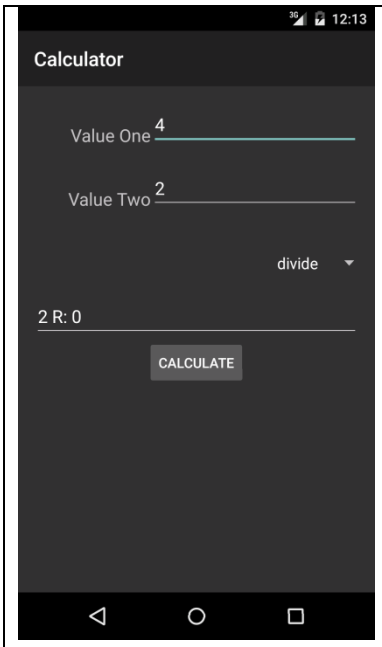


Performing subtraction

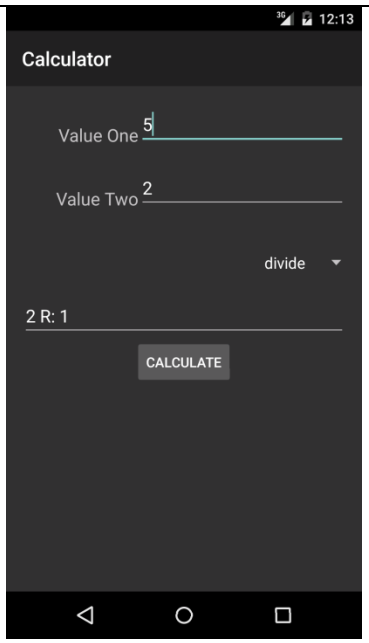


Performing multiplication

- The final result printed for integer division must include both the quotient and the remainder (even if the remainder is zero), as shown in the these screenshots:



Performing division with a zero remainder



Performing division with a non-zero remainder

- There is a special-case that you must handle with division. When the value 0 is entered in the "Value Two" box on the screen, you must not perform that calculation because an exception will be thrown, which will crash your App. Your code should therefore check to prevent that and instead send a text string to the screen warning of division by 0. The exact text to convey this is up to you, but it should be both clear and concise.

Steps for Getting Started

Please start this assignment by downloading the supplied zip file corresponding to this mini-project. Extract the contents of this zip file onto your computer. The content extracted from this zip file contains an Android Studio project. Next launch Android Studio and load the project. In addition to the UI-related classes (which you can ignore for the time being), you'll see this project supplies you with five skeleton files containing Java classes:

1. The Logic.java file contains the process() method, which receives the 3 entities passed from the UI: the two integers upon which to perform the computation and an integer value indicating the operation to perform, where the value 1=addition, 2=subtraction, 3=multiplication, and 4=division. Naturally, you should use symbolic constants for these values, rather than using "magic numbers."
2. The Add.java file contains an empty class named Add.
3. The Subtract.java file contains an empty class named Subtract.
4. The Multiply.java file contains an empty class named Multiply.
5. The Divide.java file contains an empty class named Divide.

Your work should start by modifying the skeleton file Logic.java. Open this file in the Android Studio Integrated Development Environment and look for the comment:

// TODO -- start your code here

You can then add your implementation here as you see fit. Likewise, open the other four *.java skeleton files and following similar steps until you've completed your implementation. For a more detailed analysis of the skeleton files and other parts of the UI code please watch the lesson entitled "Mini-Project Assignment Walkthrough".

Guidelines for Structuring Your Solution

We'll now discuss the guidelines for structuring your solution, which are divided into two parts.

- *Guidelines for source code design*. Given the intentionally limited capabilities of the calculator app, one solution might involve performing all the work in the process() method via a multi-branch if/else statement. However, this design would be unmanageable if we later wanted to extend our calculator to support additional operations and/or additional data types. We therefore encourage you to create an *object-oriented* solution that will provide better abstraction to simplify extensibility and refactoring in the future. In particular, to receive full credit for this assignment, you must apply Java language features we taught in recent modules of the MOOC and use the four empty *.java classes we provided by adding methods and/or instance variables to them. We also recommend you consider defining a Java interface that these four empty classes implement.
- *Guidelines for source code aesthetics (such as commenting, indentation, spacing, and identifier names).* You're required to properly indent your code and will lose points if you make significant indentation mistakes or inconsistencies. None of your lines of code should be longer than 80 characters. Please also use a programming style that's easy to read and maintain by doing things like:
  - Creating additional helper methods as needed
  - Using meaningful variable and method names
  - Making the code more readable by using "white-space" and blank lines appropriately and
  - Explaining tricky pieces of code with useful comments

Steps for Submitting Your Mini-Project Solution

For this assignment, you'll need to submit a zip file containing all the Java files and Android Studio project files needed to compile and run your calculator app. To build this zip file,

open the Gradle window in Android Studio, which is located along the far right edge of the screen all of the way up. Then go to the [m8-assignment-calculator] -> [Tasks] -> [other] menu item and double click on [_projectZip] to generate the zip file of your project in the <project root directory>/zip/ folder. After you locate this zip file on your drive, you can upload it to the Coursera platform.

Steps for Assessing Peer Solutions

Unlike previous assignments in this MOOC that were auto-graded, this mini-project will be purely peer assessed, which involves giving and receiving feedback from other learners in the MOOC. There are two steps involved in peer assessing programming assignments: First, you'll submit your assignment by following the steps we just discussed, at which point you'll be granted access to a video that walks through our solution so you can see how we implemented the calculator app. Second, you'll then review 5 submissions made by your peers, using a grading rubric we supply you with to guide your peer assessments. Your final grade on this mini-project will be calculated using the median scores you received from your peers on each assignment component. Note that there's a 20% penalty for not evaluating your peers, so please make sure to follow these instructions so that everyone benefits from helpful feedback.

As you do your evaluation, please keep an open mind and focus on the positive. Our goal is not to find every way to deduct points over small deviations from the requirements or for legitimate differences in implementation styles. Therefore, look for ways to give points when it's clear the submitter has given a good faith effort to do the project, and when it's likely that they've succeeded. Finally, remember that almost everyone is working hard and putting in serious effort. So if you've got doubts, please error on the side of giving too many points, rather than giving too few.