Lab Assignment #1 – Developing a Basic Android Application using Kotlin and Jetpack Compose.

Due Date: Mid-night (11.59 pm) **5th Oct. 2025** **Marks/Weightage: 30/8%**

End Date: Mid-night (11.59 pm) ***8th Oct 2025 with 20% late penalty. No Exceptions***

Note: You are required to demonstrate the assignment as per scheduled lab session as announced by your teacher. 25% penalty for not demonstrating the assignment

IDE: Android Studio – Meerkat Feature Drop Version (2024.3.2) and Kotlin Jetpack Compose

Purpose: The purpose of this lab assignment is to:

- Apply the basics of the Kotlin programming language.
- Develop and implement UI components using composable functions, previews, and modifiers.
- Design and construct complex UI layouts using Column, Row, Box, and Lists in Jetpack Compose.
- Apply Jetpack Compose best practices to create responsive and maintainable user interfaces.

References: Textbook, ppt slides, class examples, and Android tutorials (**http://developer.android.com/training/basics/firstapp/creating-project.html**). This material provides the necessary information that you need to complete the exercises.

Be sure to read the following general instructions carefully:

- This assignment must be completed individually by all the students.
- You will have to **demonstrate your solution in a scheduled lab session** and upload the solution on eCentennial through the **assignment link under Assessments**.

Android Project Naming Rules:

Step 01: You must name your Android Studio **project** according to the following rule:
yourfullname_COMP304-*SectionNumber*_Labnumber
For Example: **johnsmith_COMP304-001_Lab01. Save location drive name should be C:\COMP304-001-S25\Assignments or D:\COMP304-001-S25\Assignments etc.**
***If you have more than one exercise in the assignment, then you need to create separate project for each exercise and name your exercises as* johnsmith_COMP304-001_Lab01_Ex01, johnsmith_COMP304-001_Lab01_Ex02 etc…**

Step 02: Submission rules

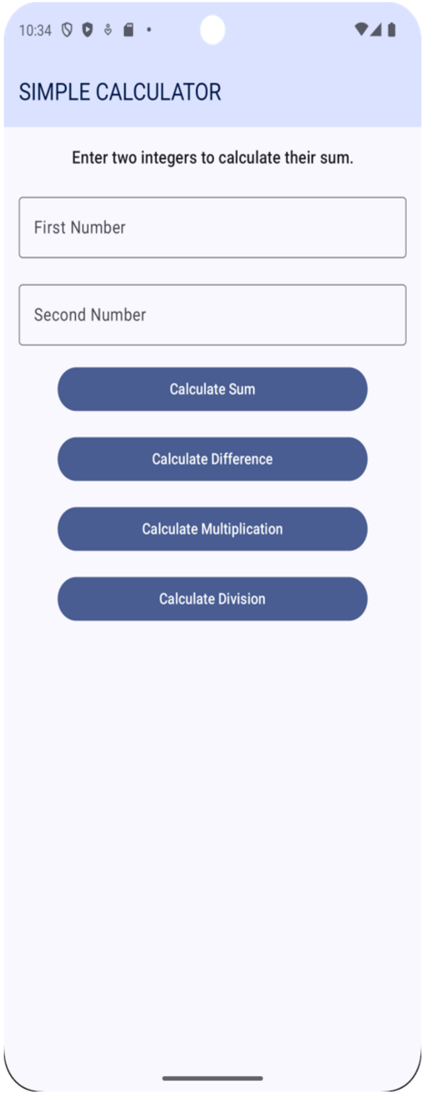
Once you complete, run and test projects for all the exercises, then submit your projects as one **zip file (ONLY .zip file format is allowed. No .rar or .7z etc. file formats allowed) and it should be** named according to the following rule:
yourfullname_COMP304-SectionNumber_Labnumber.zip.

Example: **johnsmith_COMP304-001_Lab01_ex01.zip (if your section is 001)**

Exercise 01: *[10 marks]*

Build a Simple Calculator App as shown in the screen shot below.

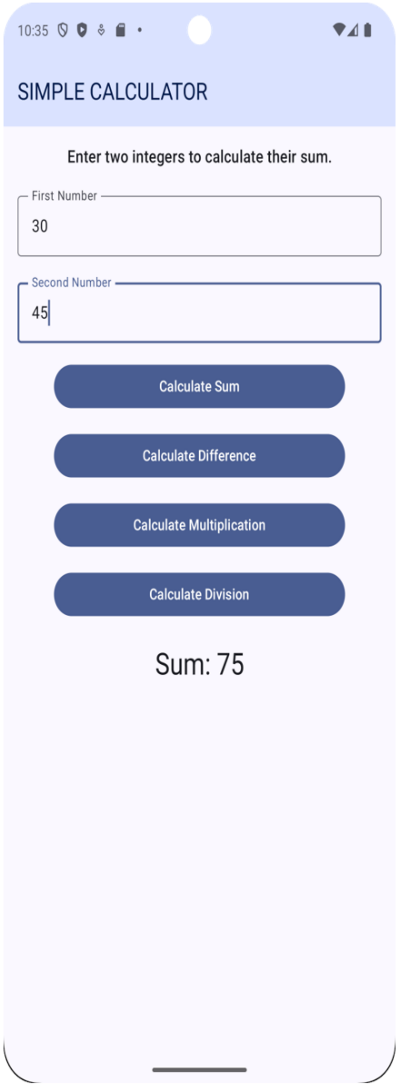
*(Refer **ButtonExample** covered during the class- under Content → WEEK04)*



Enter two integers to calculate their sum.

First Number

Second Number

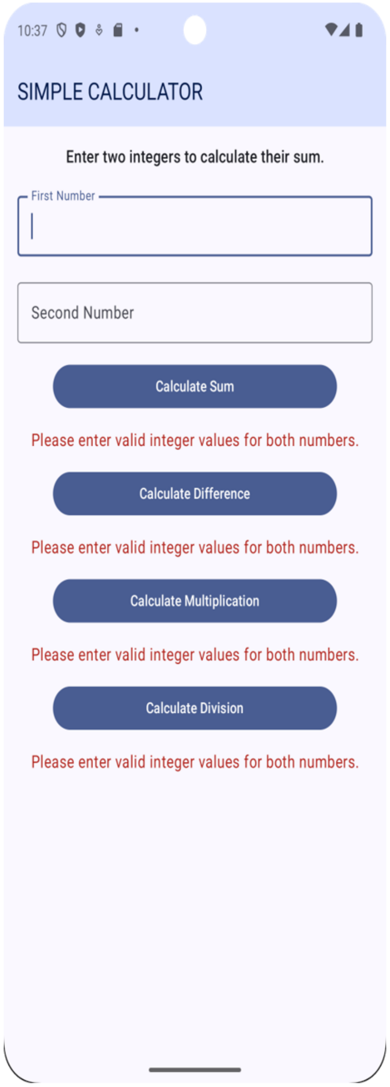


Enter two integers to calculate their sum.

First Number: 30

Second Number: 45

Sum: 75



Enter two integers to calculate their sum.

First Number

Second Number

Please enter valid integer values for both numbers.

Please enter valid integer values for both numbers.

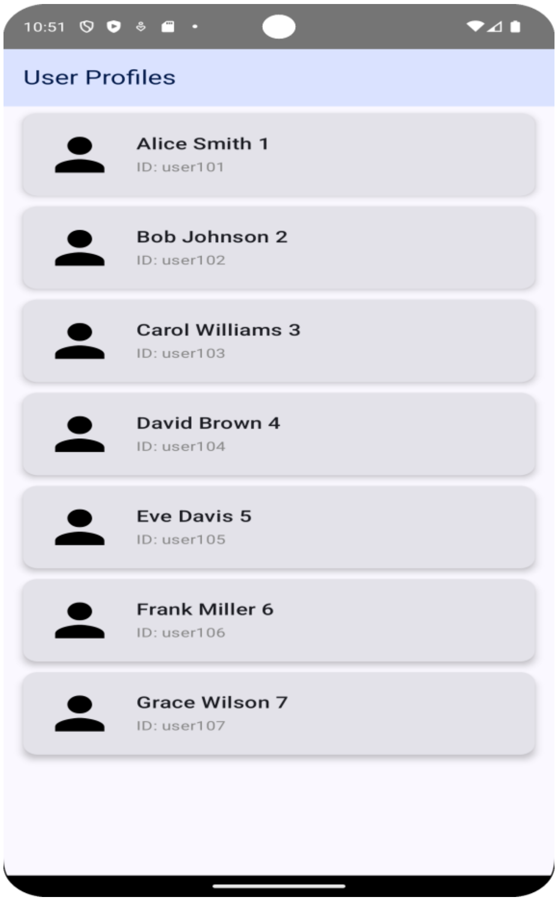
Please enter valid integer values for both numbers.

Please enter valid integer values for both numbers.

EXERCISE 02:

[10 marks]

Refer to the following screen shot using Lazy Column UI component. You need to complete/modify it as per following requirements.



User Profiles

Alice Smith 1
ID: user101

Bob Johnson 2
ID: user102

Carol Williams 3
ID: user103

David Brown 4
ID: user104

Eve Davis 5
ID: user105

Frank Miller 6
ID: user106

Grace Wilson 7
ID: user107

You need to display at least 5 products (such as iPhone, Google Pixel, Samsung S25, Moto etc.) with the following information.

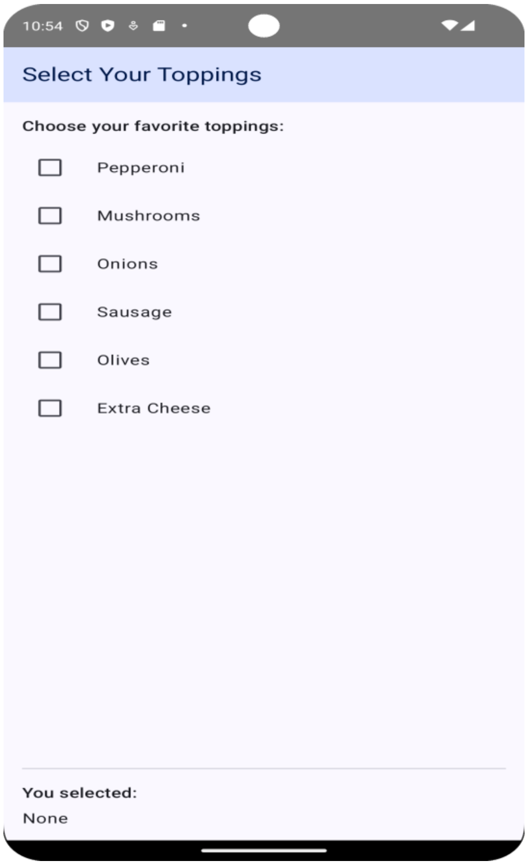
a) You need to define a data class having properties:
   i. Name of Product (such as iPhone, Google Pixel, Samsung S25, Moto etc.) with
   ii. Price
   iii. Manufacturer name
   iv. Date Launched
   v. Image picture of the product

b) Display above information using card LazyColumn

EXERCISE 03:

[10 marks]

Following App covered during the class. You need to extend it as per instructions mentioned below and build as simple pizza ordering app.



Select Your Toppings

Choose your favorite toppings:

□ Pepperoni
□ Mushrooms
□ Onions
□ Sausage
□ Olives
□ Extra Cheese

You selected:
None

a) Add three TextFields, one for Order ID and Second for Customer Name and third e-mail ID.

b) After Text Fields, add one drop down showing values for pizza size(s) – Small, Medium, Large and Extra Large

c) Pizza toppings are the same as covered in the class and shown in the screen shot above.

d) Add two radiobuttons, one for Pick-Up and Home Delivery

e) Adding button – Place Order and when it is clicked, then following information should be displayed:

   i. Order ID
   ii. Customer Name
   iii. Selected size of Pizza
   iv. Names of all the selected toppings
   v. Mode of delivery either Pick-up or delivery

1. Kotlin Basics:
   - Implement basic Kotlin programming concepts such as variables, control structures, functions, and classes.
   - Use Kotlin collections to manage the list of notes.

2. **Setup and IDE**:
   - Ensure Android Studio is set up to use Kotlin (Meerkat Feature Drop version).
   - Target Android 14 (API Level 34) or higher for your application.

Evaluation table/Rubric:

Item	Percentage of Total Mark	Details
Functionality:	**80%**	
Correct implementation of activities:		
Class code for main activity	20%	Ensure the main activity (Home Activity), Create Note Activity, and View/Edit Note Activity are correctly implemented.
UI in Jetpack Compose	20%	The UI components should be implemented using Jetpack Compose, including LazyColumn, Card, TextField, Button
Correct implementation of Event Handlers and life cycle methods:		
Event handlers for UI interactions	20%	Proper handling of user interactions, such as adding a note, editing a note, and saving changes.
Life cycle methods for activities	20%	Correct implementation of life cycle methods (e.g., onCreate, onStart, onResume) to manage activity state transitions.
Friendliness:	**15%**	
Alignments of UI controls	10%	UI controls should be properly aligned and organized, providing a visually appealing layout.
Friendly I/O	5%	The app should provide a user-friendly interface with intuitive input/output operations.
Comments, Correct Naming of Variables, Methods, Classes, etc.	**5%**	Code should be well-documented with appropriate comments. Variables, methods, and classes should follow proper naming conventions.
Total	**100%**	