COMP 3160
Midterm Exam (winter 2018)
Total Marks: 60
Time: 2 Hours
Date: March 19, 2018 at 5:30 pm

You need to create one single app for this exam. The development process is divided into three different tasks.

Task 1[20 Points]

In this task, you need to create an android service that will fetch data from "numberapi.com". Study http://numbersapi.com/#42. You need to process the JSON formatted output from this website.

The service (use IntenetService) will start with a query string passed to the service, then it will create a URL request that it will send to the "numberapi.com" server with proper formatting for the request. When the server sends a JSON response for the request, it will process that request and print the response(s) in the LogCat.

The input string will have one of the following values:
"random/math?jason"
"x/math?jason", where x is any positive integer value
"x..y/math?jason", where x and y both are positive integer values

"random/trivia?jason"
"x/trivia?jason", where x is any positive integer value
"x..y/trivia?jason", where x and y both are positive integer values

Try these for sample results from the API.
http://numbersapi.com/random/math?json
http://numbersapi.com/1..40/math?json
http://numbersapi.com/1..4/trivia?json
http://numbersapi.com/2015/trivia?json

Task 2: [20 Points]

In this task, you should update the service that you have created in task 1 so that it will send the request to the "numbersapi.com" server only it the response for this request is not already stored in a local database. If the response is not stored, then store the JSON response(s) in a local database on the phone. In this task, you must you ROOM database for storing data in a local database. Show the status of this update using a Toast message.

Task 3: [20 Points]

Create an UI for the app that will have two text views and two buttons. Give the buttons "Trivia" and "Math" as captions.

When users tap on the “Math” button, it will fetch math responses from the “numbersapi.com” and if the user taps on “Trivia” button, then it will fetch trivia response from the site.
Of course, before taping on these buttons, the user must enter valid numbers in the two text views. If the second text view is empty (and the first text view contains a number), then use the number from the first text view and return a single response for that number; otherwise, return batch responses for the range of numbers between the first number and second number from the website.

The result will be shown in a Recycler View to display the data from the Database created in Task 2. The recycler view will be updated automatically whenever a new item/response inserted into to the database (this will happen when the service adds a new entry into the database if that entry does not exist in the database). If the database already contains the response, then it should query the database and return the result from the local database (i.e., do not send this request to the server, instead return query result from the local database).

For this task, you must use LiveData and ViewModel architecture to update the UI.

Submission: upload your project in GitHub or Google Drive, then create a sharable link to your project and submit that link in the moodle submission page.