Worksheet 3

CS 501 – Mobile Application Development

Fall 2022

Android Activities: Lifecycle, Resources, Localization, Bundles, Multiple Activities, Intents, Saving State

Date: _____

Team Members: _____

This worksheet is to be done in collaboration with your project team. Although we are working with Android, the concepts apply for any device.

Part 1

Activity Lifecycle Revisited, Saving State.

1. **What is the difference between Early and Late Event Binding? Provide 2 examples of when you might prefer to use Early Binding.**

_____

_____

_____

_____

_____

_____

_____

_____

_____

2. **True or False: A single event handler can be shared with multiple views. _____**

If so, how? _____

_____

_____

_____

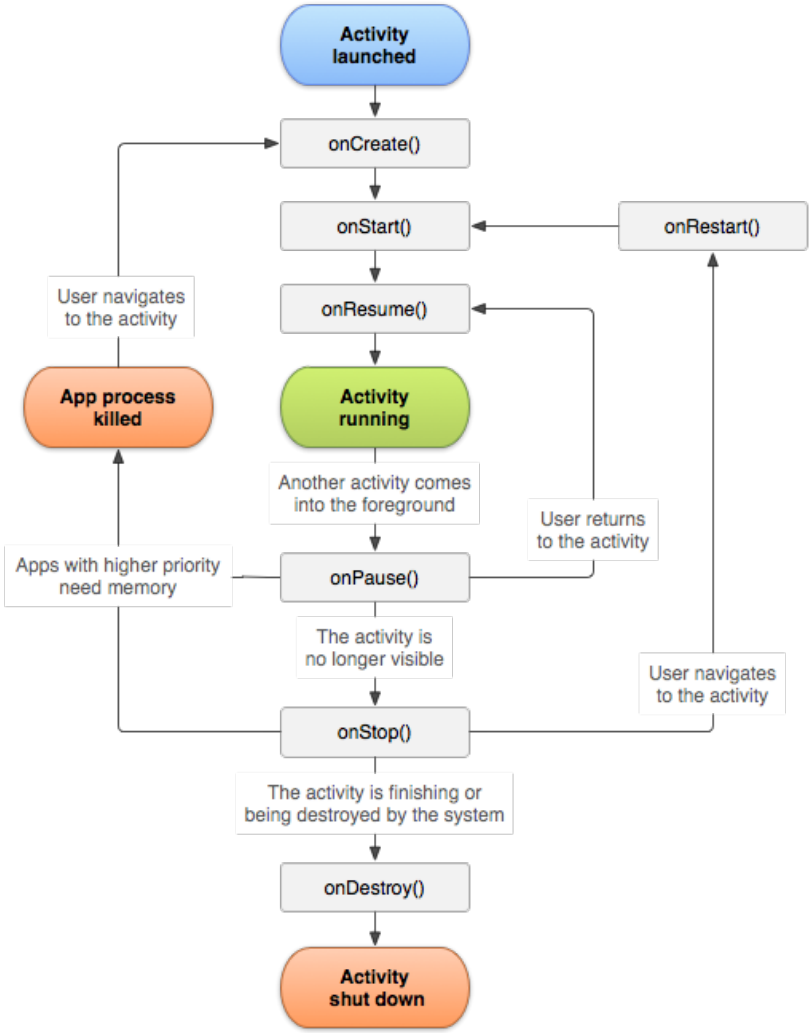
_____

_____

_____

_____

For the next few questions, consider the Activity Lifecycle Diagram which shows some new lifecycle events.



3. **True or False: The Android Framework can kill your application if it's low on memory when it stops.**
   _____

4. **True or False: When an Activity is killed, your application is killed. _____**

5. **Which Activity Lifecycle event(s) should you call upon, when storing and recovering application state, eg, when the phone is rotated.**
   _____
   _____
   _____
   _____

6. **Android utilizes the _____ class, sort of like a Hashmap of key/value pairs, to store and retrieve Activity State Information.**

7. **What is a layout in Android?**

_____

_____

_____

_____

8. **True or False: In order to render Views, an Activity must have a layout as its base element (in the XML). _____**

9. **True or False: A layout can exist within another layout? _____**

10. **Why is it important to Call super() when overriding base Activity Events?**

_____

_____

_____

_____

Part 2

Implement a program with a blank Activity that logs the Activity Lifecycle as you interact with both the Activity and the device. Log the events listed to the right. *Use the official Activity Lifecycle from Android to identify the key lifecycle events.* Submit the answers here and links to the programs on GitHub.

https://developer.android.com/guide/components/activities/activity-lifecycle.html

Life Cycle Events to Track:

- `onCreate(..)`
- `onStart(..)`
- `onRestart(..)`
- `onResume(..)`
- `onPause(..)`
- `onStop(..)`
- `onDestroy(..)`

2a. What events are called when the App first starts up?

_____

_____

_____

2b. What events are called when the user clicks the home button on the device and then "restarts" the App?

_____

_____

_____

2c. What events are called when the user hits the back button, and then "restarts" the app by clicking on its icon?

_____

_____

_____

2d. What events are called if you rotate the phone, switching between portrait and landscape then back to portrait?

_____

_____

_____

2e. What events are called if you open another application and then switch back to the app?

_____

_____

_____

2f. Now Add 3 views to your Activity. When the Button View is clicked "Hello" should appear in both EditText and the TextView. The Activity lifecycle will not change, but you may notice something *odd* about the state of the Text in the EditText/TextViews as you interact with the device.

2g. Describe what happens to the "Hello" text inside EditText & TextViews as you leave and reenter your app in the different ways described above.

_____

_____

_____

_____

_____

_____

_____

_____

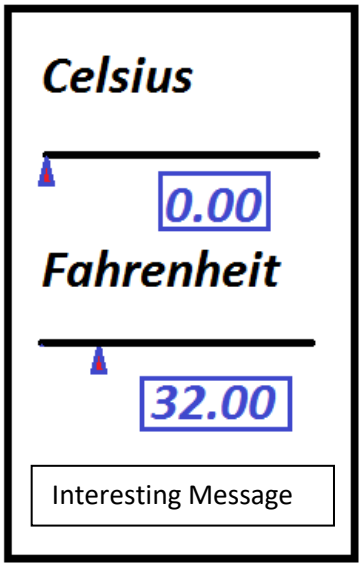
Part 3: Temperature Converter, both ways.

For this next App, you will be required to read the documentation for the Seekbar Component. Use a ConstraintLayout for this Activity.

1. **Implement a Celsius to Fahrenheit converter to utilize two SeekBar Components, one to represent Celsius the other for Fahrenheit.**
   a. **Moving one SeekBar, will not only change its own value, but also update its counterpart. For Example dragging the Fahrenheit SeekBar to 32°F will update the Celsius SeekBar to 0°C and vice versa.**
   b. **Use a Celsius range of 0°-100°C. Use a Fahrenheit Range of 0°-212°F.**
   c. **The Fahrenheit SeekBar cannot remain below 32°F. That is, if a user were to drag the Fahrenheit SeekBar below 32°F, it should automatically snap back to 32°F, leaving the Celsius value at 0°C.**

2. **The *Interesting Message* should read `"I wish it were warmer."` when the temperature is less than or equal to 20°C degrees, otherwise it should read `"I wish it were colder."`. Choose whatever min/max ranges you like for each seekbar.**

3. **Localize your App for two additional languages of your choosing. You must localize the strings, "Celsius", "Fahrenheit", as well as the two *Interesting Messages* above. *Note: Be sure to choose languages that your phone can render.***



Part 4 - Flash Card App

1. Multiple Activities, supporting both Landscape and Portrait Mode.

Create a flashcard App with two activities, a simple login, for which you may simulate logging in with a hardcoded username and password (please tell us what these are). After entering the correct user/pass combination, the user should be routed to a Flash Card Activity.

a. When the user presses the "Generate 10 Random problems" button, 10 random addition and subtraction problems will be generated. Out of 10 problems, approximately half should be addition and half subtraction, these are random with a 50/50 probability of getting an addition or subtraction problem, each time.

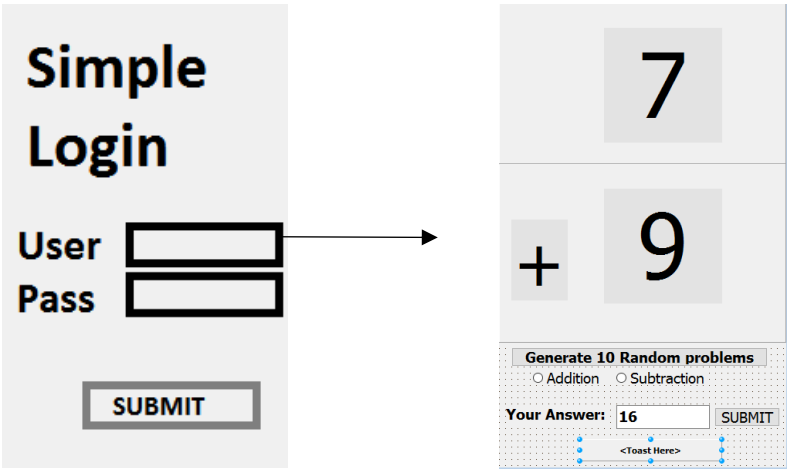
b. Your program will utilize Java as the code-behind layer of your Activity and will generate 10 random problems, students will enter the answer into the textbox, then depress submit. When the user completes all 10 answers, just provide a Toast with their score, eg, "9 out of 10".

c. The top operand, (i.e., - Operand 1) must always be between 1-99. Operand 2 must always be between 1-20.

d. Be sure the user can restart the game and play multiple times. Feel free to use the template below as a guide (just a guide, not a requirement).

e. The "Generate 10 Random problems" button should become disabled once the game starts. After the users submits answers to all 10 problems, will the button become re-enabled.

We will improve upon the design later, don't worry about telling students which problems they got right or wrong, or giving them immediate or delayed feedback.



1. Provide a descriptive Storyboard for your App. Much of the design and user interaction is up to you. Use the above template as a guide.

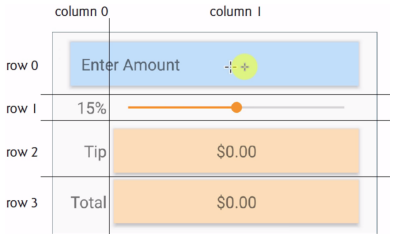
2. Update the App so that once the user successfully logs in, a Toast is displayed when the second Activity starts up. "Welcome <user>", where <user> is the username entered in the first Activity.

3. The user should not have to start over just because the screen was rotated. Update the code to store the user's score and other important state information, such as the current problem on the screen, when the screen is rotated, state of the "Generate 10 Random problems" button, etc. This only applies to the second Activity (not the Login).

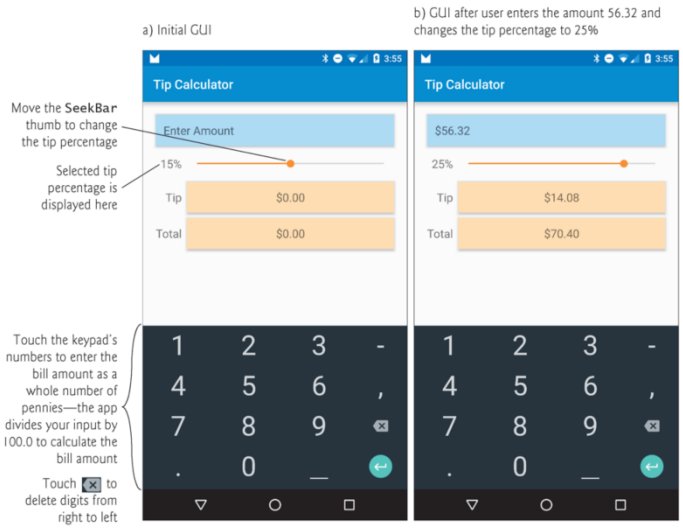
4. Modify the code once again, so that the App "Looks Good" in both portrait and landscape mode. This only applies to the second Activity (not the Login).

Part IV: Tip Calculator

Implement a tip calculator App, according to the Story board below, using any layout(s) you like. In this case your final app, must look like the storyboard outlined below. You will need to experiment with the layout properties



TIP Calculator, Labeled GridLayout View



Sample TIP Calculator at Runtime