Q.1) Write a Program to draw a route between two locations.

Code:

MapActivity.java:

```
package com.example.myapplication;

import android.Manifest;
import android.content.pm.PackageManager;
import android.graphics.Color;
import android.location.Location;
import android.os.AsyncTask;
import android.os.Build;
import android.os.Bundle;
import android.util.Log;
import android.widget.Toast;

import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import androidx.fragment.app.FragmentActivity;

import
com.google.android.gms.common.ConnectionResu
lt;
import
com.google.android.gms.common.api.GoogleApiCl
ient;
import
com.google.android.gms.location.LocationListener
;
import
com.google.android.gms.location.LocationRequest
;
import
com.google.android.gms.location.LocationServices
;
import
com.google.android.gms.maps.CameraUpdateFact
ory;
import com.google.android.gms.maps.GoogleMap;
import
com.google.android.gms.maps.OnMapReadyCallba
ck;
import
com.google.android.gms.maps.SupportMapFragm
ent;
import
com.google.android.gms.maps.model.BitmapDescr
iptorFactory;
import
com.google.android.gms.maps.model.LatLng;
import
com.google.android.gms.maps.model.Marker;
import
com.google.android.gms.maps.model.MarkerOpti
ons;
import
com.google.android.gms.maps.model.PolylineOpti
ons;

import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

public class MapsActivity extends FragmentActivity
implements OnMapReadyCallback,
GoogleApiClient.ConnectionCallbacks,
GoogleApiClient.OnConnectionFailedListener,
LocationListener {

private GoogleMap mMap;
ArrayList<LatLng> MarkerPoints;
GoogleApiClient mGoogleApiClient;
Location mLastLocation;
Marker mCurrLocationMarker;
LocationRequest mLocationRequest;

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

if (android.os.Build.VERSION.SDK_INT >=
Build.VERSION_CODES.M) {
checkLocationPermission();
}
// Initializing
MarkerPoints = new ArrayList<>();

// Obtain the SupportMapFragment and get
notified when the map is ready to be used.
SupportMapFragment mapFragment =
(SupportMapFragment)
getSupportFragmentManager()
.findFragmentById(R.id.map);
mapFragment.getMapAsync(this);
}

/**
* Manipulates the map once available.
* This callback is triggered when the map is ready to be used.
* This is where we can add markers or lines, add listeners or move the camera. In this case,
* we just add a marker near Sydney, Australia.
* If Google Play services is not installed on the device, the user will be prompted to install
* it inside the SupportMapFragment. This method will only be triggered once the user has
* installed Google Play services and returned to the app.
*/
@Override
public void onMapReady(GoogleMap googleMap) {
mMap = googleMap;

//Initialize Google Play Services
if (android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)
== PackageManager.PERMISSION_GRANTED) {
buildGoogleApiClient();
mMap.setMyLocationEnabled(true);
}
}
else {
buildGoogleApiClient();
mMap.setMyLocationEnabled(true);
}

// Setting onclick event listener for the map
mMap.setOnMapClickListener(new GoogleMap.OnMapClickListener() {

@Override
public void onMapClick(LatLng point) {

// Already two locations
if (MarkerPoints.size() > 1) {
MarkerPoints.clear();
mMap.clear();
}

// Adding new item to the ArrayList
MarkerPoints.add(point);

// Creating MarkerOptions
MarkerOptions options = new MarkerOptions();

// Setting the position of the marker
options.position(point);

/**
* For the start location, the color of marker is GREEN and
* for the end location, the color of marker is RED.
*/
if (MarkerPoints.size() == 1) {
options.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_GREEN));
} else if (MarkerPoints.size() == 2) {
options.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_RED));
}


// Add new marker to the Google Map Android API V2
mMap.addMarker(options);

// Checks, whether start and end locations are captured
if (MarkerPoints.size() >= 2) {
LatLng origin = MarkerPoints.get(0);
LatLng dest = MarkerPoints.get(1);

// Getting URL to the Google Directions API
String url = getUrl(origin, dest);
Log.d("onMapClick", url.toString());
FetchUrl FetchUrl = new FetchUrl();

// Start downloading json data from Google Directions API
FetchUrl.execute(url);
//move map camera
mMap.moveCamera(CameraUpdateFactory.newLatLng(origin));
mMap.animateCamera(CameraUpdateFactory.zoomTo(11));
}

}
});

}

private String getUrl(LatLng origin, LatLng dest) {

// Origin of route
String str_origin = "origin=" + origin.latitude + "," + origin.longitude;

// Destination of route
String str_dest = "destination=" + dest.latitude + "," + dest.longitude;


// Sensor enabled
String sensor = "sensor=false";

// Building the parameters to the web service
String parameters = str_origin + "&" + str_dest + "&" + sensor;
```

```
// Output format
String output = "json";

// Building the url to the web service
String url =
"https://maps.googleapis.com/maps/api/direction
s/" + output + "?" + parameters;


return url;
}

/**
* A method to download json data from url
*/
private String downloadUrl(String strUrl) throws
IOException {
String data = "";
InputStream iStream = null;
HttpURLConnection urlConnection = null;
try {
URL url = new URL(strUrl);

// Creating an http connection to communicate
with url
urlConnection = (HttpURLConnection)
url.openConnection();

// Connecting to url
urlConnection.connect();

// Reading data from url
iStream = urlConnection.getInputStream();

BufferedReader br = new BufferedReader(new
InputStreamReader(iStream));

StringBuffer sb = new StringBuffer();

String line = "";
while ((line = br.readLine()) != null) {
sb.append(line);
}

data = sb.toString();
Log.d("downloadUrl", data.toString());
br.close();

} catch (Exception e) {
Log.d("Exception", e.toString());
} finally {
iStream.close();
urlConnection.disconnect();
}
return data;
}

// Fetches data from url passed
private class FetchUrl extends AsyncTask<String,
Void, String> {

@Override
protected String doInBackground(String... url) {

// For storing data from web service
String data = "";

try {
// Fetching the data from web service
data = downloadUrl(url[0]);
Log.d("Background Task data", data.toString());
} catch (Exception e) {
Log.d("Background Task", e.toString());
}
return data;
}

@Override
protected void onPostExecute(String result) {
super.onPostExecute(result);

ParserTask parserTask = new ParserTask();

// Invokes the thread for parsing the JSON data
parserTask.execute(result);

}
}

/**
* A class to parse the Google Places in JSON format
*/
private class ParserTask extends AsyncTask<String,
Integer, List<List<HashMap<String, String>>>> {

// Parsing the data in non-ui thread
@Override
protected List<List<HashMap<String, String>>>
doInBackground(String... jsonData) {

JSONObject jObject;
List<List<HashMap<String, String>>> routes = null;

try {
jObject = new JSONObject(jsonData[0]);
Log.d("ParserTask",jsonData[0].toString());
DataParser parser = new DataParser();
Log.d("ParserTask", parser.toString());

// Starts parsing data
routes = parser.parse(jObject);
Log.d("ParserTask","Executing routes");
Log.d("ParserTask",routes.toString());
} catch (Exception e) {
Log.d("ParserTask",e.toString());
e.printStackTrace();
}
return routes;
}

// Executes in UI thread, after the parsing process
@Override
protected void
onPostExecute(List<List<HashMap<String,
String>>> result) {
ArrayList<LatLng> points;
PolylineOptions lineOptions = null;

// Traversing through all the routes
for (int i = 0; i < result.size(); i++) {
points = new ArrayList<>();
lineOptions = new PolylineOptions();

// Fetching i-th route
List<HashMap<String, String>> path = result.get(i);

// Fetching all the points in i-th route
for (int j = 0; j < path.size(); j++) {
HashMap<String, String> point = path.get(j);

double lat = Double.parseDouble(point.get("lat"));
double lng = Double.parseDouble(point.get("lng"));
LatLng position = new LatLng(lat, lng);

points.add(position);
}

// Adding all the points in the route to LineOptions
lineOptions.addAll(points);
lineOptions.width(10);
lineOptions.color(Color.RED);

Log.d("onPostExecute","onPostExecute lineoptions
decoded");

}

// Drawing polyline in the Google Map for the i-th
route
if(lineOptions != null) {
mMap.addPolyline(lineOptions);
}
else {
Log.d("onPostExecute","without Polylines drawn");
}
}
}
```

```
protected synchronized void
buildGoogleApiClient() {
mGoogleApiClient = new
GoogleApiClient.Builder(this)
.addConnectionCallbacks(this)
.addOnConnectionFailedListener(this)
.addApi(LocationServices.API)
.build();
mGoogleApiClient.connect();
}

@Override
public void onConnected(Bundle bundle) {

mLocationRequest = new LocationRequest();
mLocationRequest.setInterval(1000);
mLocationRequest.setFastestInterval(1000);
mLocationRequest.setPriority(LocationRequest.PRI
ORITY_BALANCED_POWER_ACCURACY);
if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)
== PackageManager.PERMISSION_GRANTED) {
LocationServices.FusedLocationApi.requestLocatio
nUpdates(mGoogleApiClient, mLocationRequest,
this);
}

}

@Override
public void onConnectionSuspended(int i) {

}

@Override
public void onLocationChanged(Location location)
{

mLastLocation = location;
if (mCurrLocationMarker != null) {
mCurrLocationMarker.remove();
}

//Place current location marker
LatLng latLng = new LatLng(location.getLatitude(),
location.getLongitude());
MarkerOptions markerOptions = new
MarkerOptions();
markerOptions.position(latLng);
markerOptions.title("Current Position");
markerOptions.icon(BitmapDescriptorFactory.defa
ultMarker(BitmapDescriptorFactory.HUE_MAGENT
A));
mCurrLocationMarker =
mMap.addMarker(markerOptions);

//move map camera
mMap.moveCamera(CameraUpdateFactory.newLa
tLng(latLng));
mMap.animateCamera(CameraUpdateFactory.zoo
mTo(11));

//stop location updates
if (mGoogleApiClient != null) {
LocationServices.FusedLocationApi.removeLocatio
nUpdates(mGoogleApiClient, this);
}

}

@Override
public void onConnectionFailed(ConnectionResult
connectionResult) {

}

public static final int
MY_PERMISSIONS_REQUEST_LOCATION = 99;
public boolean checkLocationPermission(){
if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)
!= PackageManager.PERMISSION_GRANTED) {

// Asking user if explanation is needed
if
(ActivityCompat.shouldShowRequestPermissionRa
tionale(this,
Manifest.permission.ACCESS_FINE_LOCATION)) {

// Show an explanation to the user
*asynchronously* -- don't block
// this thread waiting for the user's response! After
the user
// sees the explanation, try again to request the
permission.

//Prompt the user once explanation has been
shown
ActivityCompat.requestPermissions(this,
new
String[]{Manifest.permission.ACCESS_FINE_LOCATI
ON},
MY_PERMISSIONS_REQUEST_LOCATION);


} else {
// No explanation needed, we can request the
permission.
ActivityCompat.requestPermissions(this,
new
String[]{Manifest.permission.ACCESS_FINE_LOCATI
ON},
MY_PERMISSIONS_REQUEST_LOCATION);
}
return false;
} else {
return true;
}
}

@Override
public void onRequestPermissionsResult(int
requestCode,
String permissions[], int[] grantResults) {
switch (requestCode) {
case MY_PERMISSIONS_REQUEST_LOCATION: {
// If request is cancelled, the result arrays are
empty.
if (grantResults.length > 0
&& grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {

// permission was granted. Do the
// contacts-related task you need to do.
if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)
== PackageManager.PERMISSION_GRANTED) {

if (mGoogleApiClient == null) {
buildGoogleApiClient();
}
mMap.setMyLocationEnabled(true);
}

} else {

// Permission denied, Disable the functionality that
depends on this permission.
Toast.makeText(this, "permission denied",
Toast.LENGTH_LONG).show();
}
return;
}

// other 'case' lines to check for other permissions
this app might request.
// You can add here other case statements
according to your requirement.
}
}
}
```

DataParser.java:

```
package com.example.myapplication;
import
com.google.android.gms.maps.model.LatLng;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

public class DataParser {

/** Receives a JSONObject and returns a list of lists
containing latitude and longitude */
public List<List<HashMap<String,String>>>
parse(JSONObject jObject){
List<List<HashMap<String, String>>> routes = new
ArrayList<>() ;
JSONArray jRoutes;
JSONArray jLegs;
JSONArray jSteps;
try {
jRoutes = jObject.getJSONArray("routes");
/** Traversing all routes */
for(int i=0;i<jRoutes.length();i++){
jLegs = (
(JSONObject)jRoutes.get(i)).getJSONArray("legs");
List path = new ArrayList<>();
/** Traversing all legs */
for(int j=0;j<jLegs.length();j++){
jSteps = (
(JSONObject)jLegs.get(j)).getJSONArray("steps");
/** Traversing all steps */
for(int k=0;k<jSteps.length();k++){
String polyline = "";
polyline =
(String)((JSONObject)((JSONObject)jSteps.get(k)).g
et("polyline")).get("points");
List<LatLng> list = decodePoly(polyline);
/** Traversing all points */
for(int l=0;l<list.size();l++){
HashMap<String, String> hm = new HashMap<>();
hm.put("lat", Double.toString((list.get(l)).latitude)
);
hm.put("lng",
Double.toString((list.get(l)).longitude) );
path.add(hm);
}
}
routes.add(path);
}
}

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


return routes;
}
/**
* Method to decode polyline points
* Courtesy :
http://jeffreysambells.com/2010/05/27/decoding-
polylines-from-google-maps-direction-api-with-
java
* */
private List<LatLng> decodePoly(String encoded) {
List<LatLng> poly = new ArrayList<>();
int index = 0, len = encoded.length();
int lat = 0, lng = 0;
while (index < len) {
int b, shift = 0, result = 0;
do {
b = encoded.charAt(index++) - 63;
result |= (b & 0x1f) << shift;
shift += 5;
} while (b >= 0x20);
int dlat = ((result & 1) != 0 ? ~(result >> 1) : (result
>> 1));
lat += dlat;
shift = 0;
result = 0;
do {
b = encoded.charAt(index++) - 63;
result |= (b & 0x1f) << shift;
shift += 5;
} while (b >= 0x20);
int dlng = ((result & 1) != 0 ? ~(result >> 1) : (result
>> 1));
lng += dlng;
LatLng p = new LatLng((((double) lat / 1E5)),
(((double) lng / 1E5)));
poly.add(p);}
return poly;
}}
```

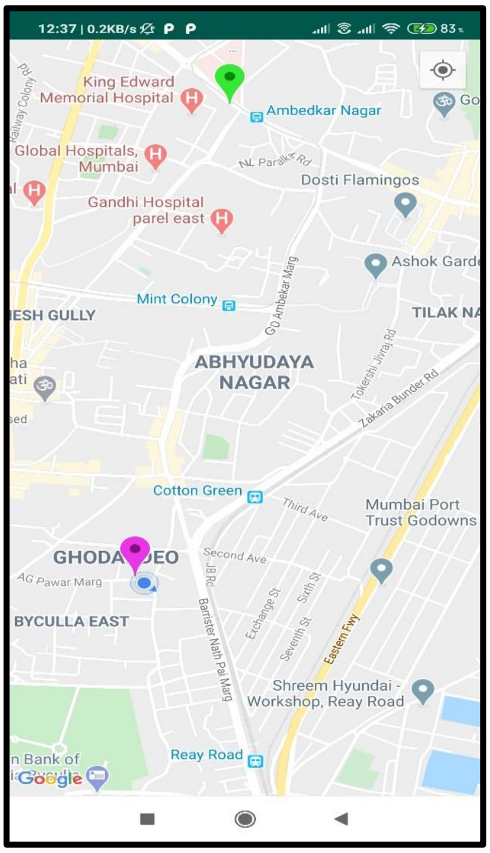
XML:

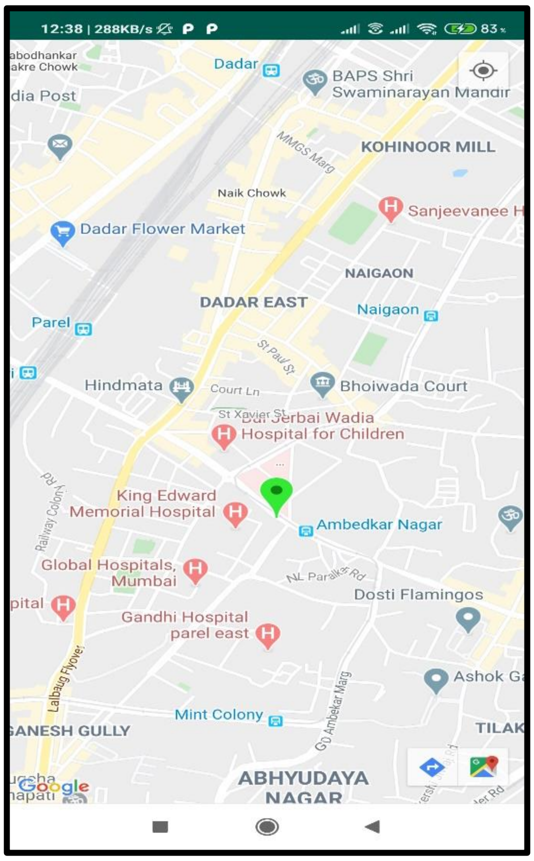
```
<?xml version="1.0" encoding="utf-8"?>
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:map="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/map"
   android:name="com.google.android.gms.maps.SupportMapFragment"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MapsActivity" />
```

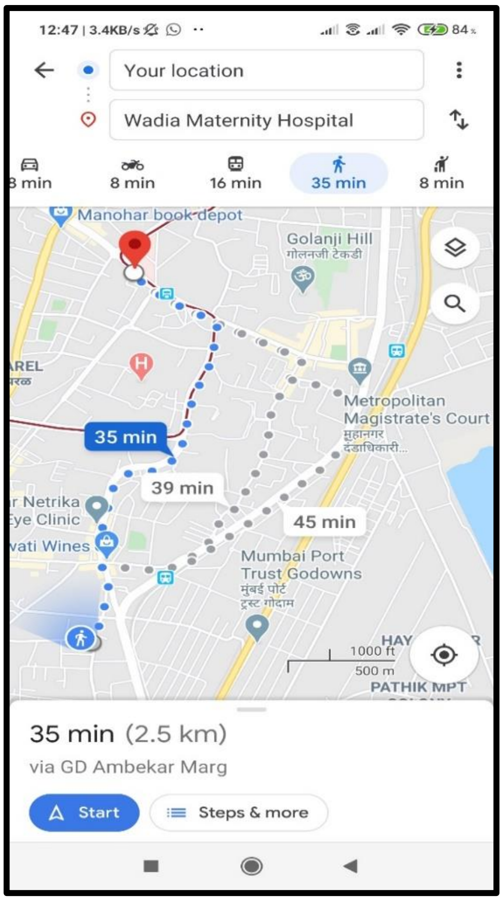
AndroidManifest:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.example.myapplication">
   <uses-permission android:name="com.javapapers.currentlocationinmap.permission.MAPS_RECEIVE" />
   <uses-permission android:name="android.permission.INTERNET" />
   <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
   <uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
   <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
   <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
   <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
   <application
      android:allowBackup="true"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:roundIcon="@mipmap/ic_launcher_round"
      android:supportsRtl="true"
      android:theme="@style/AppTheme">
      <!--
          The API key for Google Maps-based APIs is defined as a string resource.
          (See the file "res/values/google_maps_api.xml").
          Note that the API key is linked to the encryption key used to sign the APK.
          You need a different API key for each encryption key, including the release key that is used to
          sign the APK for publishing.
          You can define the keys for the debug and release targets in src/debug/ and src/release/.
      -->
      <meta-data
         android:name="com.google.android.geo.API_KEY"
         android:value="@string/google_maps_key" />
      <activity
         android:name=".MapsActivity"
         android:label="@string/title_activity_maps">
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
   </application>
</manifest>
```

Output:


12:37 | 0.2KB/s
83
King Edward Memorial Hospital
Ambedkar Nagar
Global Hospitals, Mumbai
NL Paralkar Rd
Dosti Flamingos
Gandhi Hospital parel east
Ashok Gard
Mint Colony
GD Ambekar Marg
TILAK NA
ESH GULLY
ABHYUDAYA NAGAR
Tokershi Jivraj Rd
Zakaria Bunder Rd
Cotton Green
Third Ave
Mumbai Port Trust Godowns
GHODAPDEO
Second Ave
AG Pawar Marg
BYCULLA EAST
Barrister Nath Pai Marg
Exchange St
Sixth St
Seventh St
Eastern Fwy
Shreem Hyundai - Workshop, Reay Road
Reay Road
Google


12:38 | 288KB/s
83
Dadar
BAPS Shri Swaminarayan Mandir
dia Post
MMGS Marg
KOHINOOR MILL
Naik Chowk
Sanjeevanee H
Dadar Flower Market
NAIGAON
DADAR EAST
Naigaon
Parel
St Paul St
Hindmata
Court Ln
Bhoiwada Court
St Xavier St
Bai Jerbai Wadia Hospital for Children
King Edward Memorial Hospital
Ambedkar Nagar
Railway Colony Rd
Global Hospitals, Mumbai
NL Paralkar Rd
Dosti Flamingos
Gandhi Hospital parel east
Lalbaug Flyover
Ashok Ga
Mint Colony
GD Ambekar Marg
ANESH GULLY
TILAK
ABHYUDAYA NAGAR
Google


12:47 | 3.4KB/s
84
Your location
Wadia Maternity Hospital
8 min
8 min
16 min
35 min
8 min
Manohar book depot
Golanji Hill
REL
Metropolitan Magistrate's Court
35 min
39 min
45 min
Netrika Eye Clinic
vati Wines
Mumbai Port Trust Godowns
1000 ft
500 m
PATHIK MPT
35 min (2.5 km)
via GD Ambekar Marg
Start
Steps & more