

ATME COLLEGE OF ENGINEERING

13th KM Stone, Bannur Road, Mysore - 560 028



A T M E
College of Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

(ACADEMIC YEAR 2021-22)

LABORATORY MANUAL



SUBJECT: MOBILE APPLICATION DEVELOPMENT

SUBJECT CODE: 18CSMP68

SEMESTER: VI

2018 CBCS Scheme

Prepared By

Verified By

Approved By

Mr. Rajiv

**Mr. Raghuram A S
Mrs. Kavya P O**

Dr. Puttegowda D

Foreman

Faculty Co-Ordinators

Professor & Head, CSE

INSTITUTIONAL MISSION AND VISION

Objectives

- To provide quality education and groom top-notch professionals, entrepreneurs and leaders for different fields of engineering, technology and management.
- To open a Training-R & D-Design-Consultancy cell in each department, gradually introduce doctoral and postdoctoral programs, encourage basic & applied research in areas of social relevance, and develop the institute as a center of excellence.
- To develop academic, professional and financial alliances with the industry as well as the academia at national and transnational levels.
- To develop academic, professional and financial alliances with the industry as well as the academia at national and transnational levels.
- To cultivate strong community relationships and involve the students and the staff in local community service.
- To constantly enhance the value of the educational inputs with the participation of students, faculty, parents and industry.

Vision

- Development of academically excellent, culturally vibrant, socially responsible and globally competent human resources.

Mission.

- To keep pace with advancements in knowledge and make the students competitive and capable at the global level.
- To create an environment for the students to acquire the right physical, intellectual, emotional and moral foundations and shine as torch bearers of tomorrow's society.
- To strive to attain ever-higher benchmarks of educational excellence.

Department of Computer Science & Engineering

Vision of the Department

- To develop highly talented individuals in Computer Science and Engineering to deal with real world challenges in industry, education, research and society.

Mission of the Department

- To inculcate professional behavior, strong ethical values, innovative research capabilities and leadership abilities in the young minds & to provide a teaching environment that emphasizes depth, originality and critical thinking.
- Motivate students to put their thoughts and ideas adoptable by industry or to pursue higher studies leading to research.

Program outcomes (POs)

Engineering Graduates will be able to:

- **PO1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems
- **PO2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
- **PO3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
- **PO4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
- **PO5. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
- **PO6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice

- **PO7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
- **PO8. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- **PO9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- **PO10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- **PO11. Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
- **PO12. Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Educational Objectives (PEO'S):

1. Empower students with a strong basis in the mathematical, scientific and engineering fundamentals to solve computational problems and to prepare them for employment, higher learning and R&D.
2. Gain technical knowledge, skills and awareness of current technologies of computer science engineering and to develop an ability to design and provide novel engineering solutions for software/hardware problems through entrepreneurial skills.
3. Exposure to emerging technologies and work in teams on interdisciplinary projects with effective communication skills and leadership qualities.
4. Ability to function ethically and responsibly in a rapidly changing environment by Applying innovative ideas in the latest technology, to become effective professionals in Computer Science to bear a life-long career in related areas.

Program Specific Outcomes (PSOs)

1. Ability to apply skills in the field of algorithms, database design, web design, cloud computing and data analytics.
2. Apply knowledge in the field of computer networks for building network and internet based applications.

MOBILE APPLICATION DEVELOPMENT

Subject Code	:	18CSMP68	IA Marks	:	40
Hours/Week	:	0:2:2	Exam Marks	:	60
Total Hours	:	3 Hours/Week	Exam Hours	:	03

Credits - 2

Laboratory Objectives: This laboratory (18CSMP68) will enable students to:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services.

Descriptions (if any): --

1. The installation procedure of the Android Studio/Java software must be demonstrated and carried out in groups.
2. Students should use the latest version of Android Studio/Java/ Kotlin to execute these programs. Diagrams given are for representational purposes only, students are expected to improvise on them.
3. Part B programs should be developed as an application and are to be demonstrated as a mini project in a group by adding extra features or the students can also develop their application and demonstrate it as a mini-project. (Projects/programs are not limited to the list given in Part B).

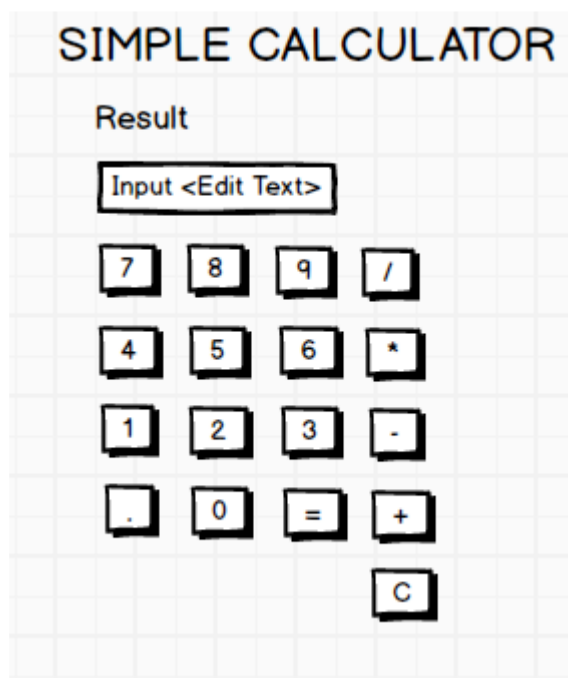
Programs List:

PART A

1. Create an application to design a Visiting Card. The Visiting card should have a company logo at the top right corner. The company name should be displayed in Capital letters, aligned to the center. Information like the name of the employee, job title, phone number, address, email, fax and the website address is to be displayed. Insert a horizontal line between the job title and the phone number.



2. Develop an Android application using controls like Button, TextView, EditText for designing a calculator having basic functionality like Addition, Subtraction, Multiplication, and Division.



3. Create a SIGN Up activity with Username and Password. Validation of password should happen based on the following rules:

- Password should contain uppercase and lowercase letters.
- Password should contain letters and numbers.
- Password should contain special characters.
- Minimum length of the password (the default value is 8).

On successful SIGN UP proceed to the next Login activity. Here the user should SIGN IN using the Username and Password created during signup activity. If the Username and Password are matched then navigate to the next activity which displays a message saying “Successful Login” or else display a toast message saying “Login Failed”. The user is given only two attempts and after that display a toast message saying “Failed Login Attempts” and disable the SIGN IN button. Use Bundle to transfer information from one activity to another.

LOGIN ACTIVITY

Username:

Password:

SIGN IN

SIGNUP ACTIVITY

Username:

Password:

SIGN UP

4. Develop an application to set an image as wallpaper. On click of a button, the wallpaper image should start to change randomly every 30 seconds.

CHANGING WALLPAPER APPLICATION

CLICK HERE TO CHANGE WALLPAPER

5. Write a program to create an activity with two buttons START and STOP. On Pressing of the START button, the activity must start the counter by displaying the numbers from One and the counter must keep on counting until the STOP button is pressed. Display the counter value in a TextView control.

COUNTER APPLICATION

Counter Value

START

STOP

6. Create two files of XML and JSON type with values for City_Name, Latitude, Longitude, Temperature, and Humidity. Develop an application to create an activity with two buttons to parse the XML and JSON files which when clicked should display the data in their respective layouts side by side.

PARSING XML AND JSON DATA													
<div>Parse XML Data</div> <div>Parse JSON Data</div>	<table border="1"> <thead> <tr> <th>XML DATA</th> <th>JSON Data</th> </tr> </thead> <tbody> <tr> <td>City_Name: Mysore</td> <td>City_Name: Mysore</td> </tr> <tr> <td>Latitude: 12.295</td> <td>Latitude: 12.295</td> </tr> <tr> <td>Longitude: 76.639</td> <td>Longitude: 76.639</td> </tr> <tr> <td>Temperature: 22</td> <td>Temperature: 22</td> </tr> <tr> <td>Humidity: 90%</td> <td>Humidity: 90%</td> </tr> </tbody> </table>	XML DATA	JSON Data	City_Name: Mysore	City_Name: Mysore	Latitude: 12.295	Latitude: 12.295	Longitude: 76.639	Longitude: 76.639	Temperature: 22	Temperature: 22	Humidity: 90%	Humidity: 90%
XML DATA	JSON Data												
City_Name: Mysore	City_Name: Mysore												
Latitude: 12.295	Latitude: 12.295												
Longitude: 76.639	Longitude: 76.639												
Temperature: 22	Temperature: 22												
Humidity: 90%	Humidity: 90%												

7. Develop a simple application with one Edit Text so that the user can write some text in it. Create a button called “Convert Text to Speech” that converts the user input text into voice.

TEXT TO SPEECH APPLICATION

Convert Text to Speech

8. Create an activity like a phone dialer withCALL and SAVE buttons. On pressing the CALL button, it must call the phone number and on pressing the SAVE button it must save the number to the phone contacts.

CALL AND SAVE APPLICATION

1234567890

DEL

1

2

3

4

5

6

7

8

9

*

0

#

CALL

SAVE

PART –B (MINI-PROJECT)

1. Write a program to enter Medicine Name, Date and Time of the Day as input from the user and store it in the SQLite database. Input for Time of the Day should be either Morning or Afternoon or Evening or Night. Trigger an alarm based on the Date and Time of the Day and display the Medicine Name.

MEDICINE DATABASE

Medicine Name:

Date:

Time of the Day:

2. Develop a content provider application with an activity called “Meeting Schedule” which takes Date, Time and Meeting Agenda as input from the user and store this information into the SQLite database. Create another application with an activity called “Meeting Info” having DatePicker control, which on the selection of a date should display the Meeting Agenda information for that particular date, else it should display a toast message saying “No Meeting on this Date”.


MEETING SCHEDULE

Date:

Time:

Meeting Agenda:

MEETING INFO

Pick a date to get meeting info: 

Mon, Jul 23

JULY 2018						
S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

[CANCEL](#) [OK](#)

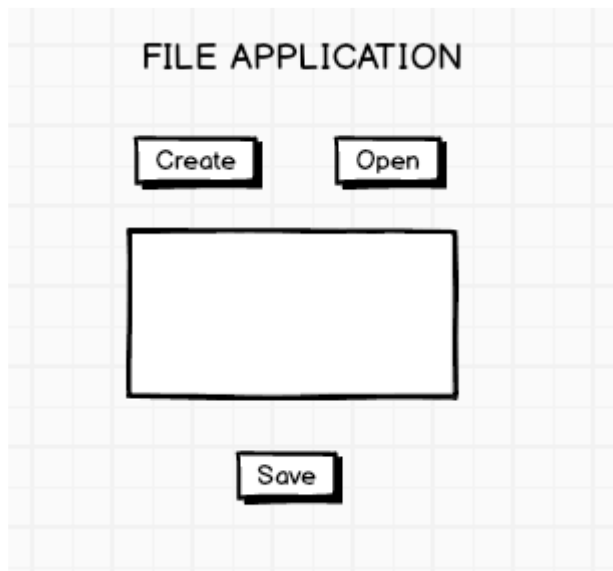
3. Create an application to receive an incoming SMS which is notified to the user. On clicking this SMS notification, the message content and the number should be displayed on the screen. Use appropriate emulator control to send the SMS message to your application.

SMS APPLICATION

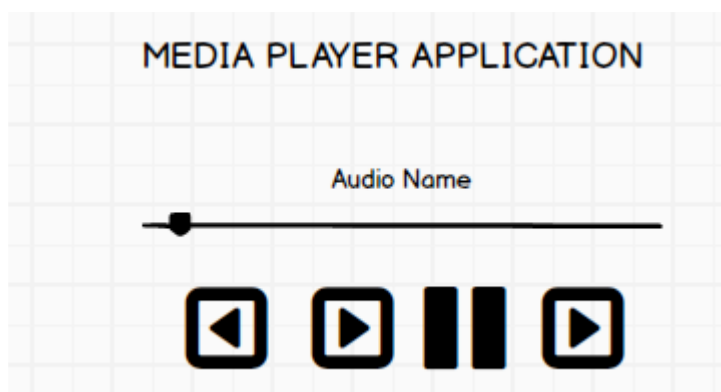
Display SMS Number

Display SMS Message

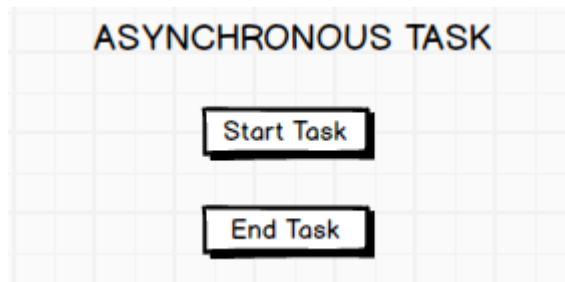
4. Write a program to create an activity having a Text box, and also Save, Open and Create buttons. The user has to write some text in the Text box. On pressing the Create button the text should be saved as a text file in MkSDcard. On subsequent changes to the text, the Save button should be pressed to store the latest content to the same file. On pressing the Open button, it should display the contents from the previously stored files in the Text box. If the user tries to save the contents in the Textbox to a file without creating it, then a toast message has to be displayed saying “First Create a File”.



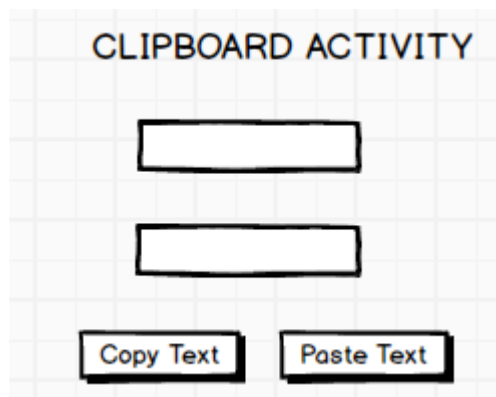
5. Create an application to demonstrate a basic media player that allows the user to Forward, Backward, Play and Pause an audio. Also, make use of the indicator in the seek bar to move the audio forward or backward as required.



6. Develop an application to demonstrate the use of Asynchronous tasks in android. The asynchronous task should implement the functionality of a simple moving banner. On pressing the Start Task button, the banner message should scroll from right to left. On pressing the Stop Task button, the banner message should stop. Let the banner message be “Demonstration of Asynchronous Task”.



7. Develop an application that makes use of the clipboard framework for copying and pasting of the text. The activity consists of two EditText controls and two Buttons to trigger the copy and paste functionality.



8. Create an AIDL service that calculates Car Loan EMI. The formula to calculate EMI is

$$E = P * (r(1+r)^n) / ((1+r)^n - 1)$$

where

E = The EMI payable on the car loan amount

P = The Car loan Principal Amount

r = The interest rate value computed on a monthly basis

n = The loan tenure in the form of months

The down payment amount has to be deducted from the principal amount paid towards buying the Car. Develop an application that makes use of this AIDL service to calculate the EMI. This application should have four EditText to read the Principal Amount, Down Payment, Interest Rate, Loan Term (in months) and a button named as “Calculate Monthly EMI”. On click of this button, the result should be shown in a TextView. Also, calculate the EMI by varying the Loan Term and Interest Rate values.

CAR EMI CALCULATOR

Principal Amount:

Down Payment:

Interest Rate:

Loan Term (in months):

EMI: Result

Calculate Monthly EMI

Laboratory outcomes: After studying these laboratory programs, students will be able to

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Procedure to Conduct Practical Examination

- Experiment distribution
 - For laboratories having only one part: Students are allowed to pick one experiment from the lot with equal opportunity.
 - For laboratories having PART A and PART B: Students are allowed to pick one experiment from PART A and one experiment from PART B, with equal opportunity.
- Change of experiment is allowed only once and marks allotted for procedure to be made zero of the changed part only.
- Marks Distribution (*Courseed to change in accordance with university regulations*)
 - For laboratories having only one part – Procedure + Execution + Viva-Voce: $15+70+15=100$ Marks
 - For laboratories having PART A and PART B
 - Part A – Procedure + Execution + Viva = $6 + 28 + 6 = 40$ Marks
 - Part B – Procedure + Execution + Viva = $9 + 42 + 9 = 60$ Marks

Text Books:

1. Google Developer Training, "Android Developer Fundamentals Course – Concept Reference", Google Developer Training Team, 2017.
<https://www.gitbook.com/book/google-developer-training/android-developer-fundamentalscourse-concepts/details>
(Download pdf file from the above link)

Reference Books:

1. Erik Hellman, "Android Programming – Pushing the Limits", 1st Edition, Wiley India Pvt Ltd, 2014. ISBN-13: 978-8126547197
2. Dawn Griffiths and David Griffiths, "Head First Android Development", 1st Edition, O'Reilly SPD Publishers, 2015. ISBN-13: 978-9352131341
3. Bill Phillips, Chris Stewart and Kristin Marsicano, "Android Programming: The Big Nerd Ranch Guide", 3rd Edition, Big Nerd Ranch Guides, 2017. ISBN-13: 978-0134706054

CONTENT LIST

SL.NO.	EXPERIMENT NAME	PAGE NO.
1	Introduction	1
2	Program 1: Create an application to design a Visiting Card. The Visiting card should have a company logo at the top right corner. The company name should be displayed in Capital letters, aligned to the center. Information like the name of the employee, job title, phone number, address, email, fax and the website address is to be displayed. Insert a horizontal line between the job title and the phone number.	13
3	Program 2: Develop an Android application using controls like Button, TextView, EditText for designing a calculator having basic functionality like Addition, Subtraction, Multiplication, and Division. Draw a colour cube and spin it using OpenGL transformation matrices.	16
4	Program 3: Create a SIGN Up activity with Username and Password. Validation of password should happen based on the following rules: <ul style="list-style-type: none"> • Password should contain uppercase and lowercase letters. • Password should contain letters and numbers. • Password should contain special characters. • Minimum length of the password (the default value is 8). On successful SIGN UP proceed to the next Login activity. Here the user should SIGN IN using the Username and Password created during signup activity. If the Username and Password are matched, then navigate to the next activity which displays a message saying “Successful Login” or else display a toast message saying “Login Failed”. The user is given only two attempts and after that display a toast message saying “Failed Login Attempts” and disable the SIGN IN button. Use Bundle to transfer information from one activity to another.	32
5	Program 4: Develop an application to set an image as wallpaper. On click of a button, the wallpaper image should start to change randomly every 30 seconds.	39
6	Program 5: Write a program to create an activity with two buttons START and STOP. On Pressing of the START button, the activity must start the counter by displaying the numbers from One and the counter must keep on counting until the STOP button is pressed. Display the counter value in a TextView control.	43
7	Program 6: Create two files of XML and JSON type with values for City_Name, Latitude, Longitude, Temperature, and Humidity. Develop an application to create an activity with two buttons to parse the XML and JSON files which when clicked should display the data in their respective layouts side by side.	46
8	Program 7: Develop a simple application with one Edit Text so	50

	that the user can write some text in it. Create a button called “Convert Text to Speech” that converts the user input text into voice.	
9	Program 8: Create an activity like a phone dialer with CALL and SAVE buttons. On pressing the CALL button, it must call the phone number and on pressing the SAVE button it must save the number to the phone contacts.	53
10	PART B(MINI-PROJECT) Program 1: Write a program to enter Medicine Name, Date and Time of the Day as input from the user and store it in the SQLite database. Input for Time of the Day should be either Morning or Afternoon or Evening or Night. Trigger an alarm based on the Date and Time of the Day and display the Medicine Name.	58
11	Program 2: Develop a content provider application with an activity called “Meeting Schedule” which takes Date, Time and Meeting Agenda as input from the user and store this information into the SQLite database. Create another application with an activity called “Meeting Info” having DatePicker control, which on the selection of a date should display the Meeting Agenda information for that particular date, else it should display a toast message saying “No Meeting on this Date”.	63
12	Program 3: Create an application to receive an incoming SMS which is notified to the user. On clicking this SMS notification, the message content and the number should be displayed on the screen. Use appropriate emulator control to send the SMS message to your application.	65
13	Program 4: Write a program to create an activity having a Text box, and also Save, Open and Create buttons. The user has to write some text in the Text box. On pressing the Create button the text should be saved as a text file in Mkdirsdcard. On subsequent changes to the text, the Save button should be pressed to store the latest content to the same file. On pressing the Open button, it should display the contents from the previously stored files in the Text box. If the user tries to save the contents in the Textbox to a file without creating it, then a toast message has to be displayed saying “First Create a File”.	70
14	Program 5: Create an application to demonstrate a basic media player that allows the user to Forward, Backward, Play and Pause an audio. Also, make use of the indicator in the seek bar to move the audio forward or backward as required.	75
15	Program 6: Develop an application to demonstrate the use of Asynchronous tasks in android. The asynchronous task should implement the functionality of a simple moving banner. On pressing the Start Task button, the banner message should scroll from right to left. On pressing the Stop Task button, the banner message should stop. Let the banner message be “Demonstration of Asynchronous Task”.	79

16	Program 7: Develop an application that makes use of the clipboard framework for copying and pasting of the text. The activity consists of two EditText controls and two Buttons to trigger the copy and paste functionality.	86
17	<p>Program 8 : Create an AIDL service that calculates Car Loan EMI. The formula to calculate EMI is</p> $E = P * (r(1+r)^n)/((1+r)^n-1)$ <p>where</p> <p>E = The EMI payable on the car loan amount</p> <p>P = The Car loan Principal Amount</p> <p>r = The interest rate value computed on a monthly basis</p> <p>n = The loan tenure in the form of months</p> <p>The down payment amount has to be deducted from the principal amount paid towards buying the Car. Develop an application that makes use of this AIDL service to calculate the EMI. This application should have four EditText to read the Principal Amount, Down Payment, Interest Rate, Loan Term (in months) and a button named as “Calculate Monthly EMI”. On click of this button, the result should be shown in a TextView. Also, calculate the EMI by varying the Loan Term and Interest Rate values.</p>	88
18	Viva Questions and answers	94

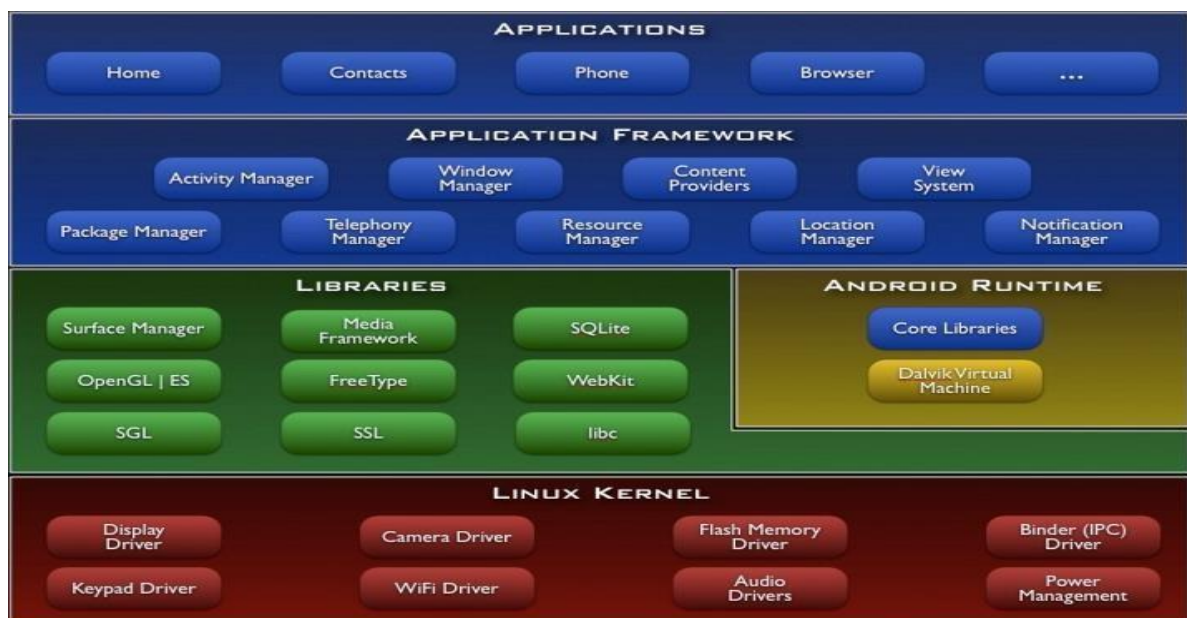
Android

Android is a mobile operating system based on a modified version of the Linux kernel and other open source software, designed primarily for touch screen mobile devices such as smart phones and tablets. Android is developed by a consortium of developers known as the Open Handset Alliance, with the main contributor and commercial marketer being Google.

Initially developed by Android Inc., which Google bought in 2005, Android was unveiled in 2007, with the first commercial Android device launched in September 2008. The current stable version is Android 10, released on September 3, 2019.

Android Architecture: -

Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram. GLUT gives you the ability to create a window, handle input and render to the screen without being Operating System dependent.



Linux kernel

At the bottom of the layers is Linux - Linux 2.6 with approximately 115 patches.

This provides basic system functionality like process management, memory management, device management like camera, keypad, display etc. Also, the kernel handles all the things that Linux is really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.

Libraries

On top of Linux kernel there is a set of libraries including open -source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc.

Android Runtime

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called Dalvik Virtual Machine which is a kind of Java Virtual Machine specially designed and optimized for Android.

The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine.

The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

Application Framework

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.

The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

Applications

You will find all the Android application at the top layer. You will write your application to be installed on this layer only. Examples of such applications are Contacts Books, Browser, and Games etc.

Android UI

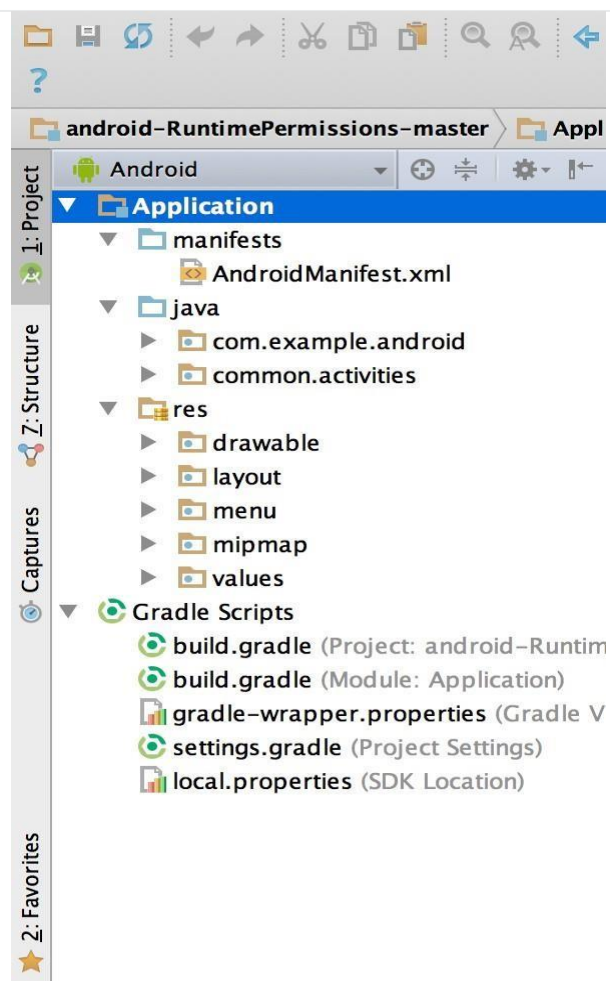
An Android application user interface is everything that the user can see and interact with.

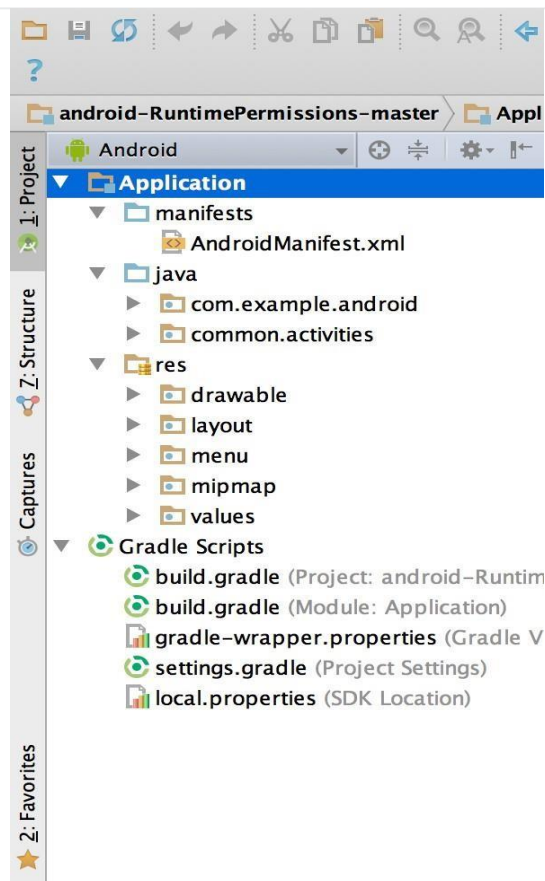
Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA. On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices
- Apply Changes to push code and resource changes to your running app without restarting your app
- Code templates and GitHub integration to help you build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to catch performance, usability, version compatibility, and other problems
- C++ and NDK support
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine

Project structure





Each project in Android Studio contains one or more modules with source code files and resource files. Types of modules include:

- Android app modules
- Library modules
- Google App Engine modules

By default, Android Studio displays your project files in the Android project view, as shown in figure 1. This view is organized by modules to provide quick access to your project's key source files.

All the build files are visible at the top level under Gradle Scripts and each app module contains the following folders:

- manifests: Contains the AndroidManifest.xml file.
- java: Contains the Java source code files, including JUnit test code.
- res: Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

The Android project structure on disk differs from this flattened representation. To see the actual file structure of the project, select Project from the Project dropdown (in figure 1, it's showing as Android).

You can also customize the view of the project files to focus on specific aspects of your app development. For example, selecting the Problems view of your project displays links to the

source files containing any recognized coding and syntax errors, such as a missing XML element closing tag in a layout file

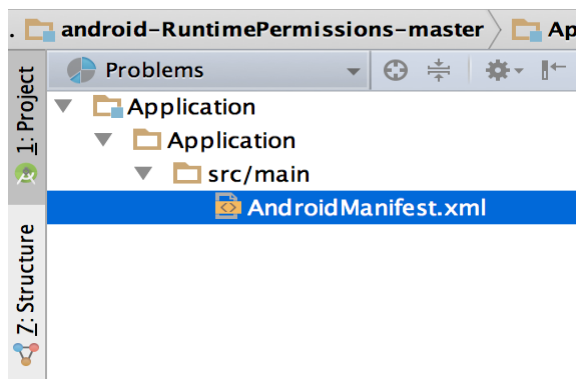


Figure 2. The project files in Problems view, showing a layout file with a problem.

The user interface

The Android Studio main window is made up of several logical areas identified in figure 3.

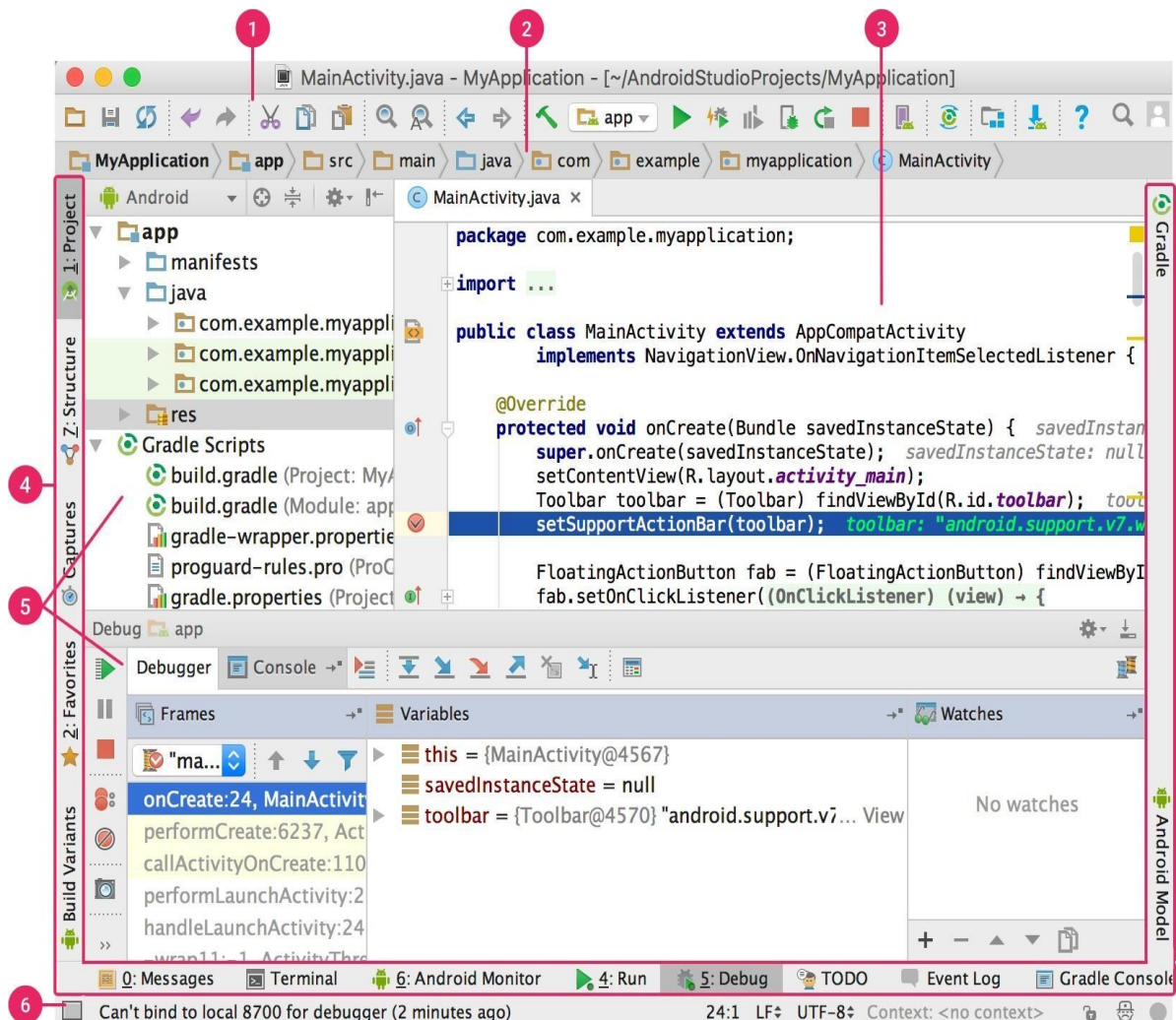


Figure 3. The Android Studio main window.

1. The toolbar lets you carry out a wide range of actions, including running your app and launching Android tools.
2. The navigation bar helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the Project window.
3. The editor window is where you create and modify code. Depending on the current file type, the editor can change. For example, when viewing a layout file, the editor displays the Layout Editor.
4. The tool window bar runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.
5. The tool windows give you access to specific tasks like project management, search, version control, and more. You can expand them and collapse them.
6. The status bar displays the status of your project and the IDE itself, as well as any warnings or messages.

You can organize the main window to give yourself more screen space by hiding or moving toolbars and tool windows. You can also use keyboard shortcuts to access most IDE features.

At any time, you can search across your source code, databases, actions, elements of the user interface, and so on, by double-pressing the Shift key, or clicking the magnifying glass in the upper right-hand corner of the Android Studio window. This can be very useful if, for example, you are trying to locate a particular IDE action that you have forgotten how to trigger.

Tool windows

Instead of using preset perspectives, Android Studio follows your context and automatically brings up relevant tool windows as you work. By default, the most commonly used tool windows are pinned to the tool window bar at the edges of the application window.

- To expand or collapse a tool window, click the tool's name in the tool window bar. You can also drag, pin, unpin, attach, and detach tool windows.
- To return to the current default tool window layout, click Window > Restore Default Layout or customize your default layout by clicking Window > Store Current Layout as Default.
- To show or hide the entire tool window bar, click the window icon in the bottom left-hand corner of the Android Studio window.
- To locate a specific tool window, hover over the window icon and select the tool window from the menu.

You can also use keyboard shortcuts to open tool windows. Table 1 lists the shortcuts for the most common windows.

Table 1. Keyboard shortcuts for some useful tool windows.

Tool window	Windows and Linux	Mac
Project	Alt+1	Command+1
Version Control	Alt+9	Command+9
Run	Shift+F10	Control+R
Debug	Shift+F9	Control+D
Logcat	Alt+6	Command+6
Return to Editor	Esc	Esc
Hide All Tool Windows	Control+Shift+F12	Command+Shift+F12

If you want to hide all toolbars, tool windows, and editor tabs, click View > **Enter Distraction Free Mode**. This enables Distraction Free Mode. To exit Distraction Free Mode, click View > **Exit Distraction Free Mode**.

You can use Speed Search to search and filter within most tool windows in Android Studio. To use Speed Search, select the tool window and then type your search query.

Code completion

Android Studio has three types of code completion, which you can access using keyboard shortcuts.

Table 2. Keyboard shortcuts for code completion.

Type	Description	Windows and Linux	Mac
Basic Completion	Displays basic suggestions for variables, types methods expressions, and so on. If you call basic completion twice in a row, you see more results, including private members and	Control+Space	Control+Space
Smart Completion	Displays relevant options based on the context. Smart completion is aware of the expected type and data flows. If you call Smart Completion twice in a row, you see more results,	Control+Shift+Space	Control+Shift+Space
Statement Completion	Completes the current for you, adding missing parentheses, brackets, braces,	Control+Shift+Enter	Shift+Command+Enter

You can also perform quick fixes and show intention actions by pressing Alt+Enter.

Android Studio Installation

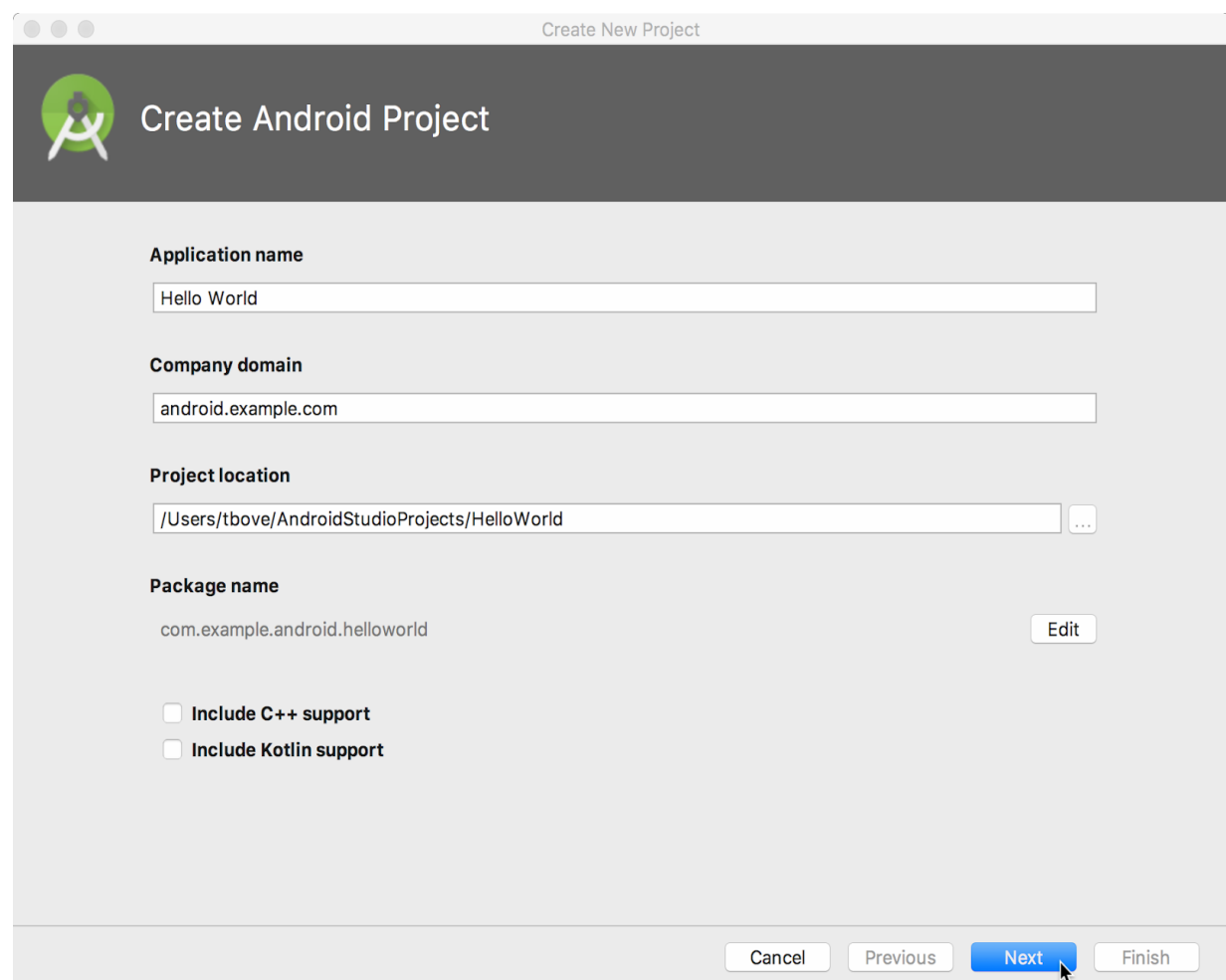
To install Android Studio on Windows, proceed as follows:

1. If you downloaded an .exe file (recommended), double-click to launch it.
If you downloaded a .zip file, unpack the ZIP, copy the android-studio folder into your Program Files folder, and then open the android-studio > bin folder and launch studio64.exe (for 64-bit machines) or studio.exe (for 32-bit machines).
2. Follow the setup wizard in Android Studio and install any SDK packages that it recommends.

Create “Hello World” application

After you successfully install Android Studio, you will create, from a template, a new project for the Hello World app. This simple app displays the string "Hello World" on the screen of the Android virtual or physical device.

1. Studio window, click Start a new Android Studio project.
2. In the Create Android Project window, enter Hello World for the Application name.



The screenshot shows the 'Create New Project' window in Android Studio. The window has a title bar with three window control buttons and the text 'Create New Project'. Below the title bar is a dark header with the Android logo and the text 'Create Android Project'. The main area contains several form fields and checkboxes:

- Application name:** A text field containing 'Hello World'.
- Company domain:** A text field containing 'android.example.com'.
- Project location:** A text field containing '/Users/tbove/AndroidStudioProjects/HelloWorld' with a browse button (three dots) to its right.
- Package name:** A text field containing 'com.example.android.helloworld' with an 'Edit' button to its right.
- Include C++ support:** An unchecked checkbox.
- Include Kotlin support:** An unchecked checkbox.

At the bottom of the window are four buttons: 'Cancel', 'Previous', 'Next' (highlighted with a mouse cursor), and 'Finish'.

3. Verify that the default Project location is where you want to store your Hello World app and other Android Studio projects, or change it to your preferred directory.
4. Accept the default android.example.com for Company Domain, or create a unique company domain.

If you are not planning to publish your app, you can accept the default. Be aware that changing the package name of your app later is extra work.

5. Leave unchecked the options to Include C++ support and Include Kotlin support, and click Next.
6. On the Target Android Devices screen, Phone and Tablet should be selected. Ensure that API 15: Android 4.0.3 IceCreamSandwich is set as the Minimum SDK; if it is not, use the popup menu to set it.

The screenshot shows the 'Target Android Devices' dialog in Android Studio. The title bar says 'Create New Project'. The header has the Android logo and 'Target Android Devices'. The main section is titled 'Select the form factors and minimum SDK' with a subtitle: 'Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.'

Under 'Phone and Tablet' (checked), the 'Minimum SDK' is set to 'API 15: Android 4.0.3 (IceCreamSandwich)'. Below this, it says 'By targeting API 15 and later, your app will run on approximately 100% of devices.' with a link 'Help me choose'. There is an unchecked checkbox for 'Include Android Instant App support'.

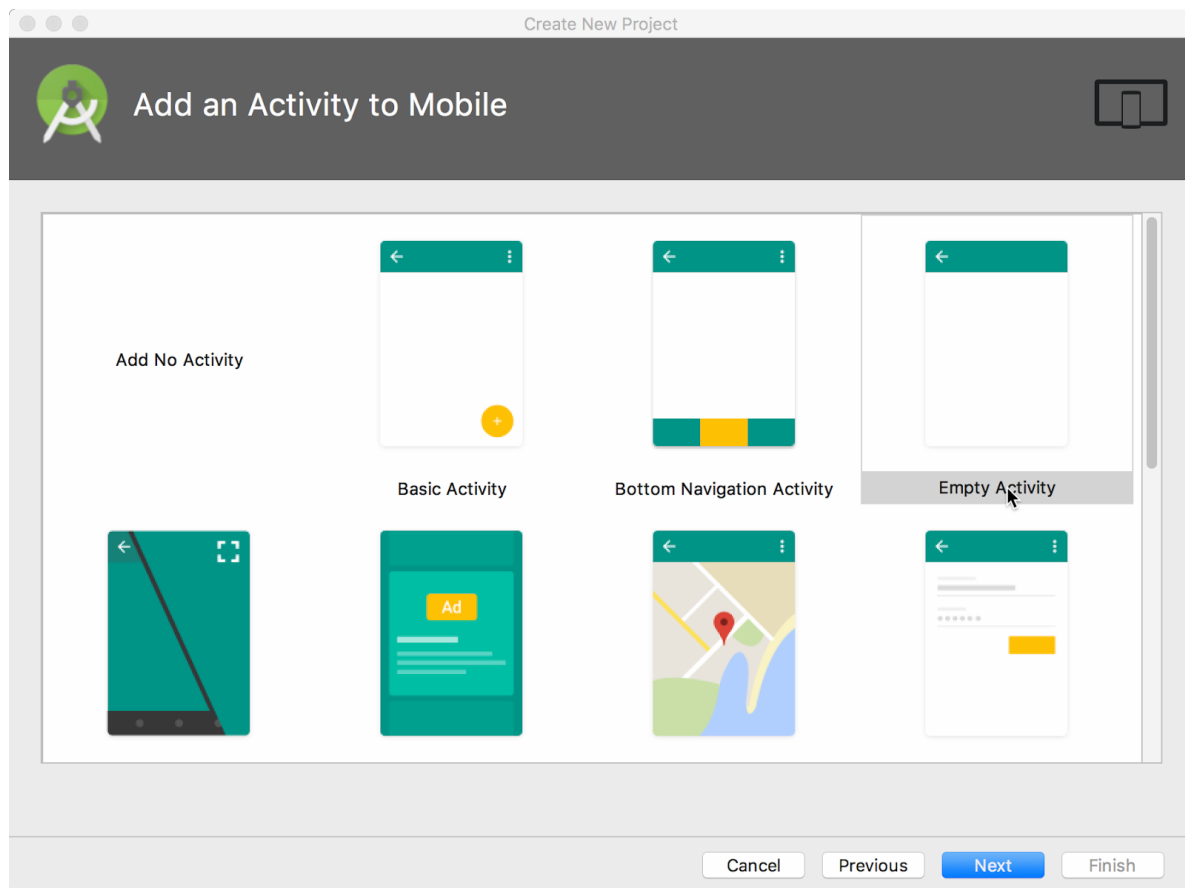
Other form factors are unchecked: 'Wear' (API 21: Android 5.0 (Lollipop)), 'TV' (API 21: Android 5.0 (Lollipop)), 'Android Auto', and 'Android Things' (API 24: Android 7.0 (Nougat)).

At the bottom, there are four buttons: 'Cancel', 'Previous', 'Next' (highlighted with a mouse cursor), and 'Finish'.

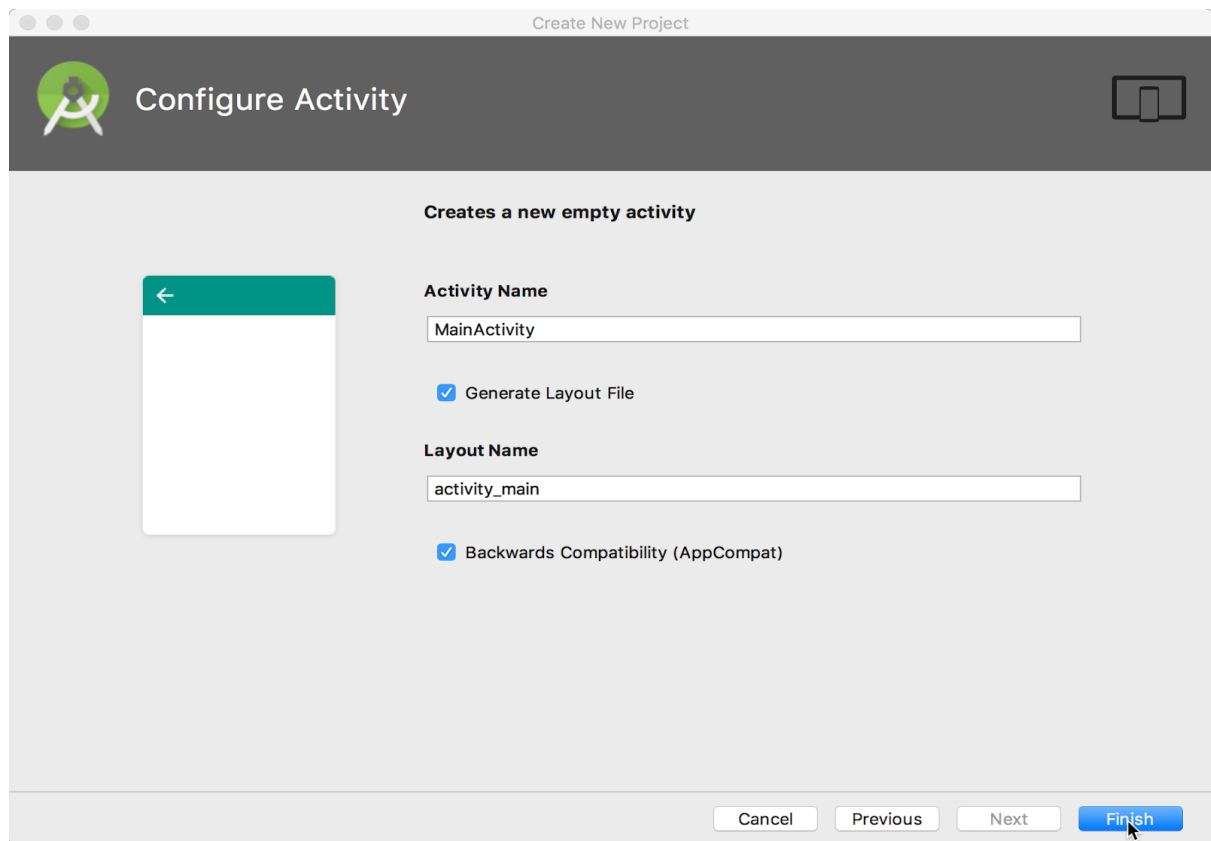
These are the settings used by the examples in the lessons for this course. As of this writing, these settings make your Hello World app compatible with 97% of Android devices active on the Google Play Store.

7. Leave unchecked the Include Instant App support and all other options. Then click next. If your project requires additional components for your chosen target SDK, Android Studio will install them automatically.

8. The Add an Activity window appears. An Activity is a single, focused thing that the user can do. It is a crucial component of any Android app. An Activity typically has a layout associated with it that defines how UI elements appear on a screen. Android Studio provides Activity templates to help you get started. For the Hello World project, choose Empty Activity as shown below, and click next.



9. The Configure Activity screen appears (which differs depending on which template you chose in the previous step). By default, the empty Activity provided by the template is named MainActivity. You can change this if you want, but this lesson uses MainActivity.
10. Make sure that the Generate Layout file option is checked. The layout name by default is activity_main. You can change this if you want, but this lesson uses activity_main.
11. Make sure that the Backwards Compatibility (App Compat) option is checked. This ensures that your app will be backwards-compatible with previous versions of Android.
12. Click Finish.



PART A

Program 1: Create an application to design a Visiting Card. The visiting card should have a company logo at the top right corner. The company name should be displayed in Capital letters, aligned to the center. Information like the name of the employee, job title, phone number, address, email, fax and the website address is to be displayed. Insert a horizontal line between the job title and the phone number.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.visitingcard">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.VisitingCard">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action
                    android:name="android.intent.action.MAIN" />

                <category
                    android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="217dp"
        android:layout_height="37dp"
        android:layout_marginStart="40dp"
        android:layout_marginLeft="40dp"
        android:layout_marginTop="24dp"
        android:text="COMPANY NAME"
        app:layout_constraintBottom_toTopOf="@+id/divider"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.121" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="76dp"
        android:text="Selva kumar S"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/divider" />

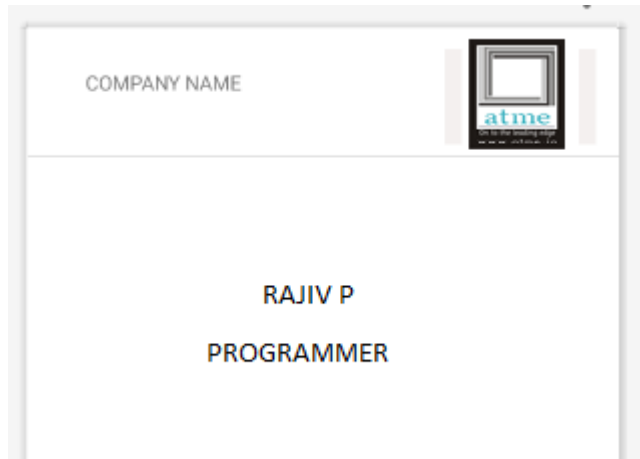
    <View
        android:id="@+id/divider"
        android:layout_width="match_parent"
        android:layout_height="1dp"
        android:layout_marginTop="88dp"
        android:background="?android:attr/listDivider"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="105dp"
        android:layout_height="66dp"
        android:layout_marginStart="32dp"
        android:layout_marginLeft="32dp"
        android:layout_marginBottom="8dp"
        android:background="#F4F0EF"
        app:layout_constraintBottom_toTopOf="@+id/divider"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.058"
        app:layout_constraintStart_toEndOf="@+id/textView2"
        app:srcCompat="@drawable/bmslogo" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
android:layout_marginTop="24dp"
android:text="Assistant Professor"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/textView" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

OUTPUT:**Program Outcome:**

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 2: Develop an Android application using controls like Button, TextView, EditText for designing a calculator having basic functionality like Addition, Subtraction, Multiplication and Division.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

XML File:

```
<?xml version="1.0" encoding="utf-8"?>

<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    tools:context="com.example.a84.calculator.MainActivity">

    <RelativeLayout

        android:layout_width="368dp"

        android:layout_height="495dp"

        android:layout_marginBottom="8dp"

        android:layout_marginEnd="8dp"

        android:layout_marginTop="8dp"

        app:layout_constraintBottom_toBottomOf="parent"

        app:layout_constraintEnd_toEndOf="parent"

        app:layout_constraintTop_toTopOf="parent">

        <Button
```

```
android:id="@+id/btn_1"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_alignParentLeft="true"

android:layout_alignParentStart="true"

android:layout_below="@+id/edText1"

android:layout_marginTop="60dp"

android:onClick="PressOne"

android:text="1"

android:textSize="18sp" />
```

<Button

```
android:id="@+id/btn_0"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_below="@+id/btn_8"

android:layout_toEndOf="@+id/btn_7"

android:layout_toRightOf="@+id/btn_7"

android:text="0"

android:textSize="18sp" />
```

<Button

```
android:id="@+id/btn_9"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_below="@+id/btn_6"

android:layout_toEndOf="@+id/btn_5"

android:layout_toRightOf="@+id/btn_5"
```

```
android:text="9"

android:textSize="18sp" />
```

```
<Button

    android:id="@+id/btn_8"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content"

    android:layout_below="@+id/btn_5"

    android:layout_toEndOf="@+id/btn_7"

    android:layout_toRightOf="@+id/btn_7"

    android:text="8"

    android:textSize="18sp" />
```

```
<Button

    android:id="@+id/btn_7"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content"

    android:layout_alignLeft="@+id/btn_4"

    android:layout_alignStart="@+id/btn_4"

    android:layout_below="@+id/btn_4"

    android:text="7"

    android:textSize="18sp" />
```

```
<Button

    android:id="@+id/btn_6"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content"

    android:layout_alignBaseline="@+id/btn_5"
```

```
android:layout_alignBottom="@+id/btn_5"

android:layout_toEndOf="@+id/btn_5"

android:layout_toRightOf="@+id/btn_5"

android:text="6"

android:textSize="18sp" />
```

```
<Button
```

```
    android:id="@+id/btn_5"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content"

    android:layout_below="@+id/btn_2"

    android:layout_toEndOf="@+id/btn_4"

    android:layout_toRightOf="@+id/btn_4"

    android:text="5"

    android:textSize="18sp" />
```

```
<Button
```

```
    android:id="@+id/btn_4"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content"

    android:layout_alignLeft="@+id/btn_1"

    android:layout_alignStart="@+id/btn_1"

    android:layout_below="@+id/btn_1"

    android:text="4"

    android:textSize="18sp" />
```

```
<Button
```

```
    android:id="@+id/btn_3"
```

```
android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_alignBaseline="@+id/btn_2"

android:layout_alignBottom="@+id/btn_2"

android:layout_toEndOf="@+id/btn_2"

android:layout_toRightOf="@+id/btn_2"

android:text="3"

android:textSize="18sp" />
```

<Button

```
android:id="@+id/btn_2"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_alignBaseline="@+id/btn_1"

android:layout_alignBottom="@+id/btn_1"

android:layout_toEndOf="@+id/btn_1"

android:layout_toRightOf="@+id/btn_1"

android:text="2"

android:textSize="18sp" />
```

<Button

```
android:id="@+id/btn_Add"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_above="@+id/btn_6"

android:layout_alignParentEnd="true"

android:layout_alignParentRight="true"

android:backgroundTint="@android:color/darker_gray"
```

```
android:text="+"  
  
android:textColor="@android:color/background_light"  
  
android:textSize="18sp" />
```

```
<Button  
  
    android:id="@+id/btn_Sub"  
  
    android:layout_width="wrap_content"  
  
    android:layout_height="wrap_content"  
  
    android:layout_alignLeft="@+id/btn_Add"  
  
    android:layout_alignStart="@+id/btn_Add"  
  
    android:layout_below="@+id/btn_Add"  
  
    android:backgroundTint="@android:color/darker_gray"  
  
    android:text="-"  
  
    android:textColor="@android:color/background_light"  
  
    android:textSize="18sp" />
```

```
<Button  
  
    android:id="@+id/btn_Mul"  
  
    android:layout_width="wrap_content"  
  
    android:layout_height="wrap_content"  
  
    android:layout_alignLeft="@+id/btn_Sub"  
  
    android:layout_alignStart="@+id/btn_Sub"  
  
    android:layout_below="@+id/btn_6"  
  
    android:backgroundTint="@android:color/darker_gray"  
  
    android:text="*"  
  
    android:textColor="@android:color/background_light"  
  
    android:textSize="18sp" />
```

<Button

```
    android:id="@+id/btn_Div"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content"

    android:layout_alignLeft="@+id/btn_Mul"

    android:layout_alignStart="@+id/btn_Mul"

    android:layout_below="@+id/btn_9"

    android:backgroundTint="@android:color/darker_gray"

    android:text="/"

    android:textColor="@android:color/background_light"

    android:textSize="18sp" />
```

<EditText

```
    android:id="@+id/edText1"

    android:layout_width="wrap_content"

    android:layout_height="wrap_content"

    android:layout_alignParentEnd="true"

    android:layout_alignParentLeft="true"

    android:layout_alignParentRight="true"

    android:layout_alignParentStart="true"

    android:layout_alignParentTop="true"

    android:layout_marginTop="22dp"

    android:ems="10"

    android:inputType="textPersonName"

    android:textAlignment="textEnd"

    android:textSize="24sp" />
```

<Button

```
android:id="@+id/btn_calc"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_below="@+id/btn_0"

android:layout_toEndOf="@+id/btn_0"

android:layout_toRightOf="@+id/btn_0"

android:backgroundTint="@android:color/holo_green_light"

android:text=""

android:textColor="@android:color/background_light"

android:textSize="18sp" />
```

<Button

```
android:id="@+id/btn_dec"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_below="@+id/btn_7"

android:layout_toLeftOf="@+id/btn_8"

android:layout_toStartOf="@+id/btn_8"

android:text="."

android:textSize="18sp" />
```

<Button

```
android:id="@+id/btn_clear"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_alignParentEnd="true"

android:layout_alignParentRight="true"

android:layout_below="@+id/btn_Div"
```



```
        android:backgroundTint="@android:color/holo_blue_dark"

        android:text="clear"

        android:textColor="@android:color/background_light"

        android:textSize="18sp" />
```

```
</RelativeLayout>
```

```
</android.support.constraint.ConstraintLayout>
```

Java File:

```
package com.example.a84.calculator;
```

```
import android.support.v7.app.AppCompatActivity;
```

```
import android.os.Bundle;
```

```
import android.view.View;
```

```
import android.widget.Button;
```

```
import android.widget.EditText;
```

```
public class MainActivity extends AppCompatActivity {
```

```
    Button
```

```
    btn_1,btn_2,btn_3,btn_4,btn_5,btn_6,btn_7,btn_8,btn_9,btn_0,btn_Add,btn_Sub,btn_Mul,btn_Div,btn_calc,btn_dec,btn_clear;
```

```
    EditText ed1;
```

```
    float Value1, Value2;
```

```
    boolean mAddition, mSubtract, mMultiplication, mDivision ;
```

```
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.activity_main);
```

```
btn_0 = (Button) findViewById(R.id.btn_0);
btn_1 = (Button) findViewById(R.id.btn_1);
btn_2 = (Button) findViewById(R.id.btn_2);
btn_3 = (Button) findViewById(R.id.btn_3);
btn_4 = (Button) findViewById(R.id.btn_4);
btn_5 = (Button) findViewById(R.id.btn_5);
btn_6 = (Button) findViewById(R.id.btn_6);
btn_7 = (Button) findViewById(R.id.btn_7);
btn_8 = (Button) findViewById(R.id.btn_8);
btn_9 = (Button) findViewById(R.id.btn_9);
btn_Add = (Button) findViewById(R.id.btn_Add);
btn_Div = (Button) findViewById(R.id.btn_Div);
btn_Sub = (Button) findViewById(R.id.btn_Sub);
btn_Mul = (Button) findViewById(R.id.btn_Mul);
btn_calc = (Button) findViewById(R.id.btn_calc);
btn_dec = (Button) findViewById(R.id.btn_dec);
btn_clear = (Button) findViewById(R.id.btn_clear);
ed1 = (EditText) findViewById(R.id.edText1);

btn_0.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        ed1.setText(ed1.getText()+"0");
    }
});

btn_1.setOnClickListener(new View.OnClickListener() {
```

```
@Override

public void onClick(View v) {

    ed1.setText(ed1.getText()+"1");

}

});

btn_2.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        ed1.setText(ed1.getText()+"2");

    }

});

btn_3.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        ed1.setText(ed1.getText()+"3");

    }

});

btn_4.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        ed1.setText(ed1.getText()+"4");

    }

});

btn_5.setOnClickListener(new View.OnClickListener() {
```

```
@Override

public void onClick(View v) {

    ed1.setText(ed1.getText()+"5");

}

});

btn_6.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        ed1.setText(ed1.getText()+"6");

    }

});

btn_7.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        ed1.setText(ed1.getText()+"7");

    }

});

btn_8.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        ed1.setText(ed1.getText()+"8");

    }

});

btn_9.setOnClickListener(new View.OnClickListener() {
```

```
@Override

public void onClick(View v) {

    ed1.setText(ed1.getText()+"9");

}

});

btn_dec.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        ed1.setText(ed1.getText()+".");

    }

});

btn_Add.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        if (ed1 == null){

            ed1.setText("");

        }else {

            Value1 = Float.parseFloat(ed1.getText() + "");

            mAddition = true;

            ed1.setText(null);

        }

    }

});

btn_Sub.setOnClickListener(new View.OnClickListener() {
```

```
@Override

public void onClick(View v) {

    Value1 = Float.parseFloat(ed1.getText() + "");

    mSubtract = true ;

    ed1.setText(null);

}

});

btn_Mul.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        Value1 = Float.parseFloat(ed1.getText() + "");

        mMultiplication = true ;

        ed1.setText(null);

    }

});

btn_Div.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        Value1 = Float.parseFloat(ed1.getText()+ "");

        mDivision = true ;

        ed1.setText(null);

    }

});

btn_calc.setOnClickListener(new View.OnClickListener() {

    @Override
```

```
public void onClick(View v) {  
  
    Value2 = Float.parseFloat(ed1.getText() + "");  
  
  
    if (mAddition == true){  
  
        ed1.setText(Value1 + Value2 + "");  
  
        mAddition=false;  
  
    }  
  
  
    if (mSubtract == true){  
  
        ed1.setText(Value1 - Value2 + "");  
  
        mSubtract=false;  
  
    }  
  
  
    if (mMultiplication == true){  
  
        ed1.setText(Value1 * Value2 + "");  
  
        mMultiplication=false;  
  
    }  
  
  
    if (mDivision == true){  
  
        ed1.setText(Value1 / Value2+ "");  
  
        mDivision=false;  
  
    }  
  
}  
  
});  
  
  
btn_clear.setOnClickListener(new View.OnClickListener() {  
  
    @Override  
  
    public void onClick(View v) {
```

```
        ed1.setText("");  
    }  
    });  
}  
  
}
```

OUTPUT:



Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 3: Create a SIGN up activity with Username and Password. Validation of password should happen based on the following rules:

- Password should contain uppercase and lowercase letters.
- Password should contain letters and numbers.
- Password should contain special characters.
- Minimum length of the password (the default value is 8)

On successful SIGN UP proceed to the next Login activity, Here the user should SIGN IN using the Username and Password created during signup activity. If the Username and Password are matched then navigate to the next activity which displays a message saying "Sucessful Login" or else display a toast message saying "Login Failed". The user is given only two attempts and after that display a toast message saying "Failed Login Attempts" and disable the SIGN IN button. Use Bundle to transfer information from one activity to another.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.example3_a">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Example3A">
        <activity android:name=".ThirdActivity"></activity>
        <activity android:name=".SecondActivity" />
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

MainActivity:

```
package com.example.example3_a;

import android.content.Intent;
import androidx.appcompat.app.AppCompatActivity;
```

```

import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.Toast;

import com.google.android.material.textfield.TextInputLayout;

import java.util.regex.Pattern;

public class MainActivity extends AppCompatActivity {

    //Defining the Views
    EditText e1,e2;
    Button bt;

    String name1, name2;

    // defining our own password pattern
    private static final Pattern PASSWORD_PATTERN =
        Pattern.compile("^" +
            "(?=.*[@#$%^&+=])" +    // at least 1 special
character
            "(?=.*\\S+$)" +          // no white spaces
            "(?=.*[A-Z])(?=.*[a-z]).*" + //upper case and lower
case letter
            ".{8,}" +                // at least 8 characters
            "$");
    /*
character
        Pattern.compile("^" +
            "(?=.*[@#$%^&+=])" +    // at least 1 special
            "(?=.*\\S+$)" +          // no white spaces
            ".{8,}" +                // at least 8 characters
            "$");*/
    private TextInputLayout editText2; //password

    @Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    //Referring the Views
    e1= (EditText) findViewById(R.id.editText);
    e2= (EditText) findViewById(R.id.editText2);

    bt= (Button) findViewById(R.id.button);

    //Creating Listener for Button
    bt.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {

    //Getting the Values from Views(Edittext & Spinner)
    name1=e1.getText().toString();
    name2=e2.getText().toString();
    //dept=s.getSelectedItem().toString();
    if(validatePassword()) {
        //Intent For Navigating to Second Activity
        Intent i = new Intent(MainActivity.this, SecondActivity.class);

```



```

        pwd= (EditText) findViewById(R.id.edtPassword);
        b1 = (Button) findViewById(R.id.btn_signin);
        //Getting the Intent
        Intent i = getIntent();
        //Getting the Values from First Activity using the Intent received
        name=i.getStringExtra("name_key");
        pass=i.getStringExtra("reg_key");
        Toast.makeText(getApplicationContext(),name + " "
+pass,Toast.LENGTH_LONG).show();
    }

    public void validate(View view) {
        String usn = un.getText().toString();
        String pswd = pwd.getText().toString();
        if( usn.equals(name) && pswd.equals(pass)) {
            Intent i = new Intent(SecondActivity.this,
ThirdActivity.class);
            startActivity(i);
        }
        else
        {
            Toast.makeText(getApplicationContext(),"Login
Failed",Toast.LENGTH_SHORT).show();
        }
    }
}

```

1. Wallpaper

Manifest.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.wallpaperchanger">
    <uses-permission android:name="android.permission.SET_WALLPAPER"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.WallPaperChanger">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>

```

XML:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Wall Paper Changer"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.064" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="104dp"
        android:text="Change Wallpaper"
        android:onClick="ChangeImage"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
        app:layout_constraintVertical_bias="0.042" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Activity.java:

```
package com.example.wallpaperchanger;

import androidx.appcompat.app.AppCompatActivity;

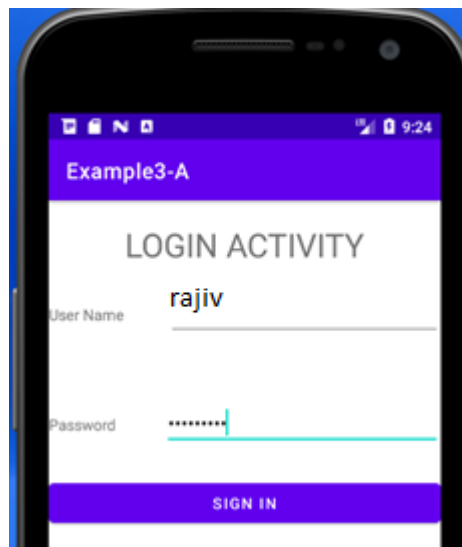
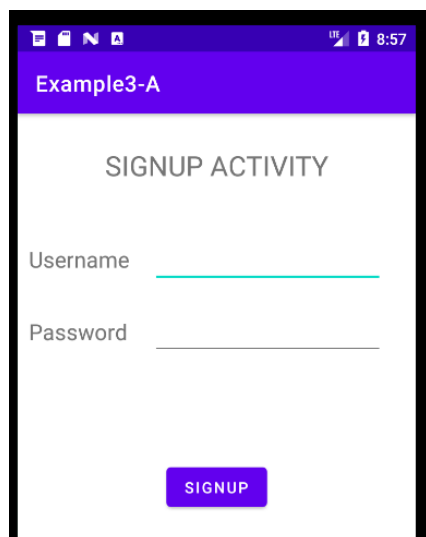
import android.app.WallpaperManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.widget.Toast;

import java.io.IOException;
import java.io.InputStream;
import java.util.Random;
```

```
public class MainActivity extends AppCompatActivity {
    int[] images;
    Handler handler;
    Runnable runnable;
    int delay = 30000;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        handler = new Handler();
    }
    @Override
    protected void onResume() {
        handler.postDelayed(runnable = new Runnable() {
            public void run() {
                handler.postDelayed(runnable, delay);
                SetWallPaper();
                //Toast.makeText(MainActivity.this, "This method is run every 30
seconds",
                                // Toast.LENGTH_SHORT).show();
            }
        }, delay);
        super.onResume();
    }
    @Override
    protected void onPause() {
        super.onPause();
        handler.removeCallbacks(runnable); //stop handler when activity not visible
        super.onPause();
    }

    public void ChangeImage(View view) {
        SetWallPaper();
    }
    private void SetWallPaper()
    {
        images = new int[] {R.drawable.a,R.drawable.b,R.drawable.c,R.drawable.d};
        int arylength = images.length;
        Random random = new Random();
        int rnum = random.nextInt(arylenght);
        Bitmap bitmap = BitmapFactory.decodeResource(getResources(),images[rnum]);
        WallpaperManager manager =
        WallpaperManager.getInstance(getApplicationContext());
        try {
            manager.setBitmap(bitmap);
            Toast.makeText(this,"Wall Paper changed",Toast.LENGTH_SHORT).show();
        }
        catch(IOException e)
        {
            Toast.makeText(this,"Error",Toast.LENGTH_SHORT).show();
        }
    }
}
```

OUTPUT:



Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 4: Develop an application to set an image as wallpaper. On click of a button, the wallpaper image should start to change randomly every 30 seconds.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

Manifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.wallpaperchanger">
    <uses-permission android:name="android.permission.SET_WALLPAPER"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.WallPaperChanger">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

XML:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Wall Paper Changer"
```



```
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.064" />

<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="104dp"
    android:text="Change Wallpaper"
    android:onClick="ChangeImage"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.497"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView"
    app:layout_constraintVertical_bias="0.042" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Activity.java:

```
package com.example.wallpaperchanger;

import androidx.appcompat.app.AppCompatActivity;

import android.app.WallpaperManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.widget.Toast;

import java.io.IOException;
import java.io.InputStream;
import java.util.Random;

public class MainActivity extends AppCompatActivity {
    int[] images;
    Handler handler;
    Runnable runnable;
    int delay = 30000;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        handler = new Handler();
    }
    @Override
    protected void onResume() {
        handler.postDelayed(runnable = new Runnable() {
            public void run() {
                handler.postDelayed(runnable, delay);
            }
        }, delay);
    }
}
```

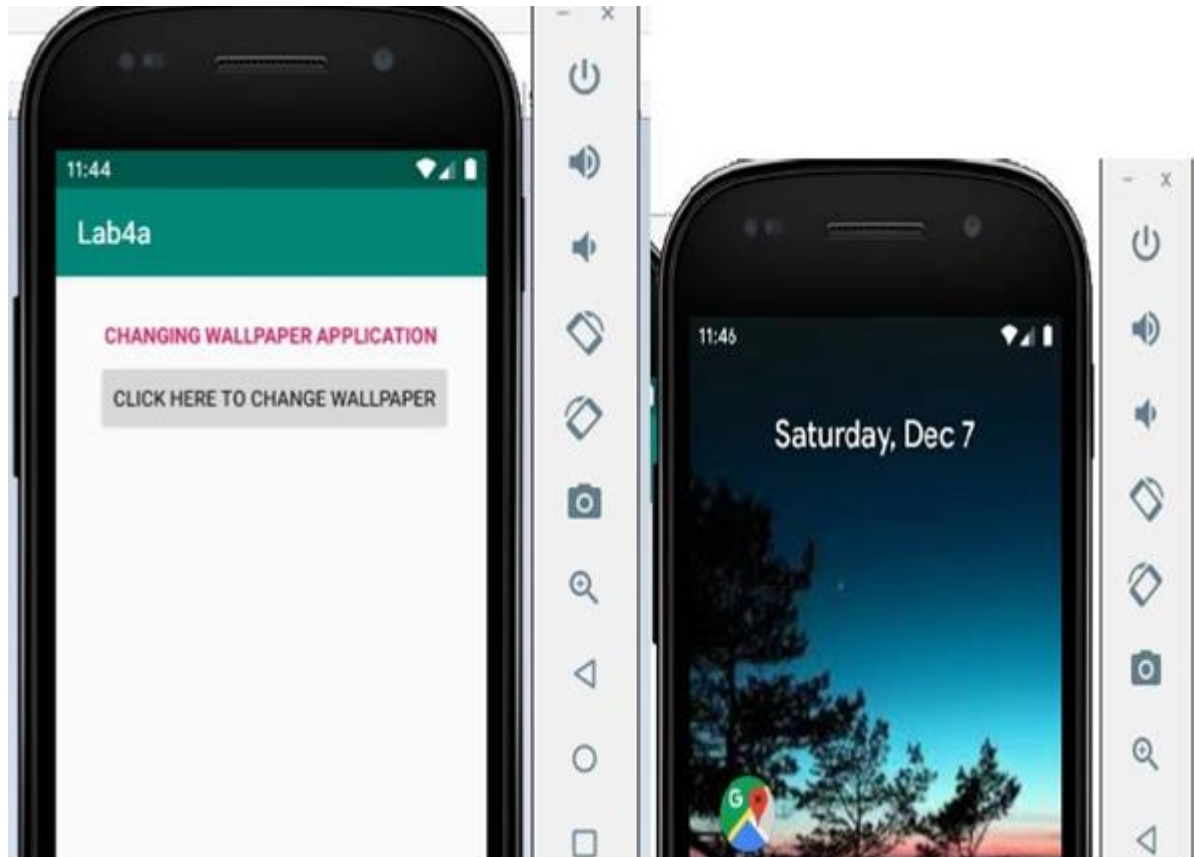
```
        SetWallPaper();
        //Toast.makeText(MainActivity.this, "This method is run every 30
seconds",
                                // Toast.LENGTH_SHORT).show();
    }
    }, delay);
    super.onResume();
}
@Override
protected void onPause() {
    super.onPause();
    handler.removeCallbacks(runnable); //stop handler when activity not
visible super.onPause();
}

public void ChangeImage(View view) {

    SetWallPaper();
}
private void SetWallPaper()
{
    images = new int[] {R.drawable.a,R.drawable.b,R.drawable.c,R.drawable.d};
    int arylength = images.length;
    Random random = new Random();
    int rnum = random.nextInt(arylenght);
    Bitmap bitmap = BitmapFactory.decodeResource(getResources(),images[rnum]);
    WallpaperManager manager =
WallpaperManager.getInstance(getApplicationContext());
    try {
        manager.setBitmap(bitmap);
        Toast.makeText(this,"Wall Paper changed",Toast.LENGTH_SHORT).show();

    }
    catch(IOException e)
    {
        Toast.makeText(this,"Error",Toast.LENGTH_SHORT).show();
    }
}
}
```

OUTPUT:



Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 5: Write a program to create an activity with two buttons START and STOP. On Pressing of the START button, the activity must start the counter by displaying the numbers from One and the counter must keep on counting until the STOP button is pressed. Display the counter value in a TextView control.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation ="vertical"
    android:layout_width="match_parent"
    android:layout_height="fill_parent"
    >
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello"
    />
    <Chronometer
        android:id="@+id/chronometer"
        android:layout_gravity="center_horizontal"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
    />
    <Button
        android:id="@+id/buttonstart"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Start"
    />
    <Button
        android:id="@+id/buttonstop"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Stop"
    />
    <Button android:id="@+id/buttonreset"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Reset"
    />
</LinearLayout>
```

MainActivity.java:

```
package com.example.program5_a;
import android.app.Activity;
import android.os.Bundle;
import android.os.SystemClock;
import android.view.View;
import android.widget.Button;
import android.widget.Chronometer;

public class MainActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        final Chronometer myChronometer =
            (Chronometer)findViewById(R.id.chronometer);
        Button buttonStart = (Button)findViewById(R.id.buttonstart);
        Button buttonStop = (Button)findViewById(R.id.buttonstop);
        Button buttonReset = (Button)findViewById(R.id.buttonreset);

        buttonStart.setOnClickListener(new Button.OnClickListener(){

            @Override
            public void onClick(View v) {
                // TODO Auto-generated method stub
                myChronometer.start();
            }
        });

        buttonStop.setOnClickListener(new Button.OnClickListener(){

            @Override
            public void onClick(View v) {
                // TODO Auto-generated method stub
                myChronometer.stop();
            }
        });

        buttonReset.setOnClickListener(new Button.OnClickListener(){

            @Override
            public void onClick(View v) {
                // TODO Auto-generated method stub
                myChronometer.setBase(SystemClock.elapsedRealtime());
            }
        });
    }
}
```

OUTPUT:



Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Mobile Application Development

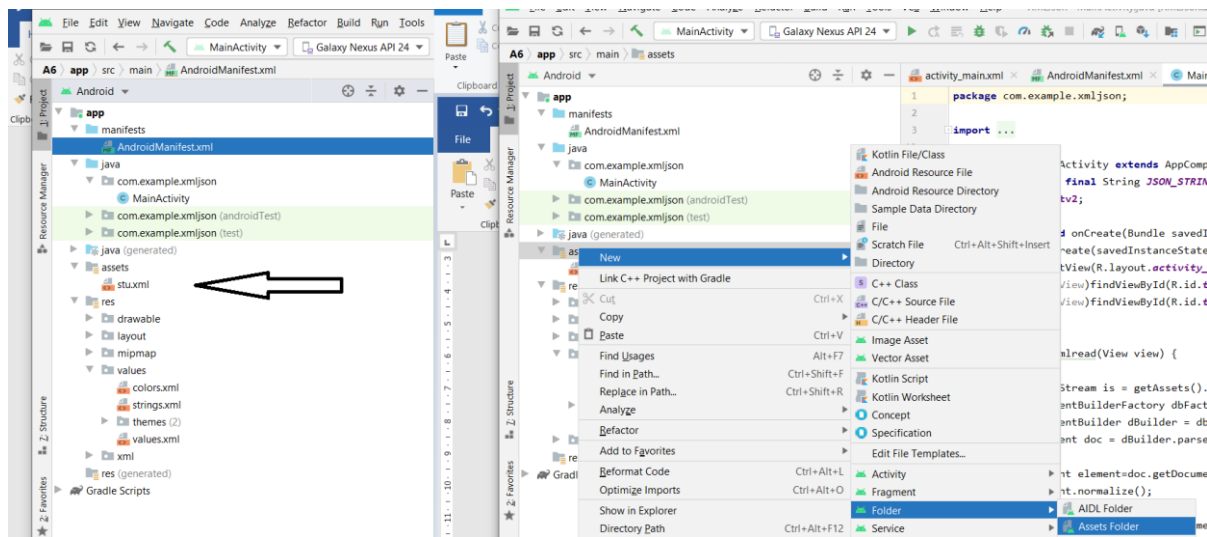
18CSMP68

Program 6: Create two files of XML and JSON type with values for City_Name, Latitude, Longitude, Temperature, and Humidity. Develop an application to create an activity with two buttons to parse the XML and JSON files which when clicked should display the data in their respective layouts side by side.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

Create Assest folder and copy paste the xml and json file



AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.xmljson">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/Theme.XMLJson">
        <meta-data
            android:name="com.google.android.actions"
            android:resource="@xml/actions" />

        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

```
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
</application>

</manifest>
```

MainActivity.java

```
package com.example.xmljson;

import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import org.json.JSONObject;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

import java.io.InputStream;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

public class MainActivity extends AppCompatActivity {
    public static final String
    JSON_STRING="{\"student\":{\"name\":\"Sachin\",\"city\":\"Bengaluru\"}}";
    TextView tv1,tv2;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        tv1=(TextView)findViewById(R.id.tv_xml);
        tv2=(TextView)findViewById(R.id.tv_json);
    }

    public void xmlread(View view) {
        try {
            InputStream is = getAssets().open("stu.xml");
            DocumentBuilderFactory dbFactory =
            DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(is);

            Element element=doc.getDocumentElement();
            element.normalize();

            NodeList nList = doc.getElementsByTagName("student");

            for (int i=0; i<nList.getLength(); i++) {

                Node node = nList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element element2 = (Element) node;
```



```
        tv1.setText(tv1.getText()+"\nName : " + getValue("name",
element2)+"\n");
        tv1.setText(tv1.getText()+"city : " + getValue("city",
element2)+"\n");
        tv1.setText(tv1.getText()+"-----");
    }
}

} catch (Exception e) {e.printStackTrace();}
}
private static String getValue(String tag, Element element) {
    NodeList nodeList =
element.getElementsByTagName(tag).item(0).getChildNodes();
    Node node = nodeList.item(0);
    return node.getNodeValue();
}

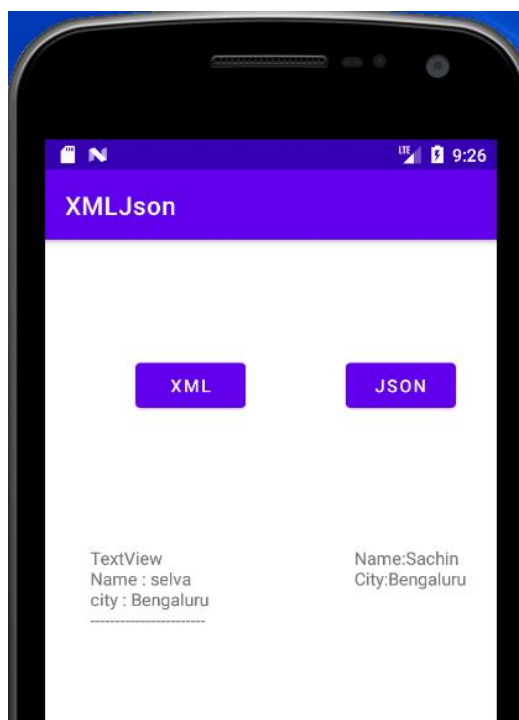
public void readJson(View view) {
    try{

        JSONObject emp=(new JSONObject(JSON_STRING)).getJSONObject("student");
        String empname=emp.getString("name");
        String city=emp.getString("city");

        String str="Name:"+empname+"\n"+"City:"+city;
        tv2.setText(str);

    }catch (Exception e) {e.printStackTrace();}
}
}
```

OUTPUT:



Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 7: Develop a simple application with one Edit Text so that the user can write some text in it. Create a button called “Convert Text to Speech” that converts the user input text into voice.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.texttospeech">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/Theme.TexttoSpeech">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

MainActivity.java

```
package com.example.texttospeech;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.speech.tts.TextToSpeech;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
```

```
import android.widget.Toast;

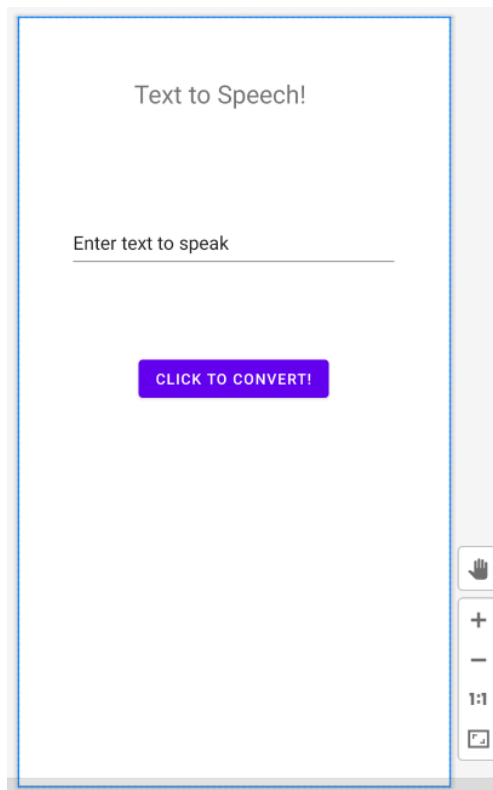
import java.util.Locale;

public class MainActivity extends AppCompatActivity {
    TextToSpeech t1;
    EditText ed1;
    Button b1;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ed1=(EditText)findViewById(R.id.editTextTextPersonName);
        b1=(Button)findViewById(R.id.button);

        t1=new TextToSpeech(getApplicationContext(), new
        TextToSpeech.OnInitListener() {
            @Override
            public void onInit(int status) {
                if(status != TextToSpeech.ERROR) {
                    t1.setLanguage(Locale.UK);
                }
            }
        });
    }

    public void Text2Speech(View view) {
        String toSpeak = ed1.getText().toString();
        Toast.makeText(getApplicationContext(),
        toSpeak,Toast.LENGTH_SHORT).show();
        t1.speak(toSpeak, TextToSpeech.QUEUE_FLUSH, null);
    }
}
```

OUTPUT:



Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 8: Create an activity like a phone dialer with CALL and SAVE buttons. On pressing the CALL button, it must call the phone number and on pressing the SAVE button it must save the number to the phone contacts.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.DataFlair.mycalculator">
    <uses-permission android:name="android.permission.CALL_PHONE" />
    <uses-permission android:name="android.intent.action.CALL_PRIVILEGED"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

MainActivity.java

```
package com.DataFlair.mycalculator;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.provider.ContactsContract;
import android.view.View;
```

```
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    double in1 = 0, i2 = 0;
    TextView editText1;
    boolean Add, Sub, Multiply, Divide, Remainder, deci;
    Button button_0, button_1, button_2, button_3, button_4, button_5, button_6,
    button_7, button_8, button_9, button_Add, button_Sub,
        button_Mul, button_Div, button_Equ, button_del, button_call,
    button_save;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        button_0 = (Button) findViewById(R.id.b0);
        button_1 = (Button) findViewById(R.id.b1);
        button_2 = (Button) findViewById(R.id.b2);
        button_3 = (Button) findViewById(R.id.b3);
        button_4 = (Button) findViewById(R.id.b4);
        button_5 = (Button) findViewById(R.id.b5);
        button_6 = (Button) findViewById(R.id.b6);
        button_7 = (Button) findViewById(R.id.b7);
        button_8 = (Button) findViewById(R.id.b8);
        button_9 = (Button) findViewById(R.id.b9);
        button_del = (Button) findViewById(R.id.BRemain);
        button_call = (Button) findViewById(R.id.buttonDel);
        button_save = (Button) findViewById(R.id.buttoneql);

        editText1 = (TextView) findViewById(R.id.display);

        button_1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                editText1.setText(editText1.getText() + "1");
            }
        });

        button_2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                editText1.setText(editText1.getText() + "2");
            }
        });

        button_3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                editText1.setText(editText1.getText() + "3");
            }
        });

        button_4.setOnClickListener(new View.OnClickListener() {
            @Override
```

```
        public void onClick(View v) {
            editText1.setText(editText1.getText() + "4");
        }
    });

    button_5.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            editText1.setText(editText1.getText() + "5");
        }
    });

    button_6.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            editText1.setText(editText1.getText() + "6");
        }
    });

    button_7.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            editText1.setText(editText1.getText() + "7");
        }
    });

    button_8.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            editText1.setText(editText1.getText() + "8");
        }
    });

    button_9.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            editText1.setText(editText1.getText() + "9");
        }
    });

    button_0.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            editText1.setText(editText1.getText() + "0");
        }
    });

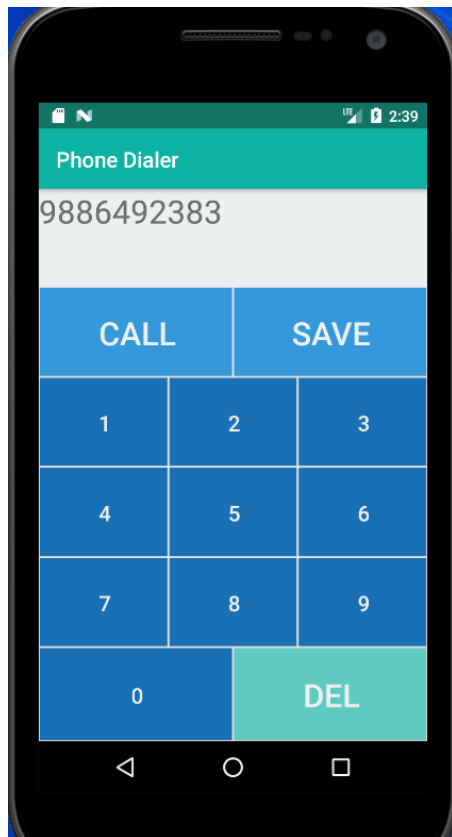
    button_del.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (editText1.getText().length() != 0) {
                editText1.setText("");
                in1 = 0.0;
                i2 = 0.0;
            }
        }
    });
});
```



```
        button_save.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(ContactsContract.Intents.Insert.ACTION);
                intent.setType(ContactsContract.RawContacts.CONTENT_TYPE);
                intent.putExtra(ContactsContract.Intents.Insert.PHONE,
edittext1.getText());
                startActivity(intent);
            }
        });

        button_call.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                final int REQUEST_PHONE_CALL = 1;
                if (android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
                    if (ContextCompat.checkSelfPermission(MainActivity.this,
Manifest.permission.CALL_PHONE) != PackageManager.PERMISSION_GRANTED) {
                        ActivityCompat.requestPermissions(MainActivity.this, new
String[]{Manifest.permission.CALL_PHONE}, REQUEST_PHONE_CALL);
                    }
                    else {
                        String number = edittext1.getText().toString();
                        Intent callIntent = new Intent(Intent.ACTION_CALL);
                        callIntent.setData(Uri.parse("tel:" + number));
                        startActivity(callIntent);
                    }
                }
            }
        });
    }
}
```

OUTPUT:



Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

PART B

Program 1: Write a program to enter Medicine Name, Date and Time of the Day as input from the user and store it in the SQLite database. Input for Time of the Day should be either Morning or Afternoon or Evening or Night. Trigger an alarm based on the Date and Time of the Day and display the Medicine Name.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

MainActivity.java:

```
package com.example.example5;
import android.app.Activity;
import android.app.AlertDialog.Builder;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends Activity implements OnClickListener
{
    EditText Rollno,Name,Marks;
    Button Insert,Delete,Update,View,ViewAll;
    SQLiteDatabase db;
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Rollno=(EditText)findViewById(R.id.Medicine);
        Name=(EditText)findViewById(R.id.meddate);
        Marks=(EditText)findViewById(R.id.medtime);
```

```
Insert=(Button)findViewById(R.id.Insert);
Delete=(Button)findViewById(R.id.Delete);
Update=(Button)findViewById(R.id.Update);
View=(Button)findViewById(R.id.View);
ViewAll=(Button)findViewById(R.id.ViewAll);

Insert.setOnClickListener(this);
Delete.setOnClickListener(this);
Update.setOnClickListener(this);
View.setOnClickListener(this);
ViewAll.setOnClickListener(this);

// Creating database and table
db=openOrCreateDatabase("StudentDB", Context.MODE_PRIVATE,
null);
db.execSQL("CREATE TABLE IF NOT EXISTS student(rollno
VARCHAR,name VARCHAR,marks VARCHAR);");
}

public void onClick(View view)
{
    // Inserting a record to the Student table
    if(view==Insert)
    {
        // Checking for empty fields
        if(Rollno.getText().toString().trim().length()==0||
            Name.getText().toString().trim().length()==0||
            Marks.getText().toString().trim().length()==0)
        {
            showMessage("Error", "Please enter all values");
            return;
        }
        db.execSQL("INSERT INTO student
VALUES('"+Rollno.getText()+"','"+Name.getText()+"
        "','"+Marks.getText()+"')");
        showMessage("Success", "Record added");
        clearText();
    }
    // Deleting a record from the Student table
    if(view==Delete)
    {
        // Checking for empty roll number
        if(Rollno.getText().toString().trim().length()==0)
        {
            showMessage("Error", "Please enter Rollno");
        }
    }
}
```

```
        return;
    }
    Cursor c=db.rawQuery("SELECT * FROM student WHERE
rollno='"+Rollno.getText()+"'", null);
    if(c.moveToFirst())
    {
        db.execSQL("DELETE FROM student WHERE
rollno='"+Rollno.getText()+"'");
        showMessage("Success", "Record Deleted");
    }
    else
    {
        showMessage("Error", "Invalid Rollno");
    }
    clearText();
}
// Updating a record in the Student table
if(view==Update)
{
    // Checking for empty roll number
    if(Rollno.getText().toString().trim().length()==0)
    {
        showMessage("Error", "Please enter Rollno");
        return;
    }
    Cursor c=db.rawQuery("SELECT * FROM student WHERE
rollno='"+Rollno.getText()+"'", null);
    if(c.moveToFirst()) {
        db.execSQL("UPDATE student SET name='"+ Name.getText()
+ "',marks='"+ Marks.getText() +
        "' WHERE rollno='"+Rollno.getText()+"'");
        showMessage("Success", "Record Modified");
    }
    else{
        showMessage("Error", "Invalid Rollno");
    }
    clearText();
}
// Display a record from the Student table
if(view==View)
{
    // Checking for empty roll number
    if(Rollno.getText().toString().trim().length()==0)
    {
        showMessage("Error", "Please enter Rollno");
        return;
    }
}
```

```
}
    Cursor c=db.rawQuery("SELECT * FROM student WHERE
rollno='"+Rollno.getText()+"'", null);
    if(c.moveToFirst())
    {
        Name.setText(c.getString(1));
        Marks.setText(c.getString(2));
    }
    else
    {
        showMessage("Error", "Invalid Rollno");
        clearText();
    }
}
// Displaying all the records
if(view==ViewAll)
{
    Cursor c=db.rawQuery("SELECT * FROM student", null);
    if(c.getCount()==0)
    {
        showMessage("Error", "No records found");
        return;
    }
    StringBuffer buffer=new StringBuffer();
    while(c.moveToNext())
    {
        buffer.append("Rollno: "+c.getString(0)+"\n");
        buffer.append("Name: "+c.getString(1)+"\n");
        buffer.append("Marks: "+c.getString(2)+"\n\n");
    }
    showMessage("Student Details", buffer.toString());
}
}
public void showMessage(String title,String message)
{
    Builder builder=new Builder(this);
    builder.setCancelable(true);
    builder.setTitle(title);
    builder.setMessage(message);
    builder.show();
}
public void clearText()
{
    Rollno.setText("");
    Name.setText("");
    Marks.setText("");
}
```

```
        Rollno.requestFocus();  
    }  
}
```

Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 2: Develop a content provider application with an activity called “Meeting Schedule” which takes Date, Time and Meeting Agenda as input from the user and store this information into the SQLite database. Create another application with an activity called “Meeting Info” having DatePicker control, which on the selection of a date should display the Meeting Agenda information for that particular date, else it should display a toast message saying “No Meeting on this Date”.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

Manifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.call">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Call">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER"
            />
            </intent-filter>
        </activity>
    </application>

</manifest>
```


MainActivity.java:

```
package com.example.call;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import java.util.Calendar;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void AddCalendarEvent(View view) {
        Calendar calendarEvent = Calendar.getInstance();
        Intent i = new Intent(Intent.ACTION_EDIT);
        i.setType("vnd.android.cursor.item/event");
        i.putExtra("beginTime", calendarEvent.getTimeInMillis());
        i.putExtra("allDay", true);
        i.putExtra("rule", "FREQ=YEARLY");
        i.putExtra("endTime", calendarEvent.getTimeInMillis() + 60 * 60 *
1000);
        i.putExtra("title", "Calendar Event");
        startActivity(i);
    }
}
```

Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 3: Create an application to receive an incoming SMS which is notified to the user. On clicking this SMS notification, the message content and the number should be displayed on the screen. Use appropriate emulator control to send the SMS message to your application.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

Manifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.smsnotification">
    <uses-permission android:name="android.permission.WRITE_SMS" />
    <uses-permission android:name="android.permission.READ_SMS" />
    <uses-permission android:name="android.permission.RECEIVE_SMS" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.SMSNotification">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
                    android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <receiver android:name=".SMSIncoming" android:exported="true">
            <intent-filter android:priority="999">
                <action
                    android:name="android.provider.Telephony.SMS_RECEIVED" />
            </intent-filter>
        </receiver>
    </application>
```

```
        </intent-filter>
    </receiver>
</application>

</manifest>
```

MainActivity.java:

```
package com.example.smsnotification;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.content.ContentResolver;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.provider.Telephony;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

    private static MainActivity inst;
    ArrayList<String> smsMessagesList = new ArrayList<String>();
    ListView smsListView;
    ArrayAdapter arrayAdapter;
    public static MainActivity instance() {
        return inst;
    }
    @Override
    public void onStart() {
        super.onStart();
        inst = this;
    }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
smsListView = (ListView) findViewById(R.id.SMSList);
arrayAdapter = new ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1, smsMessagesList);
smsListView.setAdapter(arrayAdapter);
smsListView.setOnItemClickListener(this::onItemClick);
if(ContextCompat.checkSelfPermission(getBaseContext(),
"android.permission.READ_SMS") == PackageManager.PERMISSION_GRANTED) {
    refreshSmsInbox();
}
else
{
    final int REQUEST_CODE_ASK_PERMISSIONS = 123;
    ActivityCompat.requestPermissions(MainActivity.this, new
String[]{"android.permission.READ_SMS"},
REQUEST_CODE_ASK_PERMISSIONS);
}

}

public void refreshSmsInbox() {
    ContentResolver contentResolver = getContentResolver();
    Cursor smsInboxCursor =
contentResolver.query(Uri.parse("content://sms/inbox"), null, null,
null, null);
    int indexBody = smsInboxCursor.getColumnIndex("body");
    int indexAddress = smsInboxCursor.getColumnIndex("address");
    if (indexBody < 0 || !smsInboxCursor.moveToFirst()) return;
    arrayAdapter.clear();
    do {
        String str = "SMS From: " +
smsInboxCursor.getString(indexAddress) +
"\n" + smsInboxCursor.getString(indexBody) + "\n";
        arrayAdapter.add(str);
    } while (smsInboxCursor.moveToNext());
}

public void updateList(final String smsMessage) {
    arrayAdapter.insert(smsMessage, 0);
    arrayAdapter.notifyDataSetChanged();
}

public void onItemClick(AdapterView<?> parent, View view, int pos,
long id) {
    try {
```

```
        String[] smsMessages =
smsMessagesList.get(pos).split("\n");
        String address = smsMessages[0];
        String smsMessage = "";
        for (int i = 1; i < smsMessages.length; ++i) {
            smsMessage += smsMessages[i];
        }

        String smsMessageStr = address + "\n";
        smsMessageStr += smsMessage;
        Toast.makeText(this, smsMessageStr,
Toast.LENGTH_SHORT).show();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
}
```

SMSIncoming.java:

```
package com.example.smsnotification;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.telephony.SmsManager;
import android.telephony.SmsMessage;
import android.util.Log;
import android.widget.Toast;

import com.example.smsnotification.MainActivity;

public class SMSIncoming extends BroadcastReceiver {

    public static final String SMS_BUNDLE = "pdus"; //protocol data
unit (PDU)

    public void onReceive(Context context, Intent intent) {
        Bundle intentExtras = intent.getExtras();
        if (intentExtras != null) {
            Object[] sms = (Object[]) intentExtras.get(SMS_BUNDLE);
            String smsMessageStr = "";
            for (int i = 0; i < sms.length; ++i) {
```

```
SmsMessage smsMessage =
SmsMessage.createFromPdu((byte[]) sms[i]);

String smsBody =
smsMessage.getMessageBody().toString();
String address = smsMessage.getOriginatingAddress();

smsMessageStr += "SMS From: " + address + "\n";
smsMessageStr += smsBody + "\n";
}

//Toast.makeText(context, "Hellooooooo", Toast.LENGTH_LONG).show();
Toast.makeText(context, smsMessageStr,
Toast.LENGTH_SHORT).show();

//this will update the UI with message
MainActivity inst = MainActivity.instance();
inst.updateList(smsMessageStr);
}
}
}
```

Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 4: Write a program to create an activity having a Text box, and also Save, Open and Create buttons. The user has to write some text in the Text box. On pressing the Create button the text should be saved as a text file in Mksdcard. On subsequent changes to the text, the Save button should be pressed to store the latest content to the same file. On pressing the Open button, it should display the contents from the previously stored files in the Text box. If the user tries to save the contents in the Textbox to a file without creating it, then a toast message has to be displayed saying “First Create a File”.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

Manifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.sdcard">
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.SdCard">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Java file:

```
package com.example.sdcard;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.content.pm.PackageManager;
import android.os.Build;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;

public class MainActivity extends AppCompatActivity {

    EditText txtdata;
    Button btn_save, btn_open, btn_create;
    // Storage Permissions
    private static final int REQUEST_EXTERNAL_STORAGE = 1;
    private static String[] PERMISSIONS_STORAGE = {
        Manifest.permission.READ_EXTERNAL_STORAGE,
        Manifest.permission.WRITE_EXTERNAL_STORAGE
    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        txtdata = findViewById(R.id.editTextcontent);
        btn_save = findViewById(R.id.btn_save);
        btn_open = findViewById(R.id.btn_open);
    }
}
```

```
        btn_create = findViewById(R.id.btn_create);

    }

    public void call_create(View view) {
        String filename = "myfile1.txt";
        String data = txtdata.getText().toString();

        FileOutputStream fos;
        int permission =
        ActivityCompat.checkSelfPermission(MainActivity.this,
        Manifest.permission.WRITE_EXTERNAL_STORAGE);

        if (permission != PackageManager.PERMISSION_GRANTED) {
            // We don't have permission so prompt the user
            ActivityCompat.requestPermissions(
                MainActivity.this,
                PERMISSIONS_STORAGE,
                REQUEST_EXTERNAL_STORAGE
            );
        }
        else{
            try {
                File myFile = new File("/sdcard/" + filename);
                myFile.createNewFile();
                Toast.makeText(getApplicationContext(),
filename + " Created", Toast.LENGTH_LONG).show();

            } catch (IOException e) {
                e.printStackTrace();
            }
        }

        public void call_save(View view) {
            String filename = "myfile1.txt";
            String data = txtdata.getText().toString();

            File file = new File("/sdcard/"+filename);
            if(file.exists())
            {
                FileOutputStream fos;
                try {
                    FileOutputStream fOut = new
FileOutputStream("/sdcard/"+filename);
```

```
        OutputStreamWriter myOutWriter = new
OutputStreamWriter(fOut);
        myOutWriter.append(data);
        myOutWriter.close();
        fOut.close();
        Toast.makeText(getApplicationContext(), filename + "
Saved", Toast.LENGTH_LONG).show();
        txtdata.setText("");

    } catch (IOException e) {
        Toast.makeText(getApplicationContext(), "First create a
File", Toast.LENGTH_LONG).show();
        e.printStackTrace();
    }
}

else
    Toast.makeText(getApplicationContext(), "First create a
File", Toast.LENGTH_LONG).show();
}

public void call_open(View view) {
    String filename = "myfile1.txt";
    String aDataRow = "";
    String aBuffer = "";
    try {
        File myFile = new File("/sdcard/"+filename);
        FileInputStream fIn = new FileInputStream(myFile);
        BufferedReader myReader = new BufferedReader(
            new InputStreamReader(fIn));
        while ((aDataRow = myReader.readLine()) != null) {
            aBuffer += aDataRow + "\n";
        }
        myReader.close();
    } catch (IOException e) {
        e.printStackTrace();
    }

    Toast.makeText(getApplicationContext(), aBuffer, Toast.LENGTH_LONG).show
    ();
    txtdata.setText(aBuffer);
}
}
```

Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 5: Create an application to demonstrate a basic media player that allows the user to Forward, Backward, Play and Pause an audio. Also, make use of the indicator in the seek bar to move the audio forward or backward as required.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

Java file:

```
package com.example.mediaplayer;

import androidx.appcompat.app.AppCompatActivity;

import android.media.MediaPlayer;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.widget.ImageButton;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;

import java.util.concurrent.TimeUnit;

public class MainActivity extends AppCompatActivity {
    private ImageButton forwardbtn, backwardbtn, pausebtn, playbtn;
    private MediaPlayer mPlayer;
    private TextView songName, startTime, songTime;
    private SeekBar songPrgs;
    private static int oTime = 0, sTime = 0, eTime = 0, fTime = 5000,
    bTime = 5000;
    private Handler hdlr = new Handler();
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        backwardbtn = (ImageButton)findViewById(R.id.btnBackward);
        forwardbtn = (ImageButton)findViewById(R.id.btnForward);
        playbtn = (ImageButton)findViewById(R.id.btnPlay);
        pausebtn = (ImageButton)findViewById(R.id.btnPause);
```

```
songName = (TextView)findViewById(R.id.txtSname);
startTime = (TextView)findViewById(R.id.txtStartTime);
songTime = (TextView)findViewById(R.id.txtSongTime);
songName.setText("Sample MP3");
mPlayer = MediaPlayer.create(this, R.raw.sample);
songPrgs = (SeekBar)findViewById(R.id.sBar);
songPrgs.setClickable(true);
pausebtn.setEnabled(true);

playbtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(MainActivity.this, "Playing Audio",
Toast.LENGTH_SHORT).show();
        mPlayer.start();
        eTime = mPlayer.getDuration();
        sTime = mPlayer.getCurrentPosition();
        if(oTime == 0){
            songPrgs.setMax(eTime);
            oTime =1;
        }
        songTime.setText(String.format("%d min, %d sec",
TimeUnit.MILLISECONDS.toMinutes(eTime),
            TimeUnit.MILLISECONDS.toSeconds(eTime) -
TimeUnit.MINUTES.toSeconds(TimeUnit.MILLISECONDS.toMinutes(eTime)))
);
        startTime.setText(String.format("%d min, %d sec",
TimeUnit.MILLISECONDS.toMinutes(sTime),
            TimeUnit.MILLISECONDS.toSeconds(sTime) -
TimeUnit.MINUTES.toSeconds(TimeUnit.MILLISECONDS.toMinutes(sTime)))
);

        songPrgs.setProgress(sTime);
        hdlr.postDelayed(UpdateSongTime, 100);
        pausebtn.setEnabled(true);
        playbtn.setEnabled(false);
    }
});
pausebtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mPlayer.pause();
        pausebtn.setEnabled(false);
        playbtn.setEnabled(true);
        Toast.makeText(getApplicationContext(), "Pausing
Audio", Toast.LENGTH_SHORT).show();
    }
});
```

```
    });
    forwardbtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if((sTime + fTime) <= eTime)
            {
                sTime = sTime + fTime;
                mPlayer.seekTo(sTime);
            }
            else
            {
                Toast.makeText(getApplicationContext(), "Cannot
jump forward 5 seconds", Toast.LENGTH_SHORT).show();
            }
            if(!playbtn.isEnabled()){
                playbtn.setEnabled(true);
            }
        }
    });
    backwardbtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if((sTime - bTime) > 0)
            {
                sTime = sTime - bTime;
                mPlayer.seekTo(sTime);
            }
            else
            {
                Toast.makeText(getApplicationContext(), "Cannot
jump backward 5 seconds", Toast.LENGTH_SHORT).show();
            }
            if(!playbtn.isEnabled()){
                playbtn.setEnabled(true);
            }
        }
    });
}

private Runnable UpdateSongTime = new Runnable() {
    @Override
    public void run() {
        sTime = mPlayer.getCurrentPosition();
        startTime.setText(String.format("%d min, %d sec",
TimeUnit.MILLISECONDS.toMinutes(sTime),
        TimeUnit.MILLISECONDS.toSeconds(sTime) -
TimeUnit.MINUTES.toSeconds(TimeUnit.MILLISECONDS.toMinutes(sTime))) );
    }
}
```

```
        songPrgs.setProgress(sTime);  
        hdlr.postDelayed(this, 100);  
    }  
};  
}
```

Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 6: Develop an application to demonstrate the use of Asynchronous tasks in android. The asynchronous task should implement the functionality of a simple moving banner. On pressing the Start Task button, the banner message should scroll from right to left. On pressing the Stop Task button, the banner message should stop. Let the banner message be “Demonstration of Asynchronous Task”.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

ActivityMain.xml

```
<?xml version = "1.0" encoding = "utf-8"?>

<LinearLayout xmlns:android = "http://schemas.android.com/apk/res/android"
    xmlns:tools = "http://schemas.android.com/tools"
    android:id = "@+id/rootview"
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:orientation = "vertical"
    android:background = "#c1c1c1"
    android:gravity = "center_horizontal"
    tools:context = ".MainActivity">

    <Button
        android:id = "@+id/asyncTask"
        android:text = "Download"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"/>

    <ImageView
        android:id = "@+id/image"
```



```
        android:layout_width = "300dp"

        android:layout_height = "300dp" />

<ImageView

    android:id = "@+id/image2"

    android:layout_width = "300dp"

    android:layout_height = "300dp" />

</LinearLayout>
```

MainActivity.java

```
package com.example.example6_b;

import android.app.ProgressDialog;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.os.AsyncTask;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

import android.view.View;

import android.widget.Button;

import android.widget.ImageView;

import java.io.IOException;

import java.io.InputStream;

import java.net.HttpURLConnection;

import java.net.URL;

public class MainActivity extends AppCompatActivity {

    URL imageUrl = null;
```

```
InputStream is = null;

Bitmap bmImg = null;

ImageView imageView = null;

ImageView imageView2 = null;

AsyncTaskExample asyncTask = null;

AsyncTaskExample2 asyncTask2 = null;

ProgressDialog p;

@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_main);

    Button button = findViewById(R.id.asyncTask);

    imageView = findViewById(R.id.image);

    imageView2 = findViewById(R.id.image2);

    button.setOnClickListener(new View.OnClickListener() {

        @Override

        public void onClick(View v) {

            asyncTask2 = new AsyncTaskExample2();

            asyncTask2.executeOnExecutor(AsyncTask.THREAD_POOL_EXECUTOR,

"https://www.tutorialspoint.com/cprogramming/images/logo.png");

            asyncTask = new AsyncTaskExample();

            asyncTask.executeOnExecutor(AsyncTask.THREAD_POOL_EXECUTOR,

"https://www.tutorialspoint.com/images/tp-logo-diamond.png");

        }

    });

}

private class AsyncTaskExample extends AsyncTask<String, String, Bitmap> {

    @Override

    protected void onPreExecute() {
```

```
super.onPreExecute();

p = new ProgressDialog(MainActivity.this);

p.setMessage("Please wait...It is downloading");

p.setIndeterminate(true);

p.setCancelable(false);

p.show();

}

@Override

protected Bitmap doInBackground(String... strings) {

    try {

        imageUrl = new URL(strings[0]);

        HttpURLConnection conn = (HttpURLConnection) imageUrl

            .openConnection();

        conn.setDoInput(true);

        conn.connect();

        is = conn.getInputStream();

        BitmapFactory.Options options = new BitmapFactory.Options();

        options.inPreferredConfig = Bitmap.Config.RGB_565;

        bmImg = BitmapFactory.decodeStream(is, null, options);

    } catch (IOException e) {

        e.printStackTrace();

    }

    return bmImg;

}

@Override

protected void onPostExecute(Bitmap bitmap) {

    super.onPostExecute(bitmap);

    if (imageView!= null) {
```

```
        p.hide();

        imageView.setImageBitmap(bitmap);
    } else {
        p.show();
    }
}

}

private class AsyncTaskExample2 extends AsyncTask<String, String, Bitmap> {

    @Override

    protected Bitmap doInBackground(String... strings) {

        try {

            ImageUrl = new URL(strings[0]);

            HttpURLConnection conn = (HttpURLConnection) ImageUrl

                .openConnection();

            conn.setDoInput(true);

            conn.connect();

            is = conn.getInputStream();

            BitmapFactory.Options options = new BitmapFactory.Options();

            options.inPreferredConfig = Bitmap.Config.RGB_565;

            bmImg = BitmapFactory.decodeStream(is, null, options);

        } catch (IOException e) {

            e.printStackTrace();

        }

        return bmImg;

    }

    @Override

    protected void onPostExecute(Bitmap bitmap) {

        super.onPostExecute(bitmap);

    }

}
```

```
        if (imageView2 != null) {  
            imageView2.setImageBitmap(bitmap);  
        } else {  
        }  
    }  
}  
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>  
  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
    package="com.example.example6_b">  
    <uses-permission android:name="android.permission.INTERNET"/>  
  
    <application  
        android:allowBackup="true"  
        android:icon="@mipmap/ic_launcher"  
        android:label="@string/app_name"  
        android:roundIcon="@mipmap/ic_launcher_round"  
        android:supportsRtl="true"  
        android:theme="@style/Theme.Example6B">  
        <activity android:name=".MainActivity">  
            <intent-filter>  
                <action android:name="android.intent.action.MAIN" />  
  
                <category android:name="android.intent.category.LAUNCHER" />  
            </intent-filter>  
        </activity>  
    </application>  
</manifest>
```

</activity>

</application>

</manifest>

Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 7: Develop an application that makes use of the clipboard framework for copying and pasting of the text. The activity consists of two EditText controls and two Buttons to trigger the copy and paste functionality.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

```
package com.example.clipboard;

import androidx.appcompat.app.AppCompatActivity;

import android.content.ClipData;
import android.content.ClipboardManager;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    EditText ed1, ed2;
    Button b1, b2;

    private ClipboardManager myClipboard;
    private ClipData myClip;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        ed1 = (EditText) findViewById(R.id.editText);
        ed2 = (EditText) findViewById(R.id.editText2);

        b1 = (Button) findViewById(R.id.button);
        b2 = (Button) findViewById(R.id.button2);
    }
}
```

```
myClipboard = (ClipboardManager)
getSystemService(CLIPBOARD_SERVICE);

b1.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {
        String text;
        text = ed1.getText().toString();

        myClip = ClipData.newPlainText("text", text);
        myClipboard.setPrimaryClip(myClip);

        Toast.makeText(getApplicationContext(), "Text Copied",
            Toast.LENGTH_SHORT).show();
    }
});

b2.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {
        ClipData abc = myClipboard.getPrimaryClip();
        ClipData.Item item = abc.getItemAt(0);

        String text = item.getText().toString();
        ed2.setText(text);

        Toast.makeText(getApplicationContext(), "Text Pasted",
            Toast.LENGTH_SHORT).show();
    }
});
}
```

Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Program 8: Create an AIDL service that calculates Car Loan EMI. The formula to calculate EMI is

$$E = P * (r(1+r)^n) / ((1+r)^n - 1)$$

where

E = The EMI payable on the car loan amount

P = The Car loan Principal Amount

r = The interest rate value computed on a monthly basis

n = The loan tenure in the form of months

The down payment amount has to be deducted from the principal amount paid towards buying the Car. Develop an application that makes use of this AIDL service to calculate the EMI. This application should have four EditText to read the Principal Amount, Down Payment, Interest Rate, Loan Term (in months) and a button named as “Calculate Monthly EMI”. On click of this button, the result should be shown in a TextView. Also, calculate the EMI by varying the Loan Term and Interest Rate values.

Program Objective:

- Learn and acquire the art of Android Programming.
- Configure Android studio to run the applications.
- Understand and implement Android's User interface functions.
- Create, modify and query on SQLite database.
- Inspect different methods of sharing data using services

Manifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.aidlprogram">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.AIDLProgram">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
                    android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

```
</activity>
<service
    android:name=".AdditionService">
    <intent-filter>
        <action android:name="aidlprogram" />
    </intent-filter>
</service>

</application>

</manifest>
```

<https://www.protechtraining.com/blog/post/implementing-remote-interface-using-aidl-66>

Create new AIDL file:

```
// IMyAidlInterface.aidl
package com.example.aidlprogram;

// Declare any non-default types here with import statements

interface IMyAidlInterface {
    /**
     * Demonstrates some basic types that you can use as parameters
     * and return values in AIDL.
     */
    double add(int p, int r, int n);
}
```

Rebuild the project to create Interface file automatically.

Create a service (new java file):

AdditionService.java:

```
package com.example.aidlprogram;

import android.app.Service;
```

```
import android.content.Intent;
import android.os.IBinder;
import android.os.RemoteException;
import android.util.Log;

import static java.lang.Math.pow;

public class AdditionService extends Service {
    private static final String TAG = "AdditionService";

    @Override
    public void onCreate() {
        super.onCreate();
        Log.d(TAG, "onCreate()");
    }

    @Override
    public IBinder onBind(Intent intent) {

        return new IMyAidlInterface.Stub() {
            /**
             * Implementation of the add() method
             */
            public double add(int p, int r, int n) throws
RemoteException {
                Log.d(TAG, String.format("AdditionService.add(%d,
%d)", p, r));
                return p*((r * pow((1+r),n)) /((pow((1+r),n) -1)));
            }

        };
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        Log.d(TAG, "onDestroy()");
    }
}
```

Main Activity.java:

```
package com.example.aidlprogram;
```

```
import androidx.appcompat.app.AppCompatActivity;

import android.content.ComponentName;
import android.content.Context;
import android.content.Intent;
import android.content.ServiceConnection;
import android.os.Bundle;
import android.os.IBinder;
import android.os.RemoteException;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private static final String TAG = "AIDLDemo";
    IMyAidlInterface service;
    AdditionServiceConnection connection;

    /**
     * This class represents the actual service connection. It casts
     the bound
     * stub implementation of the service to the AIDL interface.
     */
    class AdditionServiceConnection implements ServiceConnection {

        public void onServiceConnected(ComponentName name, IBinder
boundService) {
            service = IMyAidlInterface.Stub.asInterface((IBinder)
boundService);
            Log.d(TAG, "onServiceConnected() connected");
            Toast.makeText(getApplicationContext(), "Service
connected", Toast.LENGTH_LONG)
                .show();
        }

        public void onServiceDisconnected(ComponentName name) {
            service = null;
            Log.d(TAG, "onServiceDisconnected() disconnected");
            Toast.makeText(getApplicationContext(), "Service
connected", Toast.LENGTH_LONG)
                .show();
        }
    }
}
```

```
}

/** Binds this activity to the service. */
private void initService() {
    connection = new AdditionServiceConnection();
    Intent i = new Intent();
    i.setClassName("com.example.aidlprogram",
com.example.aidlprogram.AdditionService.class.getName());
    boolean ret = bindService(i, connection,
Context.BIND_AUTO_CREATE);
    Log.d(TAG, "initService() bound with " + ret);
}

/** Unbinds this activity from the service. */
private void releaseService() {
    unbindService(connection);
    connection = null;
    Log.d(TAG, "releaseService() unbound.");
}

/** Called when the activity is first created. */
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    initService();

    // Setup the UI
    Button buttonCalc = (Button) findViewById(R.id.btn_calculate);

    buttonCalc.setOnClickListener(new View.OnClickListener() {
        TextView result = (TextView)
findViewById(R.id.txt_result);
        EditText value1 = (EditText)
findViewById(R.id.edtPrinciple);
        EditText value2 = (EditText)
findViewById(R.id.edtDownpayment);
        EditText value3 = (EditText)
findViewById(R.id.edtInterestRate);
        EditText value4 = (EditText)
findViewById(R.id.edtLoanTerm);
        public void onClick(View v) {
            int P,Dp,r, n;
            double res = 0;
            P = Integer.parseInt(value1.getText().toString());
```

```
Dp = Integer.parseInt(value2.getText().toString());
r = Integer.parseInt(value2.getText().toString());
n = Integer.parseInt(value2.getText().toString());

//Toast.makeText(getApplicationContext(),String.valueOf(v1),Toast.LENGTH_LONG).show();

//Toast.makeText(getApplicationContext(),String.valueOf(v2),Toast.LENGTH_LONG).show();
        try {
            res = service.add(P, r,n);
        } catch (RemoteException e) {
            Log.d(TAG, "onClick failed with: " + e);
            e.printStackTrace();
        }
        result.setText(String.valueOf(res));
    }
});
}

/** Called when the activity is about to be destroyed. */
@Override
protected void onDestroy() {
    releaseService();
}

}
```

Program Outcome:

- Create, test and debug Android application by setting up Android development environment.
- Implement adaptive, responsive user interfaces that work across a wide range of devices.
- Infer long running tasks and background work in Android applications.
- Demonstrate methods in storing, sharing and retrieving data in Android applications.
- Infer the role of permissions and security for Android applications.

Viva questions

1. What is Android?
2. What Is the Google Android SDK?
3. What is the Android Architecture?
4. Describe the Android Framework.
5. What is AAPT?
6. What is the importance of having an emulator within the Android environment?
7. What is the use of an activityCreator?
8. What is the importance of XML-based layouts?
9. What is Orientation?
10. What items are important in every Android project?
11. Name the languages supported for Android development.
12. What are Intents?
13. Describe Activities.
14. What are containers?
15. What is the importance of Android in the mobile market?
16. What do you think are some disadvantages of Android?
17. What is adb?
18. What are the four essential states of an activity?
19. What is ANR?
20. Which elements can occur only once and must be present?
21. How are escape characters used as attribute?
22. Difference between margin & padding?
23. What is ConstraintLayout?
24. When is the onStop() method invoked?
25. What is GUI?
26. What role does Dalvik play in Android development?
27. What is the AndroidManifest.xml?
28. What is the importance of Default Resources?
29. What is AIDL?

30. What data types are supported by AIDL?
31. What is a Fragment?
32. What is a visible activity?
33. When is the best time to kill a foreground activity?
34. Difference between RelativeLayout and LinearLayout?
35. What Is Rss (really Simple Syndication)?
36. What Is The Relation Between Rss And Xml?
37. How Many Versions Of Rss Language Standards?
38. What is content Syndication?
39. Why Create Rss?
40. Explain Child Elements Of <channel>?
41. What Are The Benefits To Rss?
42. What Is A Desktop Rss Aggregator?
43. What Is A Rss Aggregator?
44. What Is An Online Rss Aggregator?
45. Explain The Disadvantages Of Rss?
46. What are the graphics primitives?
47. What composes a typical Android application project?
48. What is a Sticky Intent?
49. What language is supported by Android for application development?
50. What is an action?
51. Do all mobile phones support the latest Android operating system?
52. Describe android.graphics.Canvas method.
53. Describe android.graphics.Paint method
54. What are the core components under the Android application architecture?
55. Enlist the drawing objects?
56. What is Query Language?
57. What is SQLite?
58. Who was the designer of SQLite?
59. What are the most important features of SQLite?
60. What are the advantages of using SQLite?

61. How would you create a database in SQLite?
62. How would you create a table in SQLite database?
63. How can you delete the existing records from a table in SQLite?
64. Explain the difference between SQL and SQLite.
65. List Out The Standard Sqlite Commands?
66. What is Context?
67. Why bytecode cannot be run in Android?
68. What is a BuildType in Gradle? And what can you use it for?
69. What is the Android Application Architecture?
70. What Is Gps?
71. How Is Gps Used?
72. Who Uses Gps?
73. How do you find GPS location on Android?
74. How do you keep your android phone from being GPS tracked?
75. What do ADT stands for?
76. What is viewGroup in android?
77. What is a content provider in android?
78. What are the notifications available in android?
79. Where layouts are placed in android?
80. What are the different storages available in android?
81. What is the full form of SD?
82. How to launch an activity in android?
83. What is drawable folder in android?
84. What are the different versions of Android OS?
85. What is .apk extension in Android?
86. What is the basic structure for developing a game?
87. What is "Pixel Art"?
88. What do you mean by "Lag" ?
89. What are the Android tools used for developing games?
90. In Android, how you can use load texture method to load the image?
91. What is a game loop?

- 92. What are the common errors done by programmer while programming?
- 93. What are the gaming engines you can use for developing games?
- 94. Describe Lifecycle of an Activity.
- 95. Why would you do the setContentView() in onCreate() of Activity class?
- 96. What is nine-patch images tool in android?
- 97. What is a bitmap image?
- 98. Explain Bitmap Class
- 99. What is Orientation?
- 100. How to show image using ImageView in Android?
- 101. What is the difference between a regular .png and a nine-patch image?
- 102. What is Dalvik Virtual Machine?
- 103. How Android app runs on Android mobile?
- 104. What is Toast in Android?
- 105. What are Loaders in Android?