CMR
INSTITUTE OF
TECHNOLOGY

USN | | | | | | | | | | |

Internal Assessment Test 2 – Nov. 2020

Sub:	Mobile Applications							Sub Code:	18MCA52
Date:	3/11/2020	Duration:	90 min's	Max Marks:	50	Sem & Sec:	V A & B	Branch:	MCA

Note : Answer any full FIVE Questions

	PART I	MARKS	OBE	
			CO	RBT
1.	What are the different methods for getting location data? Explain	[10]	CO4	L1
2	Discuss APK file deployment in detail.	[10]	CO5	L2
3	What is the process for binding activities to services in android application?	[10]	CO5	L2
4..a.	What are notifications? Explain various types of notifications.	[6]	CO4	L2
b.	Write a note on Date Picker rand Time Picker Views	[4]	CO4	L1
5.	Write in detail how to display Google maps in your own android applications.	[10]	CO1	L2
6.	Define service. How do you create your own service in Android? Explain with a snippet code.	[10]	CO4	L1,L2
7.	Develop a mobile application to list the tourist places of Karnataka using List View.	[10]	CO5	L3
8.	Devise an application that draws basic graphical primitives (rectangle, circle) on the screen.	[10]	CO5	L3

1.What are the different methods for getting location data? Explain. (10 Marks)

Nowadays, mobile devices are commonly equipped with **GPS receivers.**

o Because of the many satellites orbiting the earth, you can use a GPS receiver to find your location easily. However, GPS requires a clear sky to work and hence does not always work indoors or where satellites can't penetrate (such as a tunnel through a mountain).

 Another effective way to locate your position is through ***cell tower triangulation***.

o When a mobile phone is switched on, it is constantly in contact with base stations surrounding it.

o By knowing the identity of cell towers, it is possible to translate this information into a physical location through the use of various databases containing the cell towers' identities and their exact geographical locations.

o The advantage of cell tower triangulation is that it works indoors, without the need to obtain information from satellites.

o It is not as precise as GPS because its accuracy depends on overlapping signal coverage, which varies quite a bit.

o Cell tower triangulation works best in densely populated areas where the cell towers are closely located.
 A third method of locating your position is to rely on **Wi-Fi triangulation.**
o Rather than connect to cell towers, the device connects to a Wi-Fi network and checks the service provider against databases to determine the location serviced by the provider.On the Android, the SDK provides the LocationManager class to help your device determine the user's physical location.
lm.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0, locationListener);
This method takes four parameters:
1. Provider -The name of the provider with which you register. In this case, you are using GPS to obtain your geographical location data.
2. minTime - The minimum time interval for notifications, in milliseconds.
3. minDistance - The minimum distance interval for notifications, in meters.
4. Listener - An object whose onLocationChanged() method will be called for each location update.

2. Discuss APK file deployment in detail. (10 Marks)

Once you have signed your APK fi les, you need a way to get them onto your users' devices. Three methods are here:
 Deploying manually using the adb.exe tool
 Hosting the application on a web server
 Publishing through the Android Market
Besides the above methods, you can install your applications on users' devices through emails,SD card, etc. As long as you can transfer the APK file onto the user's device, you can install the application.
Using the adb.exe Tool: Once your Android application is signed, you can deploy it toemulators and devices using the adb.exe (Android Debug Bridge) tool (located in theplatform-tools folder of the Android SDK). Using the command prompt in Windows,navigate to the "<Android_SDK>\platform-tools" folder. To install the application to anemulator/device (assuming the emulator is currently up and running or a device is currentlyconnected), issue the following command:
adb install "C:\Users\Wei-Meng Lee\Desktop\LBS.apk"
(Note that, here, LBS is name of the project)
Besides using the adb.exe tool to install applications, you can also use it to remove an installed application. To do so, you can use the shell option to remove an application from its installed folder:
adb shell rm /data/app/net.learn2develop.LBS.apk
Another way to deploy an application is to use the DDMS tool in Eclipse. With an emulator (or device) selected, use the File Explorer in DDMS to go to the /data/app folder and use the "Push a file onto the device" button to copy the APK file onto the device.
Using a Web Server: If you wish to host your application on your own, you can use a web server to do that. This is ideal if you have your own web hosting services and want to provide the application free of charge to your users or you can restrict access to certain groups of people. Following are the steps involved:
 Copy the signed LBS.apk fi le to c:\inetpub\wwwroot\. In addition, create a new HTML file named Install.html with the following content:

```
<html>
<title>Where Am I application</title>
<body>
Download the Where Am I application <a href="LBS.apk">here</a>
</body>
</html>
```

 On your web server, you may need to register a new MIME type for the APK file. The MIME type for the .apk extension is application/vnd.android.packagearchive.

 From the Application settings menu, check the “Unknown sources” item. You will be prompted with a warning message. Click OK. Checking this item will allow the Emulator/device to install applications from other non-Market sources (such as from a web server).
 To install the LBS.apk application from the IIS web server running on your computer, launch the Browser application on the Android Emulator/device and navigate to the URL pointing to the APK file. To refer to the computer running the emulator, you should use the special IP address of 10.0.2.2.
 Alternatively, you can also use the IP address of the host computer. Clicking the “here” link will download the APK file onto your device. Drag the notification bar down to reveal the download status. To install the downloaded application, simply tap on it and it will show the permission(s) required by this application.
 Click the Install button to proceed with the installation. When the application is installed, you can launch it by clicking the Open button.Besides using a web server, you can also e-mail your application to users as an attachment; when the users receive the e-mail they can download the attachment and install the application directly onto their device.

Publishing on Android Market: It is always better to host your application on Android market (Google Playstore). Steps involved in doing so, are explained hereunder:

 Creating a Developer Profile:

o Create a developer profile at http://market.android.com/publish/Home using a Google account.
o Pay one-time registration fees.
o Agree Android Market Developer Distribution Agreement

 Submitting Your Apps: If you intend to charge for your application, click the Setup Merchant Account link located at the bottom of the screen. Here you enter additional information such as bank account and tax ID. You will be asked to supply some details for your application. Following are the compulsory details to be provided:

o The application in APK format
o At least two screenshots. You can use the DDMS perspective in Eclipse to capture screenshots of your application running on the Emulator or real device.A high-resolution application icon. This size of this image must be 512×512 pixels.
o Provide the title of your application, its description and recent update details.
o Indicate whether your application employs copy protection, and specify a content rating.

When all these setup is done, click Publish to publish your application on the Android Market.

3. What is the process for binding activities to services in android application?(10 Marks)

Often a service simply executes in its own thread, independently of the activity that calls it. This doesn’t pose any problem if you simply want the service to perform some tasks periodically and the activity does not need to be notified of the status of the service. For example, you may have a service that periodically logs the geographical location of the device to a database. In this case, there is no need for your service to interact with any activities, because its main purpose is to save the coordinates into a database. However, suppose you want to monitor for a particular location. When the service logs an address that is near the location you are monitoring, it might need to communicate that information to the activity. In this case, you would need to devise a way for the service to interact with the activity.

BINDING ACTIVITIES TO SERVICES

Real-world services are usually more sophisticated, requiring the passing of data so that they can do the job correctly for you. Using the service demonstrated earlier that downloads a set of files, suppose you now want to let the calling activity determine what files to download, instead of hardcoding them in the service.

First, in the calling activity, you create an Intent object, specifying the service name:

```
Button btnStart = (Button) findViewById(R.id.btnStartService);
```

```
btnStart.setOnClickListener(new View.OnClickListener()
{
public void onClick(View v)
{
Intent intent = new Intent(getBaseContext(), MyService.class);
}
});
```
You then create an array of URL objects and assign it to the Intent object through its putExtra() method. Finally, you start the service using the Intent object:
```
Button btnStart = (Button) findViewById(R.id.btnStartService);
btnStart.setOnClickListener(new View.OnClickListener()
{
public void onClick(View v)
{
Intent intent = new Intent(getBaseContext(), MyService.class);
try
{
URL[] urls = new URL[]
{
new URL("http://www.amazon.com/somefiles.pdf"),
new URL("http://www.wrox.com/somefiles.pdf"),
new URL("http://www.google.com/somefiles.pdf"),
new URL("http://www.learn2develop.net/somefiles.pdf")
};
intent.putExtra("URLs", urls);
} catch (MalformedURLException e)
{
e.printStackTrace();
}
startService(intent);
}
});
```
Note that the URL array is assigned to the Intent object as an Object array. On the service's end, you need to extract the data passed in through the Intent object in the onStartCommand() method:
```
@Override
public int onStartCommand(Intent intent, int flags, int startId)
{
// We want this service to continue running until it is explicitly
// stopped, so return sticky.
Toast.makeText(this, "Service Started", Toast.LENGTH_LONG).show();
Object[] objUrls = (Object[]) intent.getExtras().get("URLs");
URL[] urls = new URL[objUrls.length];
for (int i=0; i<objUrls.length-1; i++)
{
urls[i] = (URL) objUrls[i];
}
new DoBackgroundTask().execute(urls);
return START_STICKY;
}
```

The preceding first extracts the data using the getExtras() method to return a Bundle object. It then uses the get() method to extract out the URL array as an Object array. Because in Java you cannot directly cast an array from one type to another, you have to create a loop and cast each member of the array individually. Finally, you execute the background task by passing the URL array into the execute() method.This is one way in which your activity can pass values to the service. As you can see, if you have relatively complex data to pass to the service, you have to do some additional work to ensure that the data is passed correctly. A better way to pass data is to bind the activity directly to the service so that the activity can call any public members and methods on the service directly.

4.a. What are notifications? Explain various types of notifications.(6 Marks)

Notification is a user interface element that will display outside of any other app's normal UI to indicate that an event has occurred. Users can choose to view the notification while using other apps and respond to it when it's convenient for them.

Android notification will be displayed in the Notification area and to see the details regarding the notification, the user can expand it in by open the Notification drawer.

Following are the three types of android notifications,

1. Toast Notification – Shows message that fades away after a few seconds. (Background type also)
2. Status Notification – Shows notification message and displayed till user action. (Background type also)
3. Dialog Notification – Comes out of an active Activity.

(Background type) – is result of some background Service event that may not be related to current activity. That is, we can use this notification type in Service also, added to Activity.

1. Toast NotificationThis type of notification will be used when there is no need of user interaction on seeing this message. This message occupies a rectangular box which will fade in and fade out after some

time. 

The size of the box depends on the message content.

For example, when user creates an event using calendar application it will notify the user as “Event Created” after the create action is completed. Refer the image.

Toast notification is best suited for one way information to the use where we don’t expect any response. Toast message does not stop or disturb the current activity, just the message is shown in parallel.

Example for Android Toast Notification

Toast notification can be created from an Activity or Service. Toast is the class to be used as below,

```
Context appContext = getApplicationContext();
Toast mailMessage = Toast.makeText(appContext, “Email Received.”, Toast. LENGTH_LONG);
mailMessage.setGravity(Gravity.TOP, 0, 0); //optional
mailMessage.show();
```

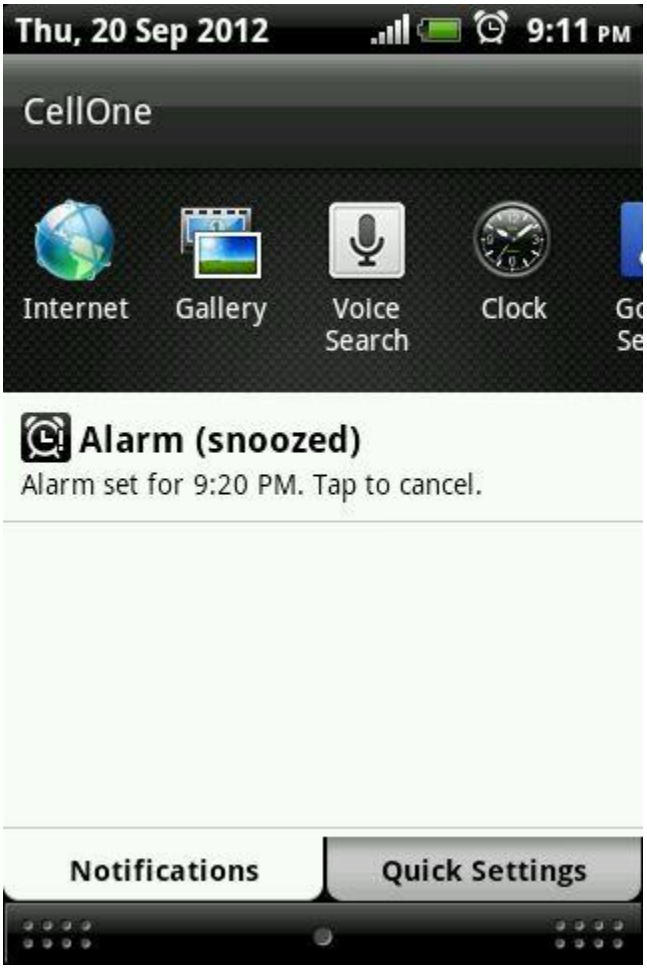
- duration – can be either LENGTH_SHORT or LENGTH_LONG
- setGravity – is used to position the message in screen. By default it shows at bottom centered. First parameter is Gravity a constant identifying location in container broadly like TOP | BOTTOM | LEFT … , second and third parameters are x, y-offset.

2. Status Notification

Status notification is used to display rich notification information especially from a (background) Service where user can interact. It will be shown as an icon with an alert in the status bar. When the user pulls down the status bar, the list of notification will be in the notification window.

For example when a SMS message is received a message icon is shown in the status bar. On pull down, the list of unread messages will be shown in the notification window.

Example shown in image: On snoozing the alarm, corresponding notification will be will be sent to the status bar with notification icon. A ticker message will be shown next to the icon for some time. In the image the clock icon represents the notification about the snooze event and the ticker message is shown next to the clock icon.



1. Create a simple notification with an icon alert. Alert can be a ticker text message or sound or vibration or flashlight.
2. Associate notification message with details shown on message expansion to activity/intent. Notification message can be a list and it is identified using a unique identifier. Existing messages can be updated too.
3. Register the notification message with notification manager. NotificationManager is a system service that manages all the notifications.

Example for Android Status Notification

```
//part 1 – notification icon alert

int icon = R.drawable.notification_icon;

// a ticker text message or sound or vibration or flashlight can be used for alert

CharSequence ticker = “Hi”;

long showAt = System.currenttimeMillis(); //immediately
```

```
Notification notification = new Notification(icon, ticker, showAt);

//part 2 – associate notification message with details shown on message expansion to activity/intent

CharSequence notificationTitle = "Notification:";

CharSequence notificationMessage = "SMS Received.";

Intent intent = new Intent(this, Activity.class);

PendingIntent objPendingIntent = PendingIntent.getActivity(this, 0, intent, 0);

Context ctx = getApplicationContext();

notification.setLatestEventInfo(ctx, notificationTitle, notificationMessage, objPendingIntent);

//part 3 – register the notification message with notification manager

private static final int notificationIdentifier = 101; //an unique number set by developer to identify a
notification, using this notification can be updated/replaced

NotificationManager notificationManager = (NotificationManager)

getSystemService(Context.NOTIFICATION_SERVICE);

notificationManager.notify(notificationIdentifier, objNotification);
```

Notification Alerts

Sound:

```
notification.defaults |= Notification.DEFAULT_SOUND;

//use the above default or set custom valuse as below

notification.sound = Uri.parse("file:///sdcard/notification/robo_da.mp3");
```

Vibration:

```
notification.defaults |= Notification.DEFAULT_VIBRATE;

//use the above default or set custom valuse as below
```

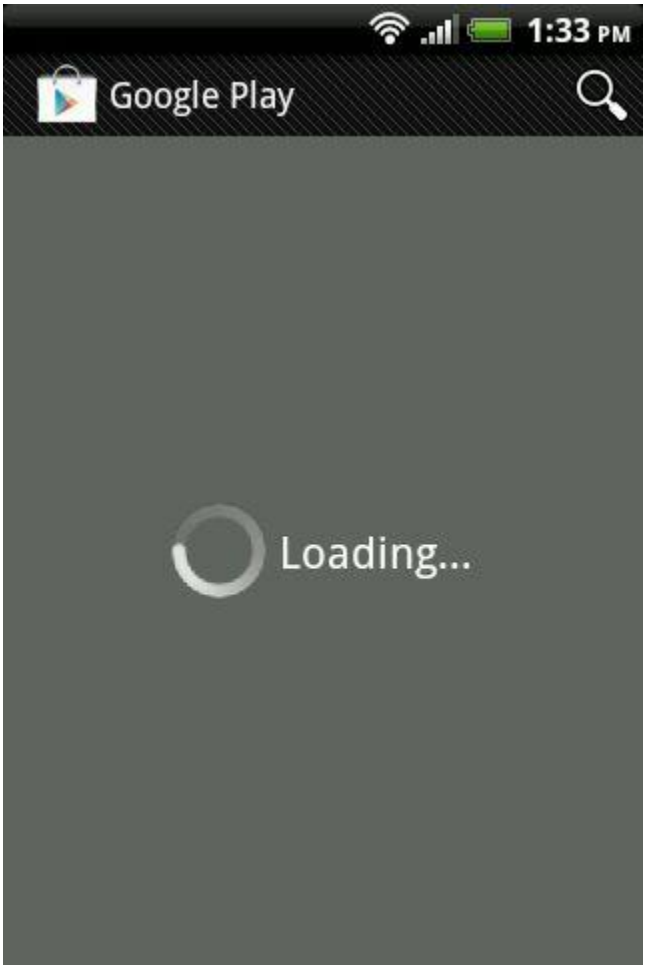
```
long[] vibrate = {0,200,100,200};

notification.vibrate = vibrate;
```

Flash Light:

```


notification.defaults |= Notification.DEFAULT_LIGHTS;

//use the above default or set custom valuse as below

notification.ledARGB = 0xffff0000;//red color

notification.ledOnMS = 400;

notification.ledOffMS = 500;

notification.flags |= Notification.FLAG_SHOW_LIGHTS;
```

3. Dialog Notification

Dialog notification is not an exact type of notification. Dialog is common in window based UIs. A small panel that appears on top of an active window and user will not be able to do any other activity other than acting on the dialog. This is same here too. From an android Activity a dialog will be launched and the Activity loses focus. User should give input and work on the dialog. Once the user action is completed the dialog is closed. Dialog has many uses and one among them is notification to user.



For example we can show a progress bar which is a notification to user. We can ask for confirmation ‘yes’ or ‘no’ from user and this is a type of notification. For all these purposes dialog notification is used. There are many types of dialogs available such as,

- AlertDialog
- ProgressDialog
- DatePickerDialog

TimePickerDialog

4.b. Write a note on Date Picker rand Time Picker Views(4 Marks)

Selecting the date and time is one of the common tasks you need to perform in a mobile application. Android supports this functionality through the TimePicker and DatePicker views. The TimePicker view enables users to select a time of the day, in either 24-hour mode or AM/PM mode. Using the DatePicker, you can enable users to select a particular date on the activity.

To add TimePicker in the android application, use the following code:

```
<TimePicker android:id=”@+id/timePicker”
android:layout_width=”wrap_content”
android:layout_height=”wrap_content” />
```

To add DatePicker in the android application, use the following code:

<DatePicker android:id=”@+id/datePicker”
android:layout_width=”wrap_content”
android:layout_height=”wrap_content” />

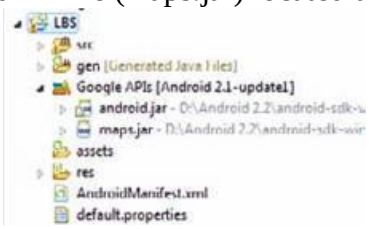
5. Write in detail how to display Google maps in your own android applications.(10 Marks)

Google Maps is one of the many applications bundled with the Android platform. In addition to simply using the Maps application, you can also embed it into your own applications and make it do some very cool things. This section describes how to use Google Maps in your Android applications and programmatically perform the following:

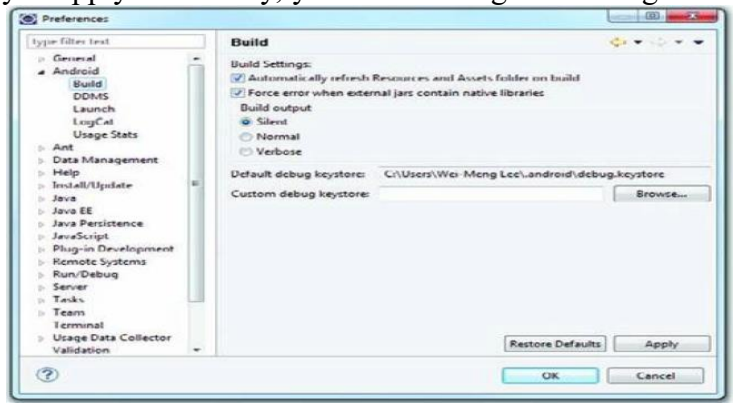
- Change the views of Google Maps.
- Obtain the latitude and longitude of locations in Google Maps.
- Perform geocoding and reverse geocoding (translating an address to latitude and longitude and vice versa).
- Add markers to Google Maps.

We will discuss how to build a project using maps.

Creating the Project: Create a new android project. In order to use Google Maps in your Android application, you need to ensure that you check the Google APIs as your build target. Google Maps is not part of the standard Android SDK, so you need to find it in the Google APIs add-on. If LBS is the name of your project, then you can see the additional
JAR file (maps.jar) located under the Google APIs folder as below–



Obtaining the Maps API Key: Beginning with the Android SDK release v1.0, you need to apply for a free Google Maps API key before you can integrate Google Maps into your Android application. When you apply for the key, you must also agree to Google’s terms of use, so be sure to read them carefully.



First, if you are testing the application on the Android Emulator or an Android device directly connected to your development machine, locate the SDK debug certificate located in the default folder

(C:\Users\<username>\.android for Windows 7 users). You can verify the existence of the debug certificate by going to Eclipse and selecting Window ➪Preferences. Expand the Android item and select Build (as shown in figure above). On the right side of the window, you will be able to see the debug certificate's location.The filename of the debug keystore is debug.keystore. This is the certificate that Eclipse uses to sign your application so that it may be run on the Android Emulator or devices.

Modify your AndroidManifest.xml fi le by adding both the <uses-library> element and the INTERNET permission.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="net.learn2develop.LBS"
      android:versionCode="1"
      android:versionName="1.0">
    <application android:icon="@drawable/icon" android:label="@string/app_name">

    <uses-library android:name="com.google.android.maps" />

        <activity android:name=".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>
    <uses-sdk android:minSdkVersion="8" />

    <uses-permission android:name="android.permission.INTERNET"></uses-permission>
</manifest>
```

Add the MapView element to your UI.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<com.google.android.maps.MapView
    android:id="@+id/mapView"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:enabled="true"
    android:clickable="true"
    android:apiKey="<YOUR KEY>" />
</LinearLayout>
```

```
import android.os.Bundle;

import com.google.android.maps.MapActivity;

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

    @Override
    protected boolean isRouteDisplayed() {
        // TODO Auto-generated method stub
        return false;
    }
}
```

6. Define service. How do you create your own service in Android? Explain with a snippet code(10 Marks)

A service is an application in Android that runs in the background without needing to interact with the user. For example, while using an application, you may want to play some background music at the same time. In this case, the code that is playing the background music has no need to interact with the user, and hence it can be run as a service. Services are also ideal for situations in which there is no need to present a UI to the user.

Following are the steps involved in creating own service.

- Create a new android application.
- Add a new class file to the project and name it MyService.java. Write the following code in it:

```
package net.learn2develop.Services;
import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.widget.Toast;
public class MyService extends Service
{
@Override
public IBinder onBind(Intent arg0)
{
return null;
}
public int onStartCommand(Intent intent, int flags, int startId)
{
Toast.makeText(this,“ServiceStarted”,Toast.LENGTH_LONG).show();
return START_STICKY;
}
public void onDestroy()
{
```

```
super.onDestroy();
Toast.makeText(this,“ServiceDestroyed”,Toast.LENGTH_LONG).show();
}
}
```

The onBind() method enables you to bind an activity to a service. This in turn enables an activity to directly access members and methods inside a service. The onStartCommand() method is called when you start the service explicitly using the startService() method. This method signifies the start of the service, and you code it to do the things you need to do for your service. In this method, you returned the constant START_STICKY so that the service will continue to run until it is explicitly stopped. The onDestroy() method is called when the service is stopped using the stopService() method. This is where you clean up the resources used by your service.

 In the AndroidManifest.xml file, add the following statement :

```
<service android:name=”.MyService” />
```

 In the activity_main.xml file, add the following statements in bold:

```
<Button android:id=”@+id/btnStartService”
android:layout_width=”fill_parent”
android:layout_height=”wrap_content”
android:text=”Start Service” />
<Button android:id=”@+id/btnStopService”
android:layout_width=”fill_parent”
android:layout_height=”wrap_content”
android:text=”Stop Service” />
```

 Add the following statements in bold to the MainActivity.java file:

```
import android.content.Intent;
import android.view.View;
import android.widget.Button;
public class MainActivity extends Activity
{
public void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
Button btnStart = (Button) findViewById(R.id.btnStartService);
btnStart.setOnClickListener(new View.OnClickListener()
{
public void onClick(View v)
{
startService(new Intent(getBaseContext(), MyService.class));
}
});
Button btnStop = (Button) findViewById(R.id.btnStopService);
btnStop.setOnClickListener(new View.OnClickListener()
{
public void onClick(View v)
{
stopService(new Intent(getBaseContext(), MyService.class));
}
});
}
```

}

7. Develop a mobile application to list the tourist places of Karnataka using List View.(10 Marks)

Activity_mail.xml

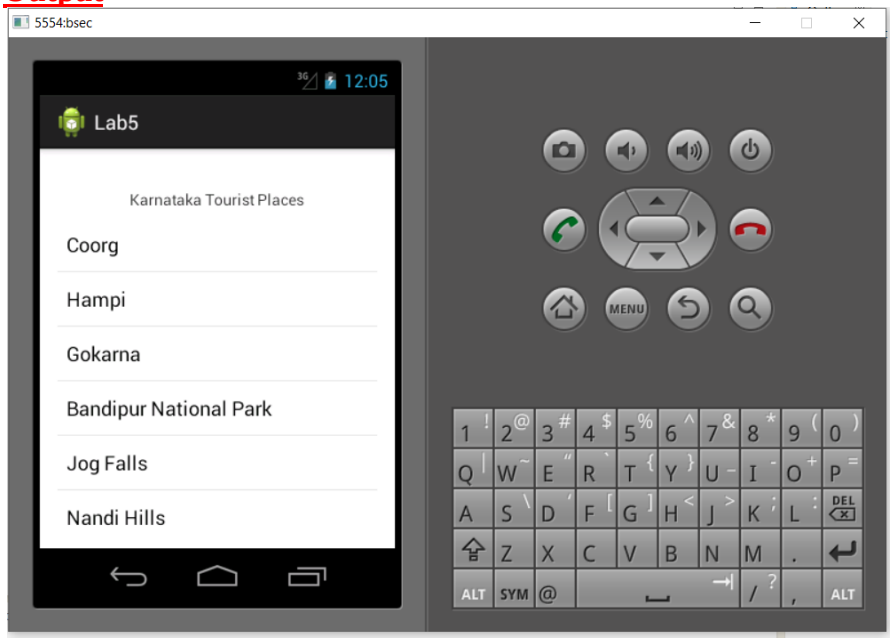
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.lab5.MainActivity" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp"
        android:text="Karnataka Tourist Places" />

    <ListView
        android:id="@+id/listView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textView1"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="14dp" >
    </ListView>

</RelativeLayout>
```

MainActivity.java

```
package com.example.lab5;


import android.app.Activity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;


public class MainActivity extends Activity {

      String[] places = {
                  "Coorg",
                  "Hampi",
                  "Gokarna",
                  "Bandipur National Park",
                  "Jog Falls",
                  "Nandi Hills",
                  "Dharmashala",
                  "Mysore",
                  "Chikmagalur",
                  "Bengaluru",
                  "Badami",
                  "Nagarhole National Park"
                  };
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ArrayAdapter adapter= (new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_1, places));
       ListView listview=(ListView) findViewById(R.id.listView1);
       listview.setAdapter(adapter);
    }


}
```

Output



8. Devise an application that draws basic graphical primitives (rectangle, circle) on the screen. (10 Marks)

Activity_main.xml

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

    <ImageView
        android:id="@+id/image"
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       />
</RelativeLayout>
```

MainActivity.java

```
package com.example.lab4;


import android.app.Activity;

import android.graphics.Bitmap;

import android.graphics.Canvas;

import android.graphics.Color;

import android.graphics.Paint;

import android.graphics.drawable.BitmapDrawable;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.ImageView;



public class MainActivity extends Activity {
```

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

 //Creating a Bitmap
   Bitmap bg = Bitmap.createBitmap(720, 1280, Bitmap.Config.ARGB_8888);

   //Setting the Bitmap as background for the ImageView
   ImageView i = (ImageView) findViewById(R.id.image);
   i.setBackgroundDrawable(new BitmapDrawable(bg));

   //Creating the Canvas Object
   Canvas canvas = new Canvas(bg);

   //Creating the Paint Object and set its color & TextSize
   Paint paint = new Paint();
   paint.setColor(Color.BLUE);
   paint.setTextSize(50);

   //To draw a Rectangle
   canvas.drawText("Rectangle", 420, 150, paint);
   canvas.drawRect(400, 200, 650, 700, paint);

   //To draw a Circle
   canvas.drawText("Circle", 120, 150, paint);
```

```
        canvas.drawCircle(200, 350, 150, paint);


        //To draw a Square

        canvas.drawText("Square", 120, 800, paint);

        canvas.drawRect(50, 850, 350, 1150, paint);


        //To draw a Line

        canvas.drawText("Line", 480, 800, paint);

        canvas.drawLine(520, 850, 520, 1150, paint);

    }

}
```

Manifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.lab4"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="16"
        android:targetSdkVersion="21" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".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>
```

OUTPUT