DATE
PAGE

Q.3 (A)

1) ____ converts Java byte code into Dalvik byte code?

→ Dalvik Executable (DEX) ~~compiler~~ compiler

2) In memory, Activity doesn't exists that is ____ state.

→ Destroyed

3) Give the name of different types sensors.

→ accelerometer, gyroscope, proximity sensor, ambient light sensor, magnetometer barometer and GPS sensor.

4) In Status data a ____ use in the android system.

→ Bundle

Q.3 (B)

1) Define content provider.

→ A content provider in Android is a component that manages access to a structured set of data. It acts as an intermediary between the data source and different applications, allowing them to securely share data. Content providers offers a standardized interface for querying, inserting, updating and deleting data, enabling applications to access and manipulate data from other apps or system components.

2) Define Live wallpaper in Android.

→ A Live wallpaper in Android is an interactive and dynamic background that can respond to user interactions or interactive elements, providing a more engaging user experience. Live wallpapers are implemented as specialized services that run continuously in the background and render the wallpaper surface.

DATE:
PAGE:

Q.3 (C)

1) Explain Alarm manager.

→ The AlarmManager class in Android is a system service that allows you to schedule tasks or operations to be executed at a specified time, even if your application is not currently running. It provides a way to perform time-based operations such as scheduling background tasks, sending notifications, updating data, syncing data with servers, and more.

* Key points:

(i) Functionality: The Alarmmanager allows you to schedule your applicatim to run at a specific time in the future.

(ii) Permissions: The Act AlarmManager requires specific permissions such as "android.permiss VIBRATE" and "android.permission.WAKE_LOCK" to function properly.

DATE:
PAGE:

2) Explain Camera API Intent.

→ The camera API intent is a mechanism in Android that allows developers to capture photos or videos using the device's camera hardware. It provides a simple and standardized way to launch the camera application, capture media content and receive the captured data back in the calling application for further processing or display.

* How the camera API intent works:

(i) Creating intent: To launch the camera application, you create an Intent object with the action "MediaStore.ACTION_IMAGE_CAPTURE" for photos or "MediaStore.ACTION_VIDEO_CAPTURE" for videos.

(ii) Launching Camera App: You start the application by calling "startActivityForResult()" method, passing the Intent as an argument.

(iii) Capturing Media :

(iv) Receiving Media : The camera application returns the captured data to the calling application through the "onActivityResult()" method.

(v) permission : Don't forget to request the necessary permissions in your AndroidManifest.xml file.

Q.3 (D)

1) write a note on Light sensor with example.

→ The light sensor, also known as an ambient light sensor or illuminance sensor, is a hardware component ambient light in the ~~door~~ device's environment. This sensor provides valuable data to the device, allowing it to adjust display brightness, optimize power consumption and enhance user experience.

* Here's a simple example demonstrating how access light sensor data in an Android application:-

=> LightSensorActivity.kt

```
class LightSensorActivity : AppCompatActivity(),
        SensorEventListener
{
    private lateinit var sensorManager: SensorManager
    private var lightSensor: Sensor? = null
    private lateinit var lightValueTextView: TextView

    override fun onCreate(savedInstanceState: Bundle?)
    {
        super.onCreate(saveInstanceState)
        setContentView(R.layout.activity_light_sensor)

        lightValueTextView = findViewById(R.id.
                                lightValueTextView)
        sensorManager = getSystemService(SENSOR_
                SERVICE) as SensorManager
        lightSensor = sensorManager.getDefaultSensor
                (Sensor.TYPE_LIGHT)
    }
```

```
override fun onResume() {
    super.onResume()
    sensorManager.registerListener(this,
        lightSensor, sensorManager.
        SENSOR_DELAY_NORMAL)
}

override fun onPause() { super.onPause()
    sensorManager.unregisterListener(this)
}

override fun onSensorChanged(event:
    SensorEvent?) {
    if(event.sensor.type == Sensor.TYPE_LIGHT)
    {
        lightValueTextView.text = "Light Intensity:
            ${lightValue} lux"
    }
}

override fun onAccuracyChanged(sensor:
    Sensor?, accuracy: Int) {
    }
}
```

2) Write a note on position sensor with example

→ The position sensor in Android devices, such as the accelerometer and gyroscope, play a crucial role in detecting and measuring the device's orientation and movement in three-dimensional space.

→ These sensor provide valuable data to applications, enabling various functionalities such as screen orientation changes, gaming controls, augmented reality experiences and motion-based gestures.

⇒ <u>Code</u> MainActivity.kt

```
kottin
class MainActivity : AppCompatActivity(),
        SensorEventListener {
    private lateinit var sensorManager:
                                SensorManager
    private var accelerometer : Sensor? = null
```

DATE:
PAGE:

```
override fun onCreate(savedInstanceState:
                      Bundle?)
{
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_position
                   Sensor)

    SensorManager = getSystemService
        (SENSOR_SERVICE) as SensorManager

    accelerometer = sensorManager.getDefaultSens
        (Sensor.TYPE_ACCELEROMETER)

}

override. fun onResume() {
{
    super.onResume():
    SensorManager.registerListener(this,
        accelerometer, SensorManager.SENSOR_
                        DELAY_NORMAL)
}
```

```
override fun onPause()
{
    super.onPause()
    sensorManager.unregisterListener(this)
}

override fun onSensorChanged(event: SensorEvent)
{
  if (event.sensor.type == Sensor.TYPE_ACCELEROMETER)
  {
    val x = event.values[0]
    val y = event.values[1]
    val z = event.values[2]

    if (oc > 5) {
    {
      requestedOrientation = ActivityInfo.SCREEN_
          ORIENTATION_LANDSCAPE
    }
    else if (oc < -5)
    {
      requestedOrientation = ActivityInfo.SCREEN_
          ORIENTATION_REVERSE_LANDSCAPE
    }
  }
}
```

override fun onAccuracyChanged (sensor: sensor?, accuracy: Int) {

}

}

Q.4 A

1) SSO stands for
→ single sign-in

2) The spectrum used by Bluetooth starts from
→ 2.402 GHz and ends at 2.480 GHz

3) What was the range of Bluetooth?
→ The range of ~~But~~ Bluetooth typically varies depending on the class of the device. For class 1 devices, the range can be up to 100 meter while for class 2 devices, it's typically around 10 meters.

4) ~~Et~~ BLE stands for
→ Bluetooth low Energy.

DATE:
PAGE:

Q.4 (B)

1) Define SAX parser in Android.

→ ~~SA~~ Simple API for XML parser in Android is a lightweight, event-driven XML parsing mechanism. It ~~reed~~ read XML documents sequentially from start to end, generating events as it encounters XML elements, attributes, and text content. SAX parser doesn't load the entire XML document into memory; making it memory-efficient for parsing large XML files.

2) What is the advantage of using Bluetooth technology?

→ Bluetooth ~~technogy~~ technology offers the advantages of wireless connectivity between devices, allowing ~~us~~ seamless with its universal compatibility, ease of use, low power consumption, versatility, and secure communication; makes ~~Bt~~ Bluetooth a popular choice for various applications, including file sharing, audio streaming, hands-free calling; remote control, and IoT connectivity.

DATE:
PAGE:

Q.4 (C)

1) Difference between SOAP v/s REST web service

→ SOAP (Simple Object Access protocol) and REST (Representational State Transfer) are popular approaches for implementing web services.

* SOAP :-

→ SOAP is a protocol-based approach that uses XML for message exchange.

→ It emphasizes a stateless client-server interaction, where each request from the client contains all the necessary information for the server to fulfill it, without relying on server-side state.

→ It relies on a predefined contract, typically described using web services Description Language to define the structure of messages and operations.

→ SOAP message are usetl usually sent over HTTP ; but other transport protocols like SMTP and JMS are also supported.

* REST :-

→ REST is an architectural style that uses standard HTTP methods for communication and typically exchanges data in lightweight formats like JSON or xml.

→ RESTful APIs are often described using informal doc documentation or standards like OpenAPI rather than formal contracts like WSDL.

→ It leverages the existing infrastructure of the web, making it simpler to implement and consume compared to SOAP.

DATE
PAGE

2) Explain Asyntask in Android.

→ In Android, Asyntask is a class provided by the Android SDK that allows developers to perform background tasks on a separate thread from the main UI thread. It's particularly useful for executing operations that might otherwise block the UI thread and cause the app to become unresponsive, such as network requests, database queries, or file I/O operations.

* When an AsyncTask is executed, it goes through a series of states:-

(i) Pending :- The AsyncTask is created but has not yet been executed.

(ii) Running :- The AsyncTask is executing its background task.

(iii) Finished :- The Asyntask has completed its execute. execution.

DATE:
PAGE:

⇒ methods :-

→ onPreExecute() :-
→ doInBackground(Params...) :-
→ onPostExecute(Result) :-
→ onProgressUpdate(progress...) :-

Q.4 (D)

1) Write a note on xml parsing using DOM with example

⇒ xml parsing using the Document object Model (DOM) is a common approach in Android for processing xml documents. DOM parsing involves loading the entire xml document into memory and representing it as a hierarchical tree structure of nodes, which can then be navigated and manipulated programmatically.

=> ~~Book~~ books.xml

```
<library>
    <book>
        <title> Android Programming </title>
        <author> John smith </author>
        <year> 2022 </year>
    </book>
    <book>
        <title> Java Essentails </title>
        <author> Jane Doe </author>
        <year> 2021 </year>
    </book>
</library>
```

=> => Kotlin (.kt file)

```
class MainActivity : AppCompatActivity()
{
    override fun onCreate(savedInstanceState:
                          Bundle?)
    {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity.xml)
```

DATE
PAGE

```
val inputStream: InputStream = assets.open
                    ("books.xml")

val inputSource = InputSource(inputStream)

val factory = DocumentBuilderFactory.newInstance()
val builder = factory.newDocumentBuilder()
val document: Document = builder.parse(inputSource)

val rootElement: Element = document.documentElement

Log.d("MainActivity", "Root element: " +
                    rootElement.nodeName)

val bookList: NodeList = rootElement.getElementsByTagName("book")
```

```
for (i in 0 until bookList.length) {
{
    val bookNode: Node = bookList.item(i)
    if (bookNode.nodeType == Node.ELEMENT_NODE
    {
        val bookElement = bookNode as Element

        val title = bookElement.getElementsByTagName
                ("title").item(0).textContent

        val author = bookElement.getElementsByTagNam
                ("author").item(0).textContent

        val year = bookElement.getElementsByTagName
                ("year").item(0).textContent

        Log.d("MainActivity", "Title: $title, Author:
                $author, year: $year")
    }
}
}
```

DATE:
PAGE:

⇒ add permission &

<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>

2) Write a note on Android WiFi with example.

→ Android provides comprehensive support for managing wifi connections, enabling developers to build applications that interact with networks for various purposes, such as connecting to available networks, retrieving network information and, performing network operations.

DATE:
PAGE:

⇒ xml file

```
<Relativelayout

    <Button
        android:id = "@+id/button1"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Enable wifi" />

    <Button
        android:id = "@+id/button2"
        android:layout_width = "match_parent"
        android:layout_height = "wrap_content"
        android:text = "Disable wifi"
        android:layout_below = "@+id/button1" />

</Relativelayout>
```

⇒ .kt file

```
class MainActivity : Activity()
{
    private lateinit var enableButton : Button
    private lateinit var disableButton : Button

    override fun onCreate (savedInstanceState:
                           Bundle?)
    {
        super.onCreate (savedInstanceState)

        setContentView (R.layout.activity_main)

        enableButton = findViewById (R.id.button1)

        disableButton = findViewById (R.id.button2)

        enableButton.setOnClickListner
    {

        val wifi = applicationContext.
                   getSystemService (Context.
                   WIFI_SERVICE) as
                   WifiManager
```

```
        wifi.isWifiEnabled = true
    }

    disableButton.setOnClickListener {
        val wifi = applicationContext.getSystemService
                (Context.WIFI_SERVICE) as
                WifiManager

        wifi.isWifiEnabled = false
    }
}

override fun onCreateOptionMenu(menu: Menu):
        Boolean {
    menuInflater.inflate(R.menu.activity_main,
    menu)
    return true
}
}
```

DATE:
PAGE:

=> permission add in manifest.xml

(1) Access_wifi_STATE
(2) INTERNET
(3) CHANGE_WIFI_STATE

Q.5 (A)

1) An image must be referenced only from ___ folders.
→ drawable folders

2) ___ method reposition the camera according to the instructions defined in the update
→ onUpdate()

3) Which debug certificate is required to integrate Google Map into your Android application ?
→ The debug certificate required to integrate google maps into your Android application is generated by the Android SDK tools when you build your application in debug mode

DATE:
PAGE:

4) ___ is a process in which street address is converted into a coordinate (latitude, ~~longitud~~ longitude)

→ Geocoding

Q.5 (B)

1) Explain Satellite Google Map in brief.

→ satellite Google maps provides high-resolution imagery of Earth's surface captured by satellites, offering users a detailed and realistic view of various landscapes, landmarks, and terrain features.

DATE:
PAGE:

2) What is LBS ? and what are the use of EBS ? in brief.

→ LBS stands for location-Based Services. These are services that utilize geographical location information to provide relevant and personalized content or functionality to users. LBS applications leverage technologies such as GPS, wi-fi, cellular networks, and RFID to determine the user's location and deliver location-specific services or information.

DATE:
PAGE:

Q.5 (C)

i) write a step of signing certificate and ProGuard rules in Android studio.

→

* signing certificate :-

i) Generate keystore :- use Build > Generate signed APK > create a new keystore with password and key alias / password.

ii) Configure signing :- In project build.gradle, add signing Configs with your keystore details.

iii) Apply signing :- In app module build.gradle, set signing Config to your signing configuration for release builds.

DATE:
PAGE:

* ProGuard Rules :-

i) Enable ProGuard (optional) :- In app module build.gradle, set minifyEnabled true for release builds.

ii) Create ProGuard Rules file (optional) :- create proguard-rules.pro in src/main to define custom rules for obfuscation.

2) Explain Drawable animation, view animation and property animation. ?

i) Drawable Animation :-

→ Drawable animation involves animating drawable resources, such as images or shapes, to create simple frame-based animations. It typically involves creating a sequence of drawable resources representing different frames of an animation and then playing them sequentially to create the illusion of movement or change.

DATE:
PAGE: 15

ii) View Animation :-

→ View animation, also known as Tween animation, involves animating properties of view objects, such as position, size, rotation, or transparency, to create animations. It uses interpolation to calculate intermediate property values between start and end values over a specified duration.

iii) Property Animation :-

→ Property animation allows animating properties of any object, not just View objects, by directly modifying their values over time. It provides more flexibility and control compared to view animation, as it can animate any property of an object and supports more complex animations, such as fluid motion, transformations, or interactions.

DATE:
PAGE:

Q.5 (D)

1) Write an Android app to add custom makers using Google Map.

~~Step 1:~~

* Step 1:- Create a New Project

* Step 2:- Generating an API key for using Google Maps.

~~Steps 3:~~

→ After generating your API key for Google Maps. we have to add this key to our Project. For adding this key in our app navigate to the value folder > google_maps_api.xml file and at line 23 you have to add your API key in the place of YOUR_API_KEY.

DATE:
PAGE:

* Step 3 :- Adding a custom marker in Google Maps.

→ For adding a custom marker to Google Maps navigate to the app > res > drawable > Right-Click on it > New > Vector Asset Assets and select the icon which we have to show on your Map.

* Step 4 :- Working with the MapsActivity.java file.

* MapActivity.java :-

```
public class MapsActivity
    extends FragmentActivity implements
        OnMapReadyCallback
{
```

DATE:
PAGE:

* MapActivity.java

```
public class MapsActivity extends ~~fragement~~
    FragementActivity implements
        OnMapReadyCallback
{

    private GoogleMap mMap;

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

        SupportMapFragment mapFragment
            =(SupportMapFragment)
            getSupportFragmentManager()
            .findFragmentById(R.id.map);

        mapFragment.getMapAsync(this);

}
```

```
@Override public void onMapReady
        (GoogleMap googleMap)
{
    mMap = googleMap;
    LatLng sydney = new LatLng(-34, 151);

    mMap.addMarker(new MarkerOptions()
      .position(sydney).title("Marker in sydney")
      .icon(BitmapFromVector
        (getApplicationContext(),
          R.drawable.ic_flag));

    mMap.moveCamera(
        cameraUpdateFactory.newLatLng
          (sydney));
}
```

```
private BitmapDescriptor
    BitmapFromVector(Context context,
        int vectorResId)
{

    Drawable vectorDrawable =
        ContextCompat.getDrawable
        (context, vectorResId);

    vectorDrawable.setBounds(
        0, 0, vectorDrawable.getIntrinsicWidth(),
        vectorDrawable.getIntrinsicHeight());

    Bitmap bitmap = Bitmap.createBitmap
    (
        vectorDrawable.getIntrinsicWidth(),
        vectorDrawable.getIntrinsicHeight(),
        Bitmap.Config.ARGB_8888);

    Canvas canvas = new Canvas(bitmap);

    vectorDrawable.draw(canvas);

    return BitmapDescriptorFactory.
        fromBitmap(bitmap); }}
```

DATE:
PAGE:

2) write an Android app for blink, bounce, rotate, zoom in / out text.

→ activity_main.xml

```
<RelativeLayout

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Animated Text"
        android:textSize="24sp"
        android:textColor="@android:color/black"
        android:layout_centerInParent="true" />

</RelativeLayout>
```

→ MainActivity.kt

```
class MainActivity : AppCompatActivity()
{
    private lateinit var textView: TextView

    override fun onCreate()
    {
        textView = findViewById(R.id.textView)

        blinkAnimation()
        bounceAnimation()
        rotateAnimation()
        zoomInOutAnimation()
    }

    private fun blinkAnimation()
    {
        val blinkAnimator = ObjectAnimator.ofInt
        (textView, "visibility", view.visible,
            view.invisible)

        blinkAnimator.duration = 500
```

DATE:
PAGE:

blinkAnimator.repeatCount = objectAnimator.INFINITE
blinkAnimator.repeatMode = objectAnimator.REVERSE

blinkAnimator.start()
}

private fun bounceAnimation()
{

val bounceAnimator = objectAnimator.ofFloat
(textViews, "translationY", 0f, -50f, 0f)

bounceAnimator.duration = 1000

bounceAnimator.repeatCount = objectAnimator.INFINITE
bounceAnimator.repeatCount = ObjectAnimator.INFINITE
bounceAnimator.interpolator = BounceInterpolator()

bounceAnimator.start()
}

```
private fun rotateAnimation()
{
    val rotateAnimation = objectAnimator.ofFloat
        (textView, "rotation", 0f, 360f)

    rotateAnimator.duration = 2000

    rotateAnimator.repeatCount = objectAnimator.INFINITE
    rotateAnimation.interpolator = LinearInterpolation()

    rotateAnimator.start()
}

private fun zoomInOutAnimation()
{
    val zoomInOutAnimator = objectAnimator.ofFloat
        (textView, "scaleX", 1f, 1.5f, 1f)

    zoomInOutAnimator.duration = 2000

    zoomInOutAnimator.repeatCount = objectAnimator.INFINITE
    zoomInOutAnimator.repeatMode = objectAnimator.REVERSE

    zoomInOutAnimator.interpolator = DecelerateInterpolator()
    zoomInOutAnimator.start() }}
```