_____

WINTER – 2022 EXAMINATION

Subject Name Mobile Application Development   **Model Answer**   **Subject Code:** 22617

Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.
8) As per the policy decision of Maharashtra State Government, teaching in English/Marathi and Bilingual (English + Marathi) medium is introduced at first year of AICTE diploma Programme from academic year 2021-2022. Hence if the students in first year (first and second semesters) write answers in Marathi or bilingual language (English +Marathi), the Examiner shall consider the same and assess the answer based on matching of concepts with model answer.

Q. No.	Sub Q. N.	*Note: As Android programs contain many of the auto generated code statements from IDE, while assessing such answers correct logical steps taken to obtain the required output can be considered.* **Answer**	Marking Scheme
1		**Attempt any <u>FIVE</u> of the following:**	**10 M**
	a)	**Define OHA and state goal of OHA.**	**2 M**
	Ans	The OHA is the group that is in charge of the Android smartphones operating system. It was created by Google. The Open Handset Alliance (OHA) is consortium of multiple companies like Samsung, Sony, Intel and many more to provide services and deploy handsets using android platform. The Open Handset Alliance (OHA) is an association whose goal is to develop open standards for mobile devices, promote innovation in mobile phones and provide a better experience for consumers at a lower cost.	(Define : 1 M, Stating goal 1 M)
	b)	**Define Android Virtual Devices (AVD).**	**2 M**
	Ans	An Android Virtual Device (AVD) is a configuration that defines the characteristics of an Android phone, tablet, Wear OS, Android TV, or Automotive OS device that you want to simulate in the Android Emulator. The AVD Manager is an interface you can launch from Android Studio that helps you create and manage AVDs.	(Define : 2 M ,any valid definition should be considered)

c)	**State how the APK files are built?**		**2 M**
Ans	• The javac compiler compiles the java source file into the class file. • The dx tool takes all the class files of your application and generates a single .dex file which is a platform specific tool. • Then Android Assets Packaging Tool (aapt) handles the packaging process and finally creates an executable file with extension .apk. • An APK file contains all of a program's code (such as .dex files), resources, assets, certificates, and manifest file.		**(For each step 1/2 M)**
d)	**Enlist the elements of UI.**		**2 M**

Ans	**Elements of UI.**
	
	    • EditText
	    • TextView
	    • ListView
	    • GridView
	    • ScrollView
	    • ImageView
	    • ToggleButton
	    • CheckBox
	    • RadioButton
	
	    OR
	Components of UI
	    • Menu Bar
	    • Toolbar
	    • Navigation Bar
	    • Editor Tabs
	    • Editor
	    • Project Explorer
	    • Status Bar
	    • Tool Buttons
			(1/2 mark for each UI element, any 4 UI elements expected)

e)	**State the uses of Intent in Android.**		**2 M**
Ans	An Intent is a messaging object you can use to request an action from another app component. Intents are used for facilitating communication between components like Activities, Services and Broadcast Receivers.		**(1 M for each use, 2 uses expected)**
f)	**Name any four methods to get location data in android.**		**2 M**
Ans	•float distanceTo(Location dest) •float getAccuracy() •float getBearing()		**(1/2 M for each method,**

		•double getAltitude() •double getLatitude() •float getSpeed() •boolean hasAccuracy() •boolean hasAltitude() •boolean hasBearing() •boolean hasBearing() •boolean hasSpeed() •void reset() •void setAccuracy(float accuracy) •void setAltitude(double altitude)	**any 4 methods expected)**
	g)	**Explain significance of content provider.**	**2 M**
	Ans	Content Providers are used to share data between two applications. This can be implemented in two ways: 1. When you want to implement the existing content provider in another application. 2. When you want to create a new content provider that can share its data with other Applications	**(1 M for each significance,**
2.		**Attempt any THREE of the following:**	**12 M**
	a)	**Describe the different features of Android.**	**4 M**
	Ans	**User Interface:** The user interface of the Android operating system is straight forward, and these features make it very user friendly. **Multiple Language Support:** Android supports multiple languages in its operating system and one can change the language very easily based on one's requirement, the International languages supported are English, Germany, Chinese, Dutch, French, German, Japanese, Korean, Russian, and many more also some native language of India is also Supported Like Hindi, Marathi, Gujarati, Punjabi and many more. **Multi-tasking:** Android provides support to run apps and services in the background with ease which allows the users to use multiple apps at the same time. **Connectivity:** Android has extensive support to the connectivity and it supports connectivity such as WiFi, Bluetooth, Hotspot, CDMA, GSM, NFC, VOLTE, UBB, VPN, 3G network band, and 4G Network Band. **Extensive Application Support:** Android have Play store which is used as the major tool to download and update applications on the operating system, however, one can download the installer(often called as APK file) and install it manually, but it is not much recommended as third party applications could be prone to some security breach in the smartphones.	**(1 M for each feature, 4 features expected)**
	b)	**Explain features of Android SDK.**	**4 M**
	Ans	Android SDK is a collection of libraries and Software Development tools that are essential for Developing Android Applications. Whenever Google releases a new version or update of	**(1 M for each**

		Android Software, a corresponding SDK also releases with it. In the updated or new version of SDK, some more features are included which are not present in the previous version. Android SDK consists of some tools which are very essential for the development of Android Application. These tools provide a smooth flow of the development process from developing and debugging. Android SDK is compatible with all operating systems such as Windows, Linux, macOS, etc. **1. Android Tool:** This tool helps in managing the Android Virtual Device projects as well as the installed components of the software development kit. **2. Emulator Tool:** It helps us in testing the applications without the need of using the application on an actual device. **3. Dalvik Debug Monitor Server(DDMS):** DDMS is very useful for debugging the Android Application. **4. Android Debug Bridge (ADB):** It is a very versatile command-line tool and is helpful for the communication between the developer and the Emulator or the Android device that is connected.	**feature, any 4 features expected)**
	c)	**Explain the Android security model.**	**4 M**
	Ans	The Android security model is primarily based on a sandbox and permission mechanism. Each application is running in a specific Dalvik virtual machine with a unique user ID assigned to it, which means the application code runs in isolation from the code of all other applications. Therefore, one application has not granted access to other applications' files. Android application has been signed with a certificate with a private key Know the owner of the application is unique. This allows the author of the application will be identified if needed. When an application is installed in the phone is assigned a user ID, thus avoiding it from affecting it other applications by creating a sandbox for it. This user ID is permanent on which devices and applications with the same user ID are allowed to run in a single process. This is a way to ensure that a malicious application has Cannot access / compromise the data of the genuine application. It is mandatory for an application to list all the resources it will Access during installation. Terms are required of an application, in the installation process should be user-based or interactive Check with the signature of the application. **Declaring and Using Permissions** The purpose of a permission is to protect the privacy of an Android user. Android apps must request permission to access sensitive user data (such as contacts and SMS), as well as certain system features (such as camera and internet). Depending on the feature, the system might grant the permission automatically or might prompt the user to approve the request. Permissions are divided into several protection levels. The protection level affects whether runtime permission requests are required. There are three protection levels that affect third-party apps: normal, signature, and dangerous permissions. **Normal permissions:** Normal permissions cover areas where your app needs to access data or resources outside the app's sandbox, but where there's very little risk to the user's privacy or	**2 M for explanation , 2 M for explaining permissions , any 2 permissions expected)**

		the operation of other apps. For example, permission to set the time zone is a normal permission. If an app declares in its manifest that it needs a normal permission, the system automatically grants the app that permission at install time. The system doesn't prompt the user to grant normal permissions, and users cannot revoke these permissions.	
		Signature permissions: The system grants these app permissions at install time, but only when the app that attempts to use permission is signed by the same certificate as the app that defines the permission.	
		Dangerous permissions: Dangerous permissions cover areas where the app wants data or resources that involve the user's private information, or could potentially affect the user's stored data or the operation of other apps. For example, the ability to read the user's contacts is a dangerous permission. If an app declares that it needs a dangerous permission, the user must explicitly grant the permission to the app. Until the user approves the permission, your app cannot provide functionality that depends on that permission. To use a dangerous permission, your app must prompt the user to grant permission at runtime. For more details about how the user is prompted, see Request prompt for dangerous permission.	
	d)	**Elaborate the need of permissions in Android. Explain the permissions to set system functionalitics like SEND-SMS, bluetooth.**	**4 M**
	Ans	The purpose of a permission is to protect the privacy of an Android user. Android apps must request permission to access sensitive user data (such as contacts and SMS), as well as certain system features (such as camera and internet). Depending on the feature, the system might grant the permission automatically or might prompt the user to approve the request. • **android. permission. SEND_SMS** Allows the app to send SMS messages. This may result in unexpected charges. Malicious apps may cost you money by sending messages without your confirmation. Following is the code snippet to set SEND_SMS permissions in manifest file. <uses-permission android: name="android.permission.SEND_SMS"/> • **android. permission. BLUETOOTH** You need to provide following permissions in AndroidManifest.xml file. <uses-permission android:name="android.permission.BLUETOOTH" /> <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />	**(2 M for explanation of need, 1 M each for explaining permissions of Sms and bluetooth)**
3.		**Attempt any THREE of the following:**	**12 M**
	a)	**Explain data and time picker with its method.**	**4 M**
	Ans	**Date Picker:** In Android, DatePicker is a widget used to select a date. It allows to select date by day, month and year in our custom UI (user interface). If we need to show this view as a dialog then we have to use a DatePickerDialog class. **Methods of DatePicker:**	**Date picker (any two methods) 2 M and time picker (any**

1. setSpinnersShown(boolean shown): This method is used to set whether the spinner of the date picker in shown or not. In this method you have to set a Boolean value either true or false. True indicates spinner is shown, false value indicates spinner is not shown. Default value for this function is true.

Syntax:

DatePicker simpleDatePicker = (DatePicker)findViewById(R.id.simpleDatePicker);

simpleDatePicker.setSpinnersShown(false);

2. getDayOfMonth(): This method is used to get the selected day of the month from a date picker.

This method returns an integer value.

DatePicker simpleDatePicker = (DatePicker) findViewById(R.id.simpleDatePicker);

int day = simpleDatePicker.getDayOfMonth();

3. getMonth(): This method is used to get the selected month from a date picker. This method returns an integer value.

DatePicker simpleDatePicker = (DatePicker)findViewById(R.id.simpleDatePicker);

int month = simpleDatePicker.getMonth();

4. getYear(): This method is used to get the selected year from a date picker. This method returns an integer value.

DatePicker simpleDatePicker = (DatePicker)findViewById(R.id.simpleDatePicker);

int year = simpleDatePicker.getYear();

5. getFirstDayOfWeek(): This method is used to get the first day of the week. This method returns an integer value.

DatePicker simpleDatePicker = (DatePicker)findViewById(R.id.simpleDatePicker);

int firstDay=simpleDatePicker.getFirstDayOfWeek();

TimePicker:

In Android, TimePicker is a widget used for selecting the time of the day in either AM/PM mode or 24 hours mode. The displayed time consist of hours, minutes and clock format. If we need to show this view as a Dialog then we have to use a TimePickerDialog class.

Methods of TimePicker:

1. setCurrentHour(Integer currentHour): This method is used to set the current hours in a time picker.

setHour(Integer hour): setCurrentHour() method was deprecated in API level 23. From api level 23 we have to use setHour(Integer hour). In this method there is only one parameter of integer type which is used to set the value for hours.

TimePicker simpleTimePicker=(TimePicker)findViewById(R.id.simpleTimePicker);

simpleTimePicker.setCurrentHour(5);

simpleTimePicker.setHour(5);

2. setCurrentMinute(Integer currentMinute): This method is used to set the current minutes in a time picker. setMinute(Integer minute): setCurrentMinute() method was deprecated in API level 23. From api level 23 we have to use setMinute(Integer minute). In this method there is only one parameter of integer type which set the value for minutes.

TimePicker simpleTimePicker=(TimePicker)findViewById(R.id.simpleTimePicker);

simpleTimePicker.setCurrentMinute(35);

simpleTimePicker.setMinute(35);

4. getCurrentMinute(): This method is used to get the current minutes from a time picker.

two methods) 2 M

		getMinute(): getCurrentMinute() method was deprecated in API level 23. From api level 23 we have to use getMinute(). This method returns an integer value.
		TimePicker simpleTimePicker = (TimePicker)findViewById(R.id.simpleTimePicker);
		int minutes = simpleTimePicker.getCurrentMinute();
		int minutes = simpleTimePicker.getMinute();
		5. setIs24HourView(Boolean is24HourView): This method is used to set the mode of the Time picker either 24 hour mode or AM/PM mode. In this method we set a Boolean value either true or false. True value indicate 24 hour mode and false value indicate AM/PM mode.
		TimePicker simpleTimePicker = (TimePicker)findViewById(R.id.simpleTimePicker);
		simpleTimePicker.setIs24HourView(true);
		6. is24HourView(): This method is used to check the current mode of the time picker. This method returns true if its 24 hour mode or false if AM/PM mode is set.
		TimePicker simpleTimePicker = (TimePicker)findViewById(R.id.simpleTimePicker);
		Boolean mode=simpleTimePicker.is24HourView();
		7.setOnTimeChangedListener(TimePicker.OnTimeChangedListener
		onTimeChanged()): This method is used to set the callback that indicates the time has been adjusted by the user.
		onTimeChanged(TimePicker view, int hourOfDay, int minute) is an override function of this listener in which we have three parameters first is for TimePicker, second for getting hour of the day and last is for getting the minutes after changing the time of the time picker.
		TimePicker simpleTimePicker = (TimePicker)findViewById(R.id.simpleTimePicker);
		simpleTimePicker.setOnTimeChangedListener(new TimePicker.OnTimeChangedListener() {
		@Override
		public void onTimeChanged(TimePicker view, int hourOfDay, int minute) {
		}
		});	

	b)	**Explain the steps to install and configure Android studio and SDK.**	**4 M**
	Ans	• Download the latest version of Android Studio from above URL and launch Android Studio.exe file by double clicking on it. • The initial android studio setup screen will open in that click Next to continue for further steps of environment setup • Now we need to select a required components to setup an android environment. Here we selected all three components (Android Studio, Android SDK and Android Virtual Device) and click Next. • Now we need to agree the License agreements to proceed further, click on I Agree button • Now we need to specify the local machine drive location to install Android Studio and Android SDK. • After selecting the location path to install required components, click Next. • Now select the start menu folder to create a shortcut for android studio and click Install • Once we click Install button the installation process will start and click Next after completion of Installation. • After that it will show installation completion wizard in that click Finish to launch android studio While launching Android Studio it will give you an option to import	**4 M for steps**

		settings from previous version of studio. In case if you don't have any previous version, select second option and click OK • Now android studio will open a welcome wizard window in that click Next to validate our current Android SDK and development environment setup • Now select a Standard installation type and click Next to install a common settings and options • Now verify settings and click Finish to complete android studio setup process • After completion of required components installation click on Finish • After completion of all required components installation, we will be able to see Android Studio welcome window	
	c)	**Explain the activity life cycle.**	**4 M**
	Ans	 Activities have a predefined life-cycle methods as follows:	**Diagram 2 M explanation 2 M**

		onCreate (): Called then the activity is created. Used to initialize the activity, for example create the user interface. **onStart ():**called when activity is becoming visible to the user. **onResume ():** Called if the activity get visible again and the user starts interacting with the activity again. Used to initialize fields, register listeners, bind to services, etc. **onPause ():** Called once another activity gets into the foreground. Always called before the activity is not visible anymore. Used to release resources or save application data. For example you unregister listeners, intent receivers, unbind from services or remove system service listeners. **onStop ():** Called once the activity is no longer visible. Time or CPU intensive shutdown operations, such as writing information to a database should be down in the onStop() method. This method is guaranteed to be called as of API 11. **onDestroy ():** called before the activity is destroyed.	
	d)	**Explain the steps to deploy app on Google Play Store.**	**4 M**
	Ans	**Step 1: Create a Developer Account** Before you can publish any app on Google Play, you need to create a Developer Account. You can easily sign up for one using your existing Google Account. You'll need to pay a one-time registration fee of $25 using your international credit or debit card. It can take up to 48 hours for your registration to be fully processed. **Step 2: Plan to Sell? Link Your Merchant Account** If you want to publish a paid app or plan to sell in-app purchases, you need to create a payments center profile, i.e. a merchant account. A merchant account will let you manage your app sales and monthly payouts, as well as analyze your sales reports right in your Play Console. **Step 3: Create an App** After creating application by clicking on 'Create Application'. Here you have to select your app's default language from the drop-down menu and then type in a title for your app. The title of your app will show on Google Play after you've published. **Step 4: Prepare Store Listing** Before you can publish your app, you need to prepare its store listing. These are all the details that will show up to customers on your app's listing on Google Play. You not necessarily complete it at once, you can always save a draft and revisit it later when you're ready to publish. The information required for your store listing is divided into several categories such as **Product Details** containing title, short and full description of the app, your app's title and description should be written with a great user experience in mind. Use the right keywords, but don't overdo it. Make sure your app doesn't come across as spam-y or promotional, or it will risk getting suspended on the Play Store.	**4 M for all steps**

Graphic Assets where you can add screenshots, images, videos, promotional graphics, and icons that showcase your app's features and functionality.

Languages & Translations, Categorization where in category can be selected to which your app belong to. **Contact Details, Privacy Policy** for apps that request access to sensitive user data or permissions, you need to enter a comprehensive privacy policy that effectively discloses how your app collects, uses, and shares that data.

Step 5: Upload APK to an App Release

Finally upload your app, by uploading APK file. Before you upload APK, you need to create an app release. You need to select the type of release you want to upload your first app version to. You can choose between an internal test, a closed test, an open test, and a production release. The first three releases allow you to test out your app among a select group of users before you make it go live for everyone to access.

This is a safer option because you can analyze the test results and optimize or fix your app accordingly if you need to before rolling it out to all users.

Once you create a production release, your uploaded app version will become accessible to everyone in the countries you choose to distribute it in and click on 'Create release.'

Step 6: Provide an Appropriate Content Rating

If you don't assign a rating to your app, it will be listed as 'Unrated'. Apps that are 'Unrated' may get removed from Google Play.

To rate your app, you need to fill out a content rating questionnaire An appropriate content rating will also help you get to the right audience, which will eventually improve your engagement rates.

Step 7: Set Up Pricing & Distribution

Before you can fill out the details required in this step, you need to determine your app's monetization strategy. Once you know how your app is going to make money, you can go ahead and set up your app as free or paid.

You can always change your app from paid to free later, but you cannot change a free app to paid. For that, you'll need to create a new app and set its price.

Step 8: Rollout Release to Publish Your App

The final step involves reviewing and rolling out your release after making sure you've taken care of everything else.

Before you review and rollout your release, make sure the store listing, content rating, and pricing and distribution sections of your app each have a green check mark next to them.

Once you're sure about the correctness of the details, select your app and navigate to 'Release management' – 'App releases.' You can always opt for reviews by clicking on 'Review' to be taken to the 'Review and rollout release' screen. Here, you can see if there are any issues or warnings you might have missed out on.

Finally, select 'Confirm rollout.' This will also publish your app to all users in your target countries on Google Play.
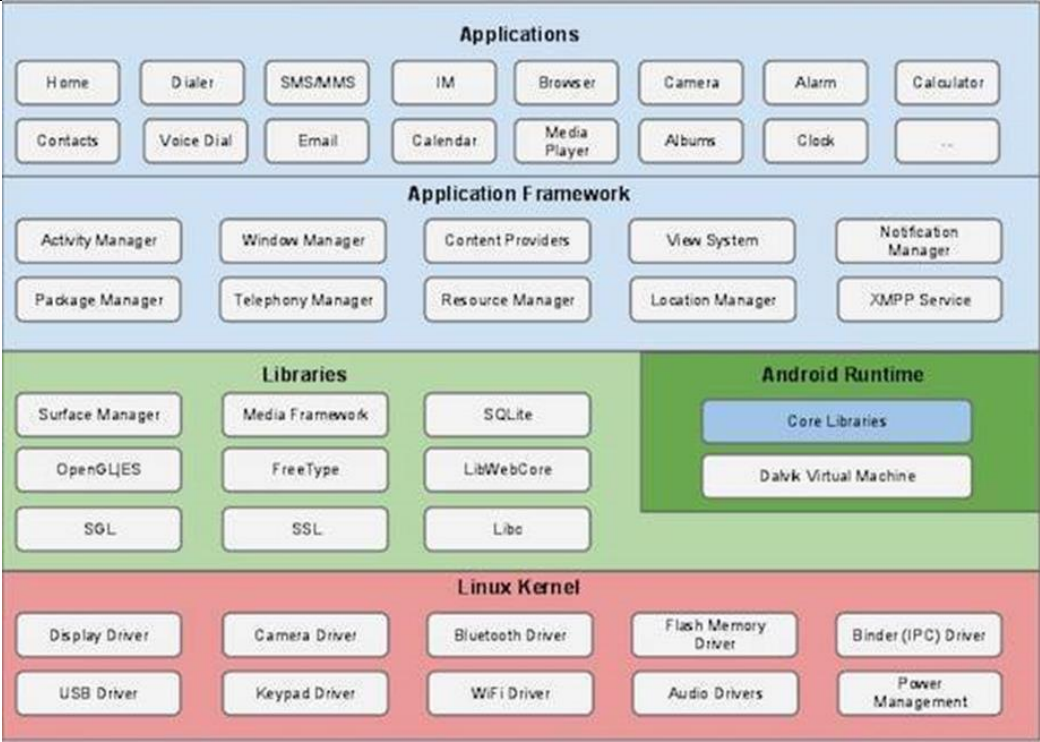
4.		Attempt any **THREE** of the following:	12 M
	a)	**Explain the fundamentals of Ul design in detail.**	4 M
	Ans	**1. Views:** • The basic building block for user interface in Android is a View, which is created from the View class and occupies a rectangular area on the screen and is responsible for drawing and event handling. • Views are the base class for all visual interface elements (commonly known as controls or widgets). All user interface UI controls, including the layout classes, are derived from View. • A View is an object/widget that draws something on the sreen by the help of user interact. • Examples of widgets are buttons, text boxes, labels etc. **2. View Groups:** • The ViewGroup is a subclass of View and provides invisible container that hold other Views or other ViewGroups and define their layout properties. • View groups are extensions of the View class that can contain multiple child Views. In order to Extend the ViewGroup class to create compound controls made up of interconnected child views. • A ViewGroup provides the layout in which we can order the appearance and sequence of views. Examples of ViewGroup are FrmaeLayout, LineourLayout etc. **3. Fragments:** • Fragments represents a portion of user interface in an Activity. Fragments, introduced in Android 3.0 which uses API level 11, are used to encapsulate portions of your UI. This encapsulation makes fragments particularly useful when optimizing our UI layouts for different screen sizes and creating reusable user interface (UI) elements. • Each Fragment includes its own user interface (UI) layout and receives the related input events but is tightly bound to the activity into which each must be embedded. Fragments are similar to UI view controller in iPhone development. **4. Activities:** • Activities dictate the UI and handle the user interaction to the smart phone screen. Activities represent a single screen that user interact. • Activities are the Android equivalent of Forms in traditional windows desktop development. To display a UI we assign a View (Usually a layout or Fragment) to an Activity	1 M for each
	b)	**Write a program to display a rectangular progress bar.**	4 M

	Ans	**Xml File:**
		<?xml version="1.0" encoding="utf-8"?>
		<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
		   xmlns:tools="http://schemas.android.com/tools"
		   android:layout_width="match_parent"
		   android:layout_height="match_parent"
		   tools:context=".MainActivity">
			xml file 2M,
			
			java file 2 M

```xml
<ProgressBar
    android:id="@+id/progressBar"
    style="@android:style/Widget.ProgressBar.Horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_marginTop="20dp"
    android:indeterminate="false"
    android:max="100"
    android:minHeight="50dp"
    android:minWidth="200dp"
    android:progress="1" />
<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/progressBar"
    android:layout_below="@+id/progressBar"/>
</RelativeLayout>
```

Java file:

```java
package in.edu.vpt.progressbar;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.os.Handler;
import android.os.Looper;
import android.widget.ProgressBar;
import android.widget.TextView;
public class MainActivity extends AppCompatActivity {
    private ProgressBar progressBar;
    private int progressStatus = 0;
    private TextView textView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        progressBar = (ProgressBar) findViewById(R.id.progressBar);
        textView = (TextView) findViewById(R.id.textView);
        // Start long running operation in a background thread
        new Thread(new Runnable() {
            public void run() {
```

		``java
		        while (progressStatus < 100) {
		          progressStatus += 1;
		          new Handler(Looper.getMainLooper()).post(new Runnable() {
		            public void run() {
		              progressBar.setProgress(progressStatus);
		              textView.setText(progressStatus+"/"+progressBar.getMax());
		            }
		          });
		          try {
		            // Sleep for 200 milliseconds.
		            Thread.sleep(200);
		          } catch (InterruptedException e) {
		            e.printStackTrace();
		          }
		        }
		      }
		    }).start();
		  }
		}
		``	

c)		**Explain Android system architecture in detail.**	**4 M**
	Ans	 **1. Applications:** • The top layer of android architecture is Applications. The native and third party applications like Contacts, Email, Music, Gallery, Clock, Games, etc. whatever we will build those will be installed on this layer only.	**Diagram 2 M,** **Explanation 2 M**

		• The application layer runs within the Android run time using the classes and services made available from the application framework. **2. Application Framework:** • The Application Framework provides the classes used to create an Android application. It also provides a generic abstraction for hardware access and manages the user interface and application resources. • It basically provides the services through which we can create the particular class and make that class helpful for the Applications creation. • The application framework includes services like telephony service, location services, notification manager, NFC service, view system, etc. which we can use for application development as per our requirements. **3. Android Runtime:** • Android Runtime environment is an important part of Android rather than an internal part and it contains a components like core libraries and the Dalvik virtual machine. • The Android run time is the engine that powers our applications along with the libraries and it forms the basis for the application framework. (i) Dalvik Virtual Machine (DVM) is a register-based virtual machine like Java Virtual Machine (JVM). It is specially designed and optimized for android to ensure that a device can run multiple instances efficiently. It relies on the Linux kernel for threading and low-level memory management. (ii) The core libraries in android runtime will enable us to implement an android applications using standard JAVA programming language. **4. Platform Libraries:** • The Platform Libraries includes various C/C++ core libraries and Java based libraries such as SSL, libc, Graphics, SQLite, Webkit, Media, Surface Manger, OpenGL etc. to provide a support for android development. **5. Linux Kernel:** • Linux Kernel is a bottom layer and heart of the android architecture. It is heart of Android architecture that exists at the root of android architecture and contains all the low-level device drivers for the various hardware components of an Android device. • Linux Kernel is responsible for device drivers, power management, memory management, device management and resource access. It manage all the drivers such as display drivers, camera drivers, Bluetooth drivers, audio drivers, memory drivers, etc. which are mainly required for the android device during the runtime. • The Linux Kernel will provide an abstraction layer between the device hardware and the remainder of the stack. It is responsible for memory management, power management, device management, resource access, etc.	
	d)	**Explain the procedure of Geo-coding and reverse Geo-coding.**	**4 M**
	Ans	**Geo-Coding:** • If we know the latitude and longitude of a location, we can find out its address using a process known as Geocoding. Google Maps in Android supports this via the Geocoder class.	**geocoding 2 M**

		• The following code shows how we can find out the address of a location we have just touched using the getFromLocation() method: ```\nclassMapOverlay extends com.google.android.maps.Overlay\n{\n@Override\npublicboolean draw(Canvas canvas, MapViewmapView,\nboolean shadow, long when)\n{\n//...\n}\n@Override\npublicbooleanonTouchEvent(MotionEvent event, MapViewmapView)\n{\n//---when user lifts his finger---\nif (event.getAction() == 1) {\nGeoPoint p = mapView.getProjection().fromPixels(\n(int) event.getX(),\n(int) event.getY());\nGeocoder geoCoder = new Geocoder(\ngetBaseContext(), Locale.getDefault());\ntry {\nList<Address> addresses = geoCoder.getFromLocation(\np.getLatitudeE6() / 1E6,\np.getLongitudeE6() / 1E6, 1);\nString add = "";\nif (addresses.size() > 0)\n{\nfor (inti=0; i<addresses.get(0).getMaxAddressLineIndex();\ni++)\nadd += addresses.get(0).getAddressLine(i) + "n";\n}\nToast.makeText(getBaseContext(), add, Toast.LENGTH_SHORT).show();\n}\ncatch (IOException e) {\ne.printStackTrace();\n}\nreturn true;\n}\nelse\nreturn false;\n}\n}\n```	**and reverse geocoding 2 M**

		Reverse-geocoding:
		If we know the address of a location but want to know its latitude and longitude, we can do so via reverse-Geocoding. Again, we can use the Geocoder class for this purpose.
		• The following code shows how we can find the exact location of the Empire State Building by using the getFromLocationName() method:
		Geocoder geoCoder = new Geocoder(this, Locale.getDefault());
		try {
		List<Address> addresses = geoCoder.getFromLocationName(
		"empire state building", 5);
		String add = "";
		if (addresses.size() > 0) {
		p = new GeoPoint(
		(int) (addresses.get(0).getLatitude() * 1E6),
		(int) (addresses.get(0).getLongitude() * 1E6));
		mc.animateTo(p);
		mapView.invalidate();
		}
		} catch (IOException e) {
		e.printStackTrace();
		}
		Once. the location is found, the above code navigates the map to the location.	

	e)	**Develop an application to send and receive SMS (Write only Java and permission tag in manifest file).**	**4 M**

	Ans	**Permissions and <receiver> tag required in AndroidManifest.xml**
		<uses-permission android:name="android.permission.RECEIVE_SMS" />
		<uses-permission android:name="android.permission.SEND_SMS"/>
		<uses-permission android:name="android.permission.READ_SMS"/>
		<uses-permission android:name="android.permission.WRITE_SMS"/>
		
		<receiver
		  android:name=".SmsReceiver"
		  android:enabled="true"
		  android:exported="true">
		  <intent-filter>
		    <action android:name="android.provider.Telephony.SMS_RECEIVED" />
		  </intent-filter>
		</receiver>
		
		MainActivity.java
		(Cosidering appropriate layout file with 2 edit text boxes namely for phone number, message and a button for sending sms)
		
		package com.example.testreceivesms;
		import androidx.appcompat.app.AppCompatActivity;
		import androidx.core.app.ActivityCompat;
			Manifest File:1M
			
			Java Code: 3M

```java
import androidx.core.content.ContextCompat;
import android.Manifest;
import android.content.IntentFilter;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.telephony.SmsManager;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
SmsReceiver sms= new SmsReceiver();
EditText et1,et2;
    Button b1;
    @Override
    protected void onCreate (Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        et1=findViewById(R.id.etPhno);
        et2=findViewById(R.id.etmsg);
        b1=findViewById(R.id.btnSms);
if(ContextCompat.checkSelfPermission(MainActivity.this,Manifest.permission.SEND_S
MS)!=
            PackageManager.PERMISSION_GRANTED)
    {
        ActivityCompat.requestPermissions(MainActivity.this,new
            String[]{Manifest.permission.SEND_SMS},100);
    }
    b1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            try {
                String phno= et1.getText().toString();
                String msg=et2.getText().toString();
                SmsManager smsManager= SmsManager.getDefault();
                smsManager.sendTextMessage(phno,null,msg,null,null);
Toast.makeText(MainActivity.this,"Sms sent successfully",
Toast.LENGTH_LONG).show();
            }
            catch(Exception e)
            {
                Toast.makeText(MainActivity.this,"Sms failed to send... try again",
Toast.LENGTH_LONG).show();
            }
        }
    });
    }
    @Override
```

```
        protected void onStart() {
          super.onStart();
IntentFilter filter=new  IntentFilter("android.provider.Telephony.SMS_RECEIVED");
          registerReceiver(sms,filter);
        }

        @Override
        protected void onStop() {
          super.onStop();
          unregisterReceiver(sms);
        }
}
```

SmsReceiver.java
```
package com.example.testreceivesms;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.telephony.SmsMessage;
import android.widget.Toast;

public class SmsReceiver extends BroadcastReceiver {
    SmsReceiver(){}
    @Override
    public void onReceive(Context context, Intent intent) {
        Bundle bundle = intent.getExtras();

        if (bundle != null) {
            // Retrieve the SMS Messages received
            Object[] sms = (Object[]) bundle.get("pdus");

            // For every SMS message received
            for (int i=0; i < sms.length; i++) {
                // Convert Object array
                SmsMessage smsMessage = SmsMessage.createFromPdu((byte[]) sms[i]);

                String phone = smsMessage.getOriginatingAddress();
                String message = smsMessage.getMessageBody().toString();

                Toast.makeText(context, "Received from "+ phone + ": " + message,
Toast.LENGTH_SHORT).show();
            }
        }
    }
}
```

5.		**Attempt any TWO of the following:**	**12 M**

a)	Develop a program to send and receive an Email.	6 M
Ans	**Program to send Email** **activity_main.xml**	**Any correct logic program should be considered, 3M for send logic, 3M for receive logic**

Program to send Email

activity_main.xml

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

    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignParentRight="true"
        android:layout_marginTop="18dp"
        android:layout_marginRight="22dp" />

    <EditText
        android:id="@+id/editText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editText1"
        android:layout_alignLeft="@+id/editText1"
        android:layout_marginTop="20dp" />

    <EditText
        android:id="@+id/editText3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editText2"
        android:layout_alignLeft="@+id/editText2"
        android:layout_marginTop="30dp" />

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/editText1"
        android:layout_alignBottom="@+id/editText1"
        android:layout_alignParentLeft="true"
        android:text="Send To:"
        android:textColor="#0F9D58" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
```

```xml
                    android:layout_height="wrap_content"
                    android:layout_alignBaseline="@+id/editText2"
                    android:layout_alignBottom="@+id/editText2"
                    android:layout_alignParentLeft="true"
                    android:text="Email Subject:"
                    android:textColor="#0F9D58" />

            <TextView
                    android:id="@+id/textView3"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_alignBaseline="@+id/editText3"
                    android:layout_alignBottom="@+id/editText3"
                    android:text="Email Body:"
                    android:textColor="#0F9D58" />

            <Button
                    android:id="@+id/button"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_below="@+id/editText3"
                    android:layout_alignLeft="@+id/editText3"
                    android:layout_marginLeft="76dp"
                    android:layout_marginTop="20dp"
                    android:text="Send email!!" />
        </RelativeLayout>
```

MainActivity.java

```java
    import android.content.Intent;
    import android.os.Bundle;
    import android.widget.Button;
    import android.widget.EditText;
    import androidx.appcompat.app.AppCompatActivity;
    public class MainActivity extends AppCompatActivity {
        // define objects for edit text and button
        Button button;
        EditText sendto, subject, body;

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

                // Getting instance of edittext and button
                sendto = findViewById(R.id.editText1);
                subject = findViewById(R.id.editText2);
                body = findViewById(R.id.editText3);
                button = findViewById(R.id.button);
```

```
                // attach setOnClickListener to button with Intent object define in it
                button.setOnClickListener(view -> {
                        String emailsend = sendto.getText().toString();
                        String emailsubject = subject.getText().toString();
                        String emailbody = body.getText().toString();

                        // define Intent object with action attribute as ACTION_SEND
                        Intent intent = new Intent(Intent.ACTION_SEND);

                        // add three fields to intent using putExtra function
                        intent.putExtra(Intent.EXTRA_EMAIL, new String[]{emailsend});
                        intent.putExtra(Intent.EXTRA_SUBJECT, emailsubject);
                        intent.putExtra(Intent.EXTRA_TEXT, emailbody);

                        // set type of intent
                        intent.setType("message/rfc822");

                        // startActivity with intent with chooser as Email client using
createChooser function
                        startActivity(Intent.createChooser(intent, "Choose an Email client :"));
                });
        }
}
```

Program to receive Email
(Note: Receiving email is not the service of Android OS, instead it uses some third party applicaton like Gmail, so instead of receiving email in android app, a code which broadcasts message if email comes using broadcast receiver or any relevant logic can be considered.)

MainActivity.java

```
package com.example.myemailprog;
import androidx.appcompat.app.AppCompatActivity;
import android.content.IntentFilter;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {
GmailReceiver gml;
IntentFilter intf;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        gml = new GmailReceiver();
        intf = new IntentFilter("android.intent.action.VIEW");
    }

    @Override
    protected void onResume() {
        super.onResume();
```

```
        registerReceiver(gml, intf);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
       unregisterReceiver(gml);
    }
}
```

Gmailrerciever..java
```
package com.example.myemailprog;

import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.res.Resources;
import android.graphics.BitmapFactory;
import android.widget.Toast;

public class GmailReceiver extends BroadcastReceiver
 {
    @Override
    public void onReceive(Context context, Intent intent)
{
        Toast.makeText(context, "Email Received", Toast.LENGTH_LONG).show();


    }
}
```

<u>AndroidManifest.xml</u>
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.myemailprog">
    <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.MyEmailProg">
      <activity
        android:name=".MainActivity"
        android:exported="true">
        <intent-filter>
          <action android:name="android.intent.action.MAIN" />
          <category android:name="android.intent.category.LAUNCHER" />
```

_____

		`</intent-filter>`
		`</activity>`
		`<receiver android:name="GmailReceiver"`
		  `android:exported="false">`
		  `<intent-filter>`
		    `<action android:name="android.intent.action.PROVIDER_CHANGED"`
		      `android:priority="-10">`
		    `</action>`
		    `<action android:name="android.intent.action.VIEW" />`
		    `<data android:scheme="content" android:host="gmail-ls"`
		      `android:pathPattern="/unread/.*">`
		    `</data>`
		  `</intent-filter>`
		  `</receiver>`
		  `</application>`
		`</manifest>`	

b)		**Develop a program for providing bluetooth connectivity.**	**6 M**

	Ans	**activity_main.xml**
		
		``xml
		<RelativeLayout xmlns:androclass="http://schemas.android.com/apk/res/android"
		    xmlns:tools="http://schemas.android.com/tools"
		    android:layout_width="match_parent"
		    android:layout_height="match_parent"
		    tools:context=".MainActivity" >
		
		    <TextView android:text=""
		     android:id="@+id/out"
		     android:layout_width="wrap_content"
		     android:layout_height="wrap_content">
		    </TextView>
		    <Button
		
		    android:id="@+id/button1"
		        android:layout_width="wrap_content"
		        android:layout_height="wrap_content"
		        android:layout_alignParentLeft="true"
		        android:layout_alignParentTop="true"
		        android:layout_marginLeft="30dp"
		        android:layout_marginTop="49dp"
		        android:text="TURN_ON" />
		
		    <Button
		        android:id="@+id/button2"
		        android:layout_width="wrap_content"
		        android:layout_height="wrap_content"
		        android:layout_alignLeft="@+id/button1"
		        android:layout_below="@+id/button1"
		        android:layout_marginTop="27dp"
		        android:text="DISCOVERABLE" />
		``
			Layout file : 2M
			
			Java File : 3M
			
			Manifest file : 1M

```xml
    <Button
        android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/button2"
        android:layout_below="@+id/button2"
        android:layout_marginTop="28dp"
        android:text="TURN_OFF" />

</RelativeLayout>
```

AndroidManifest.xml
```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:androclass="http://schemas.android.com/apk/res/android"
    package="com.example.bluetooth"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="16" />
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.bluetooth.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

MainActivity.java
```java
package com.example.bluetooth;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.content.Context;
import android.content.Intent;
```

```java
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity {
    private static final int REQUEST_ENABLE_BT = 0;
    private static final int REQUEST_DISCOVERABLE_BT = 0;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    final TextView out=(TextView)findViewById(R.id.out);
    final Button button1 = (Button) findViewById(R.id.button1);
    final Button button2 = (Button) findViewById(R.id.button2);
    final Button button3 = (Button) findViewById(R.id.button3);
    final BluetoothAdapter mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    if (mBluetoothAdapter == null) {
      out.append("device not supported");
    }
    button1.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            if (!mBluetoothAdapter.isEnabled()) {
Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
            }
        }
    });
    button2.setOnClickListener(new View.OnClickListener() {
     @Override
public void onClick(View arg0) {
        if (!mBluetoothAdapter.isDiscovering()) {
            //out.append("MAKING YOUR DEVICE DISCOVERABLE");
    Toast.makeText(getApplicationContext(), "MAKING YOUR DEVICE DISCOVERA
BLE",
    Toast.LENGTH_LONG);

Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVER
ABLE);
            startActivityForResult(enableBtIntent, REQUEST_DISCOVERABLE_BT);

        }
    }
    });
    button3.setOnClickListener(new View.OnClickListener() {
@Override
    public void onClick(View arg0) {
```
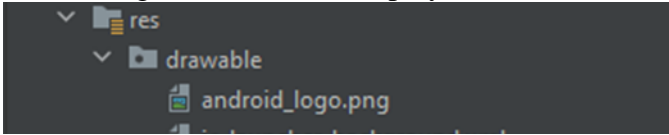
| | | ```
                mBluetoothAdapter.disable();
                //out.append("TURN_OFF BLUETOOTH");
Toast.makeText(getApplicationContext(), "TURNING_OFF BLUETOOTH", Toast.LENGTH_LONG);
              }
            });
          }
          @Override
            public boolean onCreateOptionsMenu(Menu menu) {
          // Inflate the menu; this adds items to the action bar if it is present.
          getMenuInflater().inflate(R.menu.activity_main, menu);
              return true;
            }

          }
``` | |
| **c)** | | **Develop a program to implement**

i)   List view of 5 items
ii)   Grid view of 4 x 4 items
iii)   Image view. | **6 M** |
| | **Ans** | ```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <ListView
        android:id="@+id/sample_list"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
    </ListView>

    <GridView
        android:id="@+id/gridview1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:columnWidth="50dp"
        android:gravity="center"
        android:numColumns="auto_fit"
        android:stretchMode="columnWidth" >
    </GridView>

    <ImageView
        android:id="@+id/full_logo"
        android:layout_width="match_parent"
``` | **Correct Logics for**<br><br>**ListView : 2M**<br><br>**GridView : 2M**<br><br>**ImageView : 2M** |

```
                android:layout_height="wrap_content"
                app:srcCompat="@drawable/android_logo" />

</LinearLayout>
```

Placed image that has to be displayed in drawable folder

```
 ∨  res
      ∨  drawable
            android_logo.png
```

```java
package in.msbte.controls_exam_ques;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.GridView;
import android.widget.ListView;

public class MainActivity extends AppCompatActivity {
    String[] sampleArray = {"Item 1","Item 2","Item 3","Item 4", "Item 5"};
    GridView gridView;
    static final String[] alphabets = new String[]{
        "A", "B", "C", "D", "E",
        "F", "G", "H", "I", "J",
        "K", "L", "M", "N", "O",
        "P", "Q", "R", "S", "T",
        "U", "V", "W", "X", "Y", "Z"
    };
    ArrayAdapter adapter, adapter1;

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

        //List View
        adapter = new ArrayAdapter<String>(this, R.layout.simple_item, sampleArray);
        ListView listView = (ListView) findViewById(R.id.sample_list);
        listView.setAdapter(adapter);

        //Grid View
        gridView = (GridView) findViewById(R.id.gridview1);
        adapter1 = new ArrayAdapter<String>(this, R.layout.simple_item, alphabets);
        gridView.setAdapter(adapter1);

    }
}
```

6.		**Attempt any <u>TWO</u> of the following:**	**12 M**

a)	Develop an application to store customer's details like, customer-id, customer-name, mobile number, address, pin-code and retrieve customer information using customer-id in SQLite databases. (Any other relevant logic can be considered)	6 M
Ans	**activity_main.xml**	Layout File : 1M

activity_main.xml

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

    <TextView
        android:text="Insert Customer Details"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:id="@+id/textView"
        android:gravity="center"
        android:textSize="20dp"
        android:textColor="#000000"/>

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="ID"
        android:id="@+id/editid"
        android:layout_below="@+id/textView"/>

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="Name"
        android:id="@+id/editname"
        android:layout_below="@+id/editid"/>

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="Mobile No."
        android:id="@+id/editmobile"
        android:layout_below="@+id/editname"/>
```

Java File :

Correct Logics for

Create table : 1M
Insertion of record : 2M
Retrieval of data : 2M

```xml
<EditText
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="Address"
    android:lines="3"
    android:id="@+id/editaddress"
    android:layout_below="@+id/editmobile"/>

<EditText
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="Pin Code"
    android:id="@+id/editpincode"
    android:layout_below="@+id/editaddress"/>

<Button
    android:text="Insert Data"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/editpincode"
    android:layout_centerHorizontal="true"
    android:id="@+id/button" />

<TextView
    android:text="Search Customer Details"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="30dp"
    android:layout_centerHorizontal="true"
    android:id="@+id/textView1"
    android:gravity="center"
    android:textSize="20dp"
    android:layout_below="@+id/button"
    android:textColor="#000000"/>

<EditText
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="Enter ID"
    android:id="@+id/editsearchid"
    android:layout_below="@+id/textView1"/>
```

```
    <Button
        android:text="Search Data"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editsearchid"
        android:layout_centerHorizontal="true"
        android:id="@+id/button1" />
</RelativeLayout>
```

MainActivity.java
```
package in.msbte.database;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
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 {
    SQLiteDatabase sqLiteDatabaseObj;
    EditText editTextID, editTextName, editMobileNo, editAddress, editPincode,
editSearchid;
    String cid, cname, cmobile, caddress, cpincode, sql_query, sid;
    Button EnterData, SearchData;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        EnterData = (Button)findViewById(R.id.button);
        SearchData = (Button)findViewById(R.id.button1);
        editTextID = (EditText)findViewById(R.id.editid);
        editTextName = (EditText)findViewById(R.id.editname);
        editMobileNo = (EditText)findViewById(R.id.editmobile);
        editAddress = (EditText)findViewById(R.id.editaddress);
        editPincode = (EditText)findViewById(R.id.editpincode);
        editSearchid = (EditText)findViewById(R.id.editsearchid);

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

```
                public void onClick(View view) {
                    sqLiteDatabaseObj = openOrCreateDatabase("AndroidJSonDataBase",
Context.MODE_PRIVATE, null);
                    sqLiteDatabaseObj.execSQL("CREATE TABLE IF NOT EXISTS
AndroidJSonTable(id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, cid
VARCHAR, name VARCHAR, mobile VARCHAR, address VARCHAR, pincode
VARCHAR);");
                    cid = editTextID.getText().toString();
                    cname = editTextName.getText().toString() ;
                    cmobile = editMobileNo.getText().toString();
                    caddress = editAddress.getText().toString();
                    cpincode = editPincode.getText().toString();
                    sql_query = "INSERT INTO AndroidJSonTable (cid, name, mobile, address,
pincode) VALUES('"+cid+"', '"+cname+"', '"+cmobile+"', '"+caddress+"', '"+cpincode+"');";
                    sqLiteDatabaseObj.execSQL(sql_query);
                    Toast.makeText(getApplicationContext(), "Data Inserted Successfully",
Toast.LENGTH_LONG).show();
                }
            });

        SearchData.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                sid = editSearchid.getText().toString();
                Cursor cursor =  sqLiteDatabaseObj.rawQuery( "select * from AndroidJSonTable
where cid="+sid+"", null );
                    StringBuffer buffer= new StringBuffer();
                    while (cursor.moveToNext())
                    {
                        String cid =cursor.getString(1);
                        String name =cursor.getString(2);
                        String mob =cursor.getString(3);
                        String addr =cursor.getString(4);
                        String pcode =cursor.getString(5);
                        buffer.append(cid+ "   " + name + "   " + mob +"   " + addr +"   " + pcode +"
\n");
                    Toast.makeText(getApplicationContext(), buffer,
Toast.LENGTH_LONG).show();
} }); } }
```

	b)	Write a program to find the direction from user's current location to MSBTE, Bandra. (Write only Java and manitest file). (Note : Any other relevant logic to get the required output can also be considered.)	6 M

Ans	**AndroidManifest.xml**	**Manifest file :2M**
| | ```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.msbte.google_map_currentlocationroute">

  <uses-permission android:name="android.permission.INTERNET" />
  <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
  <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
  <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">

    <meta-data
      android:name="com.google.android.gms.version"
      android:value="@integer/google_play_services_version" />
    <meta-data
      android:name="com.google.android.geo.API_KEY"
      android:value="@string/google_maps_key" />

    <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 : 4M** |
| | **MainActivity.java**<br><br>```java
import android.Manifest;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.net.Uri;
import android.provider.Settings;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
``` | |

```java
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.MapFragment;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.model.CameraPosition;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.maps.model.Polyline;
import com.google.android.gms.maps.model.PolylineOptions;
import com.karumi.dexter.Dexter;
import com.karumi.dexter.MultiplePermissionsReport;
import com.karumi.dexter.PermissionToken;
import com.karumi.dexter.listener.DexterError;
import com.karumi.dexter.listener.PermissionRequest;
import com.karumi.dexter.listener.PermissionRequestErrorListener;
import com.karumi.dexter.listener.multi.MultiplePermissionsListener;
import java.util.List;

public class MainActivity extends AppCompatActivity implements
GoogleApiClient.ConnectionCallbacks, GoogleApiClient.OnConnectionFailedListener,
com.google.android.gms.location.LocationListener , OnMapReadyCallback,
TaskLoadedCallback{

   //variables for map and route

    private GoogleMap mMap;
    private MarkerOptions place1, place2;
    Button getDirection;
    private Polyline currentPolyline;
    private MapFragment mapFragment;
    private boolean isFirstTime = true;

//variables for current location
    private static final String TAG = "MainActivity";

    private TextView tvLocation;
    private GoogleApiClient mGoogleApiClient;
    private Location mLocation;
    private LocationRequest mLocationRequest;
    private com.google.android.gms.location.LocationListener listener;
    private long UPDATE_INTERVAL = 2 * 1000;  /* 10 secs */
```

```java
private long FASTEST_INTERVAL = 2000; /* 2 sec */

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

    //code for getting current location
    requestMultiplePermissions();
    tvLocation = (TextView) findViewById((R.id.tv));
    mGoogleApiClient = new GoogleApiClient.Builder(this)
            .addConnectionCallbacks(this)
            .addOnConnectionFailedListener(this)
            .addApi(LocationServices.API)
            .build();
}

//code for drawing route

@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;
    mMap.clear();
    Log.d("mylog", "Added Markers");
    mMap.addMarker(place1);
    mMap.addMarker(place2);

    CameraPosition googlePlex = CameraPosition.builder()
            .target(new LatLng(22.7739,71.6673))
            .zoom(7)
            .bearing(0)
            .tilt(45)
            .build();

    mMap.animateCamera(CameraUpdateFactory.newCameraPosition(googlePlex), 5000,
null);
}

private String getUrl(LatLng origin, LatLng dest, String directionMode) {
    // Origin of route
    String str_origin = "origin=" + origin.latitude + "," + origin.longitude;
    // Destination of route
    String str_dest = "destination=" + dest.latitude + "," + dest.longitude;
    // Mode
    String mode = "mode=" + directionMode;
    // Building the parameters to the web service
    String parameters = str_origin + "&" + str_dest + "&" + mode;
    // Output format
    String output = "json";
```

```java
      // Building the url to the web service
      String url = "https://maps.googleapis.com/maps/api/directions/" + output + "?" +
parameters + "&key=" + getString(R.string.google_maps_key);
      return url;
    }


   @Override
   public void onTaskDone(Object... values) {
      if (currentPolyline != null)
         currentPolyline.remove();
      currentPolyline = mMap.addPolyline((PolylineOptions) values[0]);
   }


   //runtime permission method

   private void  requestMultiplePermissions(){
      Dexter.withActivity(this)
            .withPermissions(
                 Manifest.permission.ACCESS_FINE_LOCATION,
                 Manifest.permission.ACCESS_COARSE_LOCATION )
            .withListener(new MultiplePermissionsListener() {
               @Override
               public void onPermissionsChecked(MultiplePermissionsReport report) {
                  // check if all permissions are granted
                  if (report.areAllPermissionsGranted()) {
                     Toast.makeText(getApplicationContext(), "All permissions are granted by
user!", Toast.LENGTH_SHORT).show();
                  }

                  // check for permanent denial of any permission
                  if (report.isAnyPermissionPermanentlyDenied()) {
                     // show alert dialog navigating to Settings
                     openSettingsDialog();
                  }
               }

               @Override
               public void onPermissionRationaleShouldBeShown(List<PermissionRequest>
permissions, PermissionToken token) {
                  token.continuePermissionRequest();
               }
            }).
            withErrorListener(new PermissionRequestErrorListener() {
               @Override
               public void onError(DexterError error) {
                  Toast.makeText(getApplicationContext(), "Some Error! ",
Toast.LENGTH_SHORT).show();
               }
            })
```

```java
                    .onSameThread()
                    .check();
        }

    private void openSettingsDialog() {
        AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
        builder.setTitle("Required Permissions");
        builder.setMessage("This app require permission to use awesome feature. Grant them in
app settings.");
        builder.setPositiveButton("Take Me To SETTINGS", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                dialog.cancel();
                Intent intent = new
Intent(Settings.ACTION_APPLICATION_DETAILS_SETTINGS);
                Uri uri = Uri.fromParts("package", getPackageName(), null);
                intent.setData(uri);
                startActivityForResult(intent, 101);
            }
        });
        builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                dialog.cancel();
            }
        });
        builder.show();
    }

    //methods for getting current location
    @Override
    public void onConnected(Bundle bundle) {
        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
            return;
        }

        startLocationUpdates();
        mLocation = LocationServices.FusedLocationApi.getLastLocation(mGoogleApiClient);
        if(mLocation == null){
            startLocationUpdates();
        }
        if (mLocation != null) {
            // mLatitudeTextView.setText(String.valueOf(mLocation.getLatitude()));
            //mLongitudeTextView.setText(String.valueOf(mLocation.getLongitude()));
```

```
        } else {
            Toast.makeText(this, "Location not Detected", Toast.LENGTH_SHORT).show();
        }   }

    @Override
    public void onConnectionSuspended(int i) {
        Log.i(TAG, "Connection Suspended");
        mGoogleApiClient.connect();
    }
    @Override
    public void onConnectionFailed(ConnectionResult connectionResult) {
        Log.i(TAG, "Connection failed. Error: " + connectionResult.getErrorCode());
    }
    @Override
    protected void onStart() {
        super.onStart();
        if (mGoogleApiClient != null) {
            mGoogleApiClient.connect();
        }   }

    @Override
    protected void onStop() {
        super.onStop();
        if (mGoogleApiClient.isConnected()) {
            mGoogleApiClient.disconnect();
        }   }

    protected void startLocationUpdates() {
        // Create the location request
        mLocationRequest = LocationRequest.create()
            .setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY)
            .setInterval(UPDATE_INTERVAL)
            .setFastestInterval(FASTEST_INTERVAL);
        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
            return;
        }
        LocationServices.FusedLocationApi.requestLocationUpdates(mGoogleApiClient,
            mLocationRequest, this);
    }

    @Override
    public void onLocationChanged(Location location) {

        String msg = "Updated Location: " +
            Double.toString(location.getLatitude()) + "," +
```

_____

		Double.toString(location.getLongitude());
		
		 tvLocation.setText(String.valueOf(location.getLatitude() +"
		 "+String.valueOf(location.getLongitude()))));
		 Toast.makeText(this, msg, Toast.LENGTH_SHORT).show();
		
		 if(isFirstTime){
		 //code to draw path on map
		
		 getDirection = findViewById(R.id.btnGetDirection);
		 getDirection.setOnClickListener(new View.OnClickListener() {
		 @Override
		 public void onClick(View view) {
		 new FetchURL(MainActivity.this).execute(getUrl(place1.getPosition(),
		 place2.getPosition(), "driving"), "driving");
		 }
		 });
		
		 place1 = new MarkerOptions().position(new LatLng(location.getLatitude(),
		 location.getLongitude())).title("Location 1");
		 place2 = new MarkerOptions().position(new
		 LatLng(19.021824,72.8662016)).title("MSBTE");
		 mapFragment = (MapFragment)
		 getFragmentManager().findFragmentById(R.id.mapNearBy);
		 mapFragment.getMapAsync(this);
		 isFirstTime = false;
		 } }}	

c)		**Develop a simple calculator using relative layout.**	**6 M**
Ans		**activity_main.xml** <?xml version="1.0" encoding="utf-8"?> <RelativeLayout 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/heading" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text=" Calculator" android:layout_centerHorizontal="true" android:textSize="30dp" />	**Use of Relative Layout with appropriate attributes : 3M** **Logic for simple calculator with basic arithmetic operations : 3M**

```xml
<EditText
    android:id="@+id/num1"
    android:layout_below="@+id/heading"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Enter Number 1"
    android:inputType="number" />

<EditText
    android:id="@+id/num2"
    android:layout_below="@+id/num1"
    android:hint="Enter Number 2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="number" />

<TextView
    android:id="@+id/result"
    android:layout_below="@+id/num2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:layout_marginBottom="20dp"
    android:text="Result" />

<Button
    android:id="@+id/sum"
    android:layout_below="@id/result"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="10dp"
    android:text="+" />

<Button
    android:id="@+id/sub"
    android:layout_below="@id/result"
    android:layout_toRightOf="@id/sum"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="10dp"
    android:text="-" />

<Button
```

```
        android:id="@+id/div"
        android:layout_below="@id/result"
        android:layout_toRightOf="@id/sub"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="10dp"
        android:text="/" />

    <Button
        android:id="@+id/mul"
        android:layout_below="@id/result"
        android:layout_toRightOf="@id/div"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="10dp"
        android:text="x"/>
</RelativeLayout>
```

MainActivity.java
```
package in.msbte.calculator;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
public class MainActivity extends AppCompatActivity {
    public EditText e1, e2;
    Button add, sub , mul, div;
    TextView t1;
    int num1, num2;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        e1 = (EditText) findViewById(R.id.num1);
        e2 = (EditText) findViewById(R.id.num2);
        t1 = (TextView) findViewById(R.id.result);
        add = (Button) findViewById(R.id.sum);
        mul = (Button) findViewById(R.id.mul);
        div = (Button) findViewById(R.id.div);
        sub = (Button) findViewById(R.id.sub);
        add.setOnClickListener(new View.OnClickListener() {
```

```
        @Override
        public void onClick(View view) {
           num1 = Integer.parseInt(e1.getText().toString());
           num2 = Integer.parseInt(e2.getText().toString());
           int sum = num1 + num2;
           t1.setText(Integer.toString(sum));
        }
     });
     sub.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
           num1 = Integer.parseInt(e1.getText().toString());
           num2 = Integer.parseInt(e2.getText().toString());
           int sum = num1 - num2;
           t1.setText(Integer.toString(sum));
        }
     });
     mul.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
           num1 = Integer.parseInt(e1.getText().toString());
           num2 = Integer.parseInt(e2.getText().toString());
           int sum = num1 * num2;
           t1.setText(Integer.toString(sum));
        }
     });
     div.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
           num1 = Integer.parseInt(e1.getText().toString());
           num2 = Integer.parseInt(e2.getText().toString());
           int sum = num1 / num2;
           t1.setText(Integer.toString(sum));
  }
 });
}
}
```