### Factors encouraging Android native application development:

- Android has a large user base and market share.
- Provides access to device features and performance optimization.

### Compilation process of an Android application:

- Converts source code into DEX files, then packaged into an APK.
- Signed and optimized for deployment.

### Importance of Gradle in Android application development:

- It manages dependencies, ensures proper versioning, and automates the build process.

### Purpose of AndroidManifest.xml file:

- It contains essential information about the app, including its components and permissions required.

### Architecture of Android OS:

- Linux Kernel: Core system services and hardware abstraction.
- Libraries & Android Runtime: C/C++ libraries, ART runtime.
- Application Framework: API services for app developers.
- Applications: User-facing apps.

### Components of Android OS:

- Activity: Represents a single screen in an app.
- Service: Background processes without user interface.
- Intent: Messaging object to request actions from other components.
- Jetpack Compose: Toolkit for building native UIs in Kotlin.
- Gradle: Build automation tool used in Android.

### AdapterView in Android:

- A view that binds data to layout elements.
- Example: ListView, GridView.

### Kotlin code to populate a ListView:

```
val listView: ListView =
findViewById(R.id.listView)
    val items = arrayOf("Item 1", "Item 2", "Item
3")
    val adapter = ArrayAdapter(this,
android.R.layout.simple_list_item_1, items)
    listView.adapter = adapter
```

### Importance of Dependency Management in Android:

- Ensures proper library versioning and project stability.
- Gradle is the platform used for dependency management.

### Kotlin code for various tasks:

Difference of two numbers:

```
fun difference(a: Int, b: Int): Int = a - b
```

Starting an activity and sharing data:

```
val intent = Intent(this,
SecondActivity::class.java)
    intent.putExtra("key", "value")
    startActivity(intent)
```

Accepting values in another activity:

```
val value = intent.getStringExtra("key")
```

### Kotlin code for salary calculation with tax:

```
val salary: Double =
salaryInput.text.toString().toDouble()
    val tax = when {
        salary > 1000000 -> 0.10
        salary > 500000 -> 0.07
        salary > 100000 -> 0.05
        else -> 0.0
    }
    val taxAmount = salary * tax
    val salaryAfterTax = salary - taxAmount
    taxOutput.text = taxAmount.toString()
    salaryAfterTaxOutput.text =
salaryAfterTax.toString()
```

### Kotlin code to create and manipulate a database:

Create 'reporter' database and 'user' table:

```
val dbHelper = object :
SQLiteOpenHelper(context, "reporter.db", null,
```

```
1) {
        override fun onCreate(db: SQLiteDatabase)
{
            db.execSQL("CREATE TABLE user (id
INTEGER PRIMARY KEY, user_name TEXT,
reporter_category INTEGER, city TEXT)")
        }
        override fun onUpgrade(db:
SQLiteDatabase, oldVersion: Int, newVersion:
Int) {}
    }

Insert values into the table:
val values = ContentValues().apply {
        put("user_name", "John")
        put("reporter_category", 1)
        put("city", "New York")
    }
    db.insert("user", null, values)

Update values in the table:
val updatedValues = ContentValues().apply {
        put("city", "Los Angeles")
    }
    db.update("user", updatedValues, "id = ?",
arrayOf("1"))
```

### Kotlin code for bill calculator application:

```
val units = unitsInput.text.toString().toInt()
    val rate = when {
        units >= 120 -> 65.0
        units >= 60 -> 40.0
        else -> 25.0
    }
    val billAmount = units * rate
    billOutput.text = billAmount.toString()
```

### Usage and abstract methods of SQLiteOpenHelper:

- Provides an interface for creating, updating, and managing databases.
- Abstract methods:
  - onCreate: Called when the database is first created.
  - onUpgrade: Called when the database needs to be upgraded.

### Android services:

- Used for background processing without a UI.
- Suitable for tasks like playing music, fetching data.

### BroadcastReceiver in Android:

- Receives and handles broadcast messages from other apps or the system.
- Overridable methods include onReceive, used to handle the broadcast.

### Kotlin code to populate a Spinner using an array adapter:

```
val spinner: Spinner =
findViewById(R.id.spinner)
    val adapter = ArrayAdapter(this,
android.R.layout.simple_spinner_item, items)

adapter.setDropDownViewResource(android.R.l
ayout.simple_spinner_dropdown_item)
    spinner.adapter = adapter
```

### Application permissions in Android:

- Purpose: Control access to sensitive data and device features.
- Runtime permissions: Required for permissions that affect user privacy.

### Permissions required for accessing the camera:

- android.permission.CAMERA

### Kotlin code to handle runtime permissions for accessing the camera:

```
if (ContextCompat.checkSelfPermission(this,
Manifest.permission.CAMERA) !=
PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(this,
arrayOf(Manifest.permission.CAMERA),
REQUEST_CAMERA_PERMISSION)
    }
```