PRACTICAL-1

AIM :

i. **Write a program using Kotlin to implement control structures and loops.**
ii. **Write a program to implement object-oriented concepts in Kotlin.**

i. **Write a program using Kotlin to implement control structures and loops.**

A. if Expression

```
fun main(args: Array<String>) {
   val number = -10
   if (number > 0) {
      print("Positive number")
   } else {
      print("Negative number")
   }
}
```

O/p

```
Negative number
```

B. if..else..if Ladder

```
fun main(args: Array<String>) {
   val number = 0
   val result = if (number > 0)
      "positive number"
   else if (number < 0)
      "negative number"
   else
      "zero"
   println("number is $result")
}
```

O/p

```
number is zero
```

C. Nested if Expression

```
fun main(args: Array<String>) {
   val n1 = 3
   val n2 = 5
   val n3 = -2
   val max = if (n1 > n2) {
      if (n1 > n3)
         n1
      else
         n3
   } else {
      if (n2 > n3)
         n2
      else
         n3
   }
   println("max = $max")
}
```

O/p
max=5

D. Kotlin for Loop

```
fun main(args: Array<String>) {
    var text= "Kotlin"
   for (letter in text) {
        println(letter)
    }
}
```

O/p

```
K
o
t
l
i
n
```

E. Kotlin while Loop

```
// Program to compute the sum of natural numbers from 1 to 100.
fun main(args: Array<String>) {
    var sum = 0
    var i = 100
    while (i != 0) {
        sum += i     // sum = sum + i;
        --i
    }
    println("sum = $sum")
}
```

O/p
sum=5050

F. Kotlin do...while Loop

```
fun main(args: Array<String>) {
    var sum: Int = 0
    var input: String
    do {
        print("Enter an integer: ")
        input = readLine()!!
        sum += input.toInt()
    } while (input != "0")
    println("sum = $sum")
}
```

O/p

```
Enter an integer: 4
Enter an integer: 3
Enter an integer: 2
Enter an integer: -6
Enter an integer: 0
sum = 3
```

ii. Write a program to implement object-oriented concepts in Kotlin.

Kotlin Class and Object

Kotlin supports both object oriented programming (OOP) as well as functional programming. Object oriented programming is based on real time objects and classes. Kotlin also support pillars of OOP language such as encapsulation, inheritance and polymorphism.

Kotlin Class

Kotlin class is similar to Java class, a class is a blueprint for the objects which have common properties. Kotlin classes are declared using keyword class. Kotlin class has a class header which specifies its type parameters, constructor etc. and the class body which is surrounded by curly braces.

Kotlin Object

Object is real time entity or may be a logical entity which has state and behavior. It has the characteristics:

state: it represents value of an object.

behavior: it represent the functionality of an object.

Object is used to access the properties and member function of a class. Kotlin allows to create multiple object of a class.

create an example, which access the class property and member function using . operator.

```
class Account {
   var acc_no: Int = 0
   var name: String =  ""
   var amount: Float = 0.toFloat()
   fun insert(ac: Int,n: String, am: Float ) {
      acc_no=ac
      name=n
      amount=am
      println("Account no: ${acc_no} holder :${name} amount :${amount}")
   }
     fun deposit() {
      //deposite code
   }
     fun withdraw() {
      // withdraw code
   }
     fun checkBalance() {
      //balance check code
    }
 }
fun main(args: Array<String>){
   Account()
   var acc= Account()
   acc.insert(832345,"Ankit",1000f) //accessing member function
   println("${acc.name}") //accessing class property
}
```

O/p

```
Account no: 832345 holder :Ankit amount :1000.0
Ankit
```

PRACTICAL-2

AIM :

i. **Write a program using Kotlin to describe type check.**
ii. **Write a program to implement lambda expression in Kotlin.**
iii. **Write a program using Kotlin to find area of circle using function.**

i. Write a program using Kotlin to describe type check.

```
fun main() {
   val obj1: Any = "Hello"
   val obj2: Any = 42
   val obj3: Any = arrayListOf(1, 2, 3)

   checkType(obj1)
   checkType(obj2)
   checkType(obj3)
}

fun checkType(obj: Any) {
   when (obj) {
      is String -> println("$obj is a String")
      is Int -> println("$obj is an Int")
      is List<*> -> println("$obj is a List")
      else -> println("$obj is of unknown type")
   }
}
```

O/p

```
Hello is a String
42 is an Int
[1, 2, 3] is a List
```

ii. Write a program to implement lambda expression in Kotlin.

```
// Define a data class to represent a book
data class Book(val title: String, val author: String, val year: Int)

fun main() {
   // Create a list of books
   val books = listOf(
      Book("1984", "George Orwell", 1949),
      Book("To Kill a Mockingbird", "Harper Lee", 1960),
      Book("The Catcher in the Rye", "J.D. Salinger", 1951),
      Book("The Great Gatsby", "F. Scott Fitzgerald", 1925),
      Book("Pride and Prejudice", "Jane Austen", 1813)
   )

   // Filter the books based on a condition using a lambda expression
   val modernBooks = books.filter { it.year > 1950 }

   // Print the titles of modern books
   println("Modern books:")
   modernBooks.forEach { println("${it.title} by ${it.author}") }
```

```
    // Sort the books based on the year of publication using a lambda expression
    val sortedBooks = books.sortedBy { it.year }

    // Print the titles of sorted books
    println("\nBooks sorted by year:")
    sortedBooks.forEach { println("${it.title} (${it.year})") }
}
```

O/p

```
Modern books:
To Kill a Mockingbird by Harper Lee
The Catcher in the Rye by J.D. Salinger

Books sorted by year:
Pride and Prejudice (1813)
The Great Gatsby (1925)
1984 (1949)
The Catcher in the Rye (1951)
To Kill a Mockingbird (1960)
```

iii. Write a program using Kotlin to find area of circle using function.

```
import kotlin.math.PI

fun main() {
    val radius = 10.0

    if (radius != null && radius > 0) {
        val area = calculateArea(radius)
        println("The area of the circle with radius $radius is $area")
    } else {
        println("Invalid input. Please enter a valid positive number for radius.")
    }
}

fun calculateArea(radius: Double): Double {
    return PI * radius * radius
}
```

O/p

```
The area of the circle with radius 10.0 is 314.1592653589793
```

PRACTICAL-3

AIM :

i. Create an application to create Image Flipper and Image Gallery. On click on the image display the information about the image.
ii. Create an application to use Gridview for shopping cart application

i. Create an application to create Image Flipper and Image Gallery. On click on the image display the information about the image.

Open Android Studio and create a new project with an empty activity. Wait for Android Studio to finish creating your project, and then open app > res > layout > activity_main.xml.

Save image file
\app\src\main\res\drawable

MainActivity.java

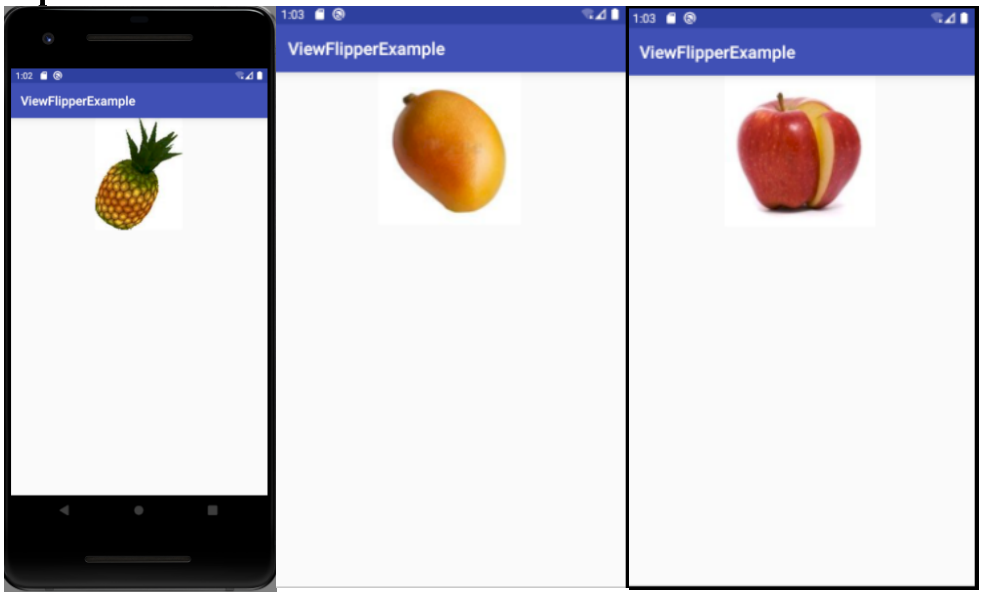
```
package com.example.ip_d.viewflipperexample;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageView;
import android.widget.ViewFlipper;

public class MainActivity extends AppCompatActivity {

  int[] images = {R.drawable.apple, R.drawable.pineapple, R.drawable.litchi,
R.drawable.mango, R.drawable.banana};     // array of images

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);

     setContentView(R.layout.activity_main);

     // get The references of ViewFlipper
     ViewFlipper simpleViewFlipper = (ViewFlipper)
findViewById(R.id.simpleViewFlipper); // get the reference of ViewFlipper

     // loop for creating ImageView's
     for (int image : images) {
        // create the object of ImageView
        ImageView imageView = new ImageView(this);
        imageView.setImageResource(image); // set image in ImageView
        simpleViewFlipper.addView(imageView); // add the created ImageView in
ViewFlipper
     }
     // Declare in and out animations and load them using AnimationUtils class
     Animation in = AnimationUtils.loadAnimation(this, android.R.anim.slide_in_left);
     Animation out = AnimationUtils.loadAnimation(this, android.R.anim.slide_out_right);
```

```
        // set the animation type's to ViewFlipper
        simpleViewFlipper.setInAnimation(in);
        simpleViewFlipper.setOutAnimation(out);
        // set interval time for flipping between views
        simpleViewFlipper.setFlipInterval(3000);
        // set auto start for flipping between views
        simpleViewFlipper.setAutoStart(true);
    }
}
```

activity_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical">

  <ViewFlipper
     android:id="@+id/simpleViewFlipper"
     android:layout_width="match_parent"
     android:layout_height="wrap_content">

    </ViewFlipper>
</LinearLayout>
```

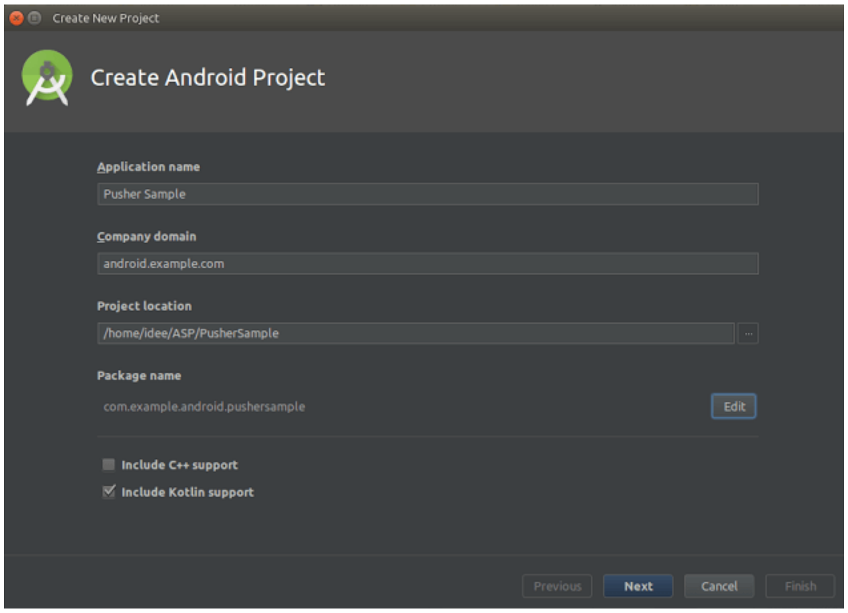
O/p



ii. Create an application to use Gridview for shopping cart application

Configuring your project

Open Android Studio, create a new project. Insert the name of your app and company domain name then select the Include Kotlin support checkbox to enable Kotlin in the project.



Application name
Pusher Sample

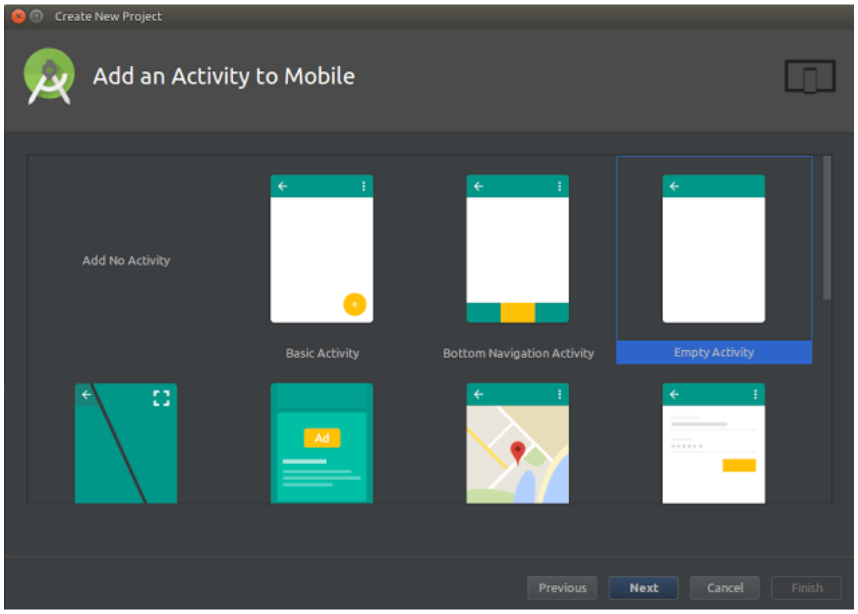
Company domain
android.example.com

Project location
/home/idee/ASP/PusherSample

Package name
com.example.android.pushersample

Include C++ support [ ]
Include Kotlin support [✓]

For this article, we will set the minimum supported Android version at 4.03 (API 15). Next, choose an empty activity template and click on Finish.



Then head over to your ../app/build.gradle file and paste this inside the dependencies block, as we'll be using these dependencies in this tutorial

```
//..app/build.gradle
 implementation 'com.google.code.gson:gson:2.8.2'
 implementation 'com.squareup.picasso:picasso:2.71828'
 implementation 'com.squareup.retrofit2:retrofit:2.4.0'
 implementation 'com.squareup.retrofit2:converter-gson:2.3.0'
 implementation 'com.squareup.retrofit2:converter-scalars:2.3.0'
 implementation 'com.android.support:design:28.0.0'
 implementation 'com.android.support:cardview-v7:28.0.0'
 implementation 'com.android.support:recyclerview-v7:28.0.0'
```

- **Retrofit: We will need the** Retrofit **library (a "type-safe HTTP client") to enable us send messages to our remote server which we will build later on.**
- **Picasso: Picasso is "A powerful image downloading and caching library for Android"**

Also, amend your styles.xml like the following. This should enable us to use a toolbar inside our application.

```
//..app/src/main/res/values/styles.xml

 <resources>

     <!-- Base application theme. -->
     <style name="AppTheme" parent="Theme.AppCompat.NoActionBar">
         <!-- Customize your theme here. -->
         <item name="colorPrimary">@color/colorPrimary</item>
         <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
         <item name="colorAccent">@color/colorAccent</item>
     </style>

 </resources>
```

Build your product model, adapter and layout

Our product should have a unique identifier, a price, a name, a description and a set of images if possible. Now that we know the structure of our product item, let's define its model. We'll build our product entity using a Kotlin data class.

Create a Product.kt file, then copy and paste the following piece of code inside:

```
//..app/src/main/java/yourPackage/Product.kt
 import com.google.gson.annotations.SerializedName

 data class Product(
    @SerializedName("description")
    var description: String? = null,

    @SerializedName("id")
    var id: Int? = null,

    @SerializedName("name")
    var name: String? = null,

    @SerializedName("price")
    var price: String? = null,

    @SerializedName("photos")
    var photos: List<Photo> = arrayListOf()
 )
```

As our product has a set of photos, we'll also define its entity. Create a Photo.kt file, then paste the following code inside as well:

```
//..app/src/main/java/yourPackage/Photo.kt
 import com.google.gson.annotations.SerializedName

 data class Photo(
    @SerializedName("filename")
    var filename: String? = null
 )
```

Next, we'll build our product adapter responsible to handle the display of our products list.

Create a ProductAdapter.kt file and paste the following inside:

```
//..app/src/main/java/yourPackage/ProductAdapter.kt

 import android.annotation.SuppressLint
 import android.content.Context
 import android.support.design.widget.Snackbar
 import android.support.v7.widget.RecyclerView
 import android.view.LayoutInflater
 import android.view.View
 import android.view.ViewGroup
```

```
   import android.widget.Toast
   import com.squareup.picasso.Picasso
   import kotlinx.android.synthetic.main.activity_main.*
   import kotlinx.android.synthetic.main.product_row_item.view.*

   class ProductAdapter(var context: Context, var products: List<Product> =
arrayListOf()) :
      RecyclerView.Adapter<ProductAdapter.ViewHolder>() {
      override fun onCreateViewHolder(p0: ViewGroup, p1: Int):
ProductAdapter.ViewHolder {
         // The layout design used for each list item
         val view = LayoutInflater.from(context).inflate(R.layout.product_row_item, null)
         return ViewHolder(view)

      }
         // This returns the size of the list.
      override fun getItemCount(): Int = products.size

      override fun onBindViewHolder(viewHolder: ProductAdapter.ViewHolder, position:
Int) {
       //we simply call the `bindProduct` function here
         viewHolder.bindProduct(products[position])
      }

      class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {

         // This displays the product information for each item
         fun bindProduct(product: Product) {

            itemView.product_name.text = product.name
            itemView.product_price.text = "$${product.price.toString()}"
            Picasso.get().load(product.photos[0].filename).fit().into(itemView.product_image)
         }

      }

   }
```

Next, let's create our product item layout. This layout file contains:

- an ImageView to display the product image
- two TextView one to display the product name and the other for the product price.

All these widgets are wrapped inside a CardView to add a shadow and a radius to the layout .

Create a product_row_item file and paste the following inside. This layout is responsible for handling the view of a single item of our list.

```
  //../app/src/main/java/res/layout/product_row_item.xml

   <?xml version="1.0" encoding="utf-8"?>
```

```xml
<android.support.v7.widget.CardView
    xmlns:card_view="http://schemas.android.com/tools"
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    app:cardUseCompatPadding="true"
    android:layout_margin="4dp"
    app:cardBackgroundColor="@android:color/white"
    app:cardCornerRadius="4dp"
    android:background="?attr/selectableItemBackground"
    app:cardElevation="3dp"
    android:foreground="?attr/selectableItemBackground"
    card_view:cardElevation="4dp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

  <LinearLayout
      android:orientation="vertical"
      android:layout_width="match_parent"
      android:layout_height="match_parent">

    <ImageView
        android:id="@+id/product_image"
        android:layout_width="match_parent"
        android:layout_height="140dp"/>

    <LinearLayout
        android:padding="10dp"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">


      <TextView
          android:textColor="@android:color/black"
          android:textSize="22sp"
          android:layout_marginBottom="12dp"
          android:id="@+id/product_name"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"/>


      <TextView
          android:textSize="19sp"
          android:textColor="@android:color/black"
          android:id="@+id/product_price"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"/>
    </LinearLayout>

    <ImageButton
        android:id="@+id/addToCart"
        android:paddingHorizontal="16dp"
        android:tint="@android:color/white"
```

```
                android:paddingVertical="4dp"
                android:src="@drawable/ic_add_shopping"
                android:layout_gravity="end"
                android:background="@color/colorAccent"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                card_view:targetApi="o"/>

        </LinearLayout>

    </android.support.v7.widget.CardView>
```

These are the links to get the drawable icons we used in our layout : ic_add_shopping **and** ic_shopping_basket**. They are to paste in ../app/src/main/res/drawable folder.**

Preparing API calls

We'll make calls to an external API to get our products data.

Create an APIConfig.kt file. This class gives us an instance of Retrofit for our network calls:

```
   //..app/src/main/java/yourPackage/APIConfig.kt
    import android.content.Context
    import okhttp3.OkHttpClient
    import retrofit2.Retrofit
    import retrofit2.converter.gson.GsonConverterFactory
    import com.google.gson.GsonBuilder
    import retrofit2.converter.scalars.ScalarsConverterFactory

    object APIConfig {

        val BASE_URL = "https://all-spices.com/api/products/"

        private var retrofit: Retrofit? = null

        var gson = GsonBuilder()
            .setLenient()
            .create()

        fun getRetrofitClient(context: Context): Retrofit {

            val okHttpClient = OkHttpClient.Builder()
                .build()

            if (retrofit == null) {
                retrofit = Retrofit.Builder()
                    .baseUrl(BASE_URL)
                    .client(okHttpClient)
                        .addConverterFactory(ScalarsConverterFactory.create())
                    .addConverterFactory(GsonConverterFactory.create(gson))
                    .build()
            }
            return retrofit!!
        }
    }
```

Next, create an API Interface file in the src/main/java/yourPackage folder called ApiService.kt. This interface is used to define endpoints to be used during network calls. For this application, we will create just one endpoint:

```
 import retrofit2.Call
  import retrofit2.http.*

  interface APIService {
     @Headers("Content-Type: application/json", "Accept: application/json")
     @GET("bestRated")
     fun getProducts(
     ): Call<List<Product>>

  }
```

Listing products

For listing products, we'll need a recycler view (a recycler view is a widget for listing a list of items, as it happens our products list). Now, move on to your src/main/java/res/layout/activity_main.xml file, amend it like the following:

```
 <?xml version="1.0" encoding="utf-8"?>
  <android.support.design.widget.CoordinatorLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        android:layout_height="match_parent"
        android:background="#fffffa"
        android:id="@+id/coordinator"
        android:layout_width="match_parent">


     <android.support.design.widget.AppBarLayout
           android:background="@android:color/transparent"
           android:fitsSystemWindows="true"
           android:layout_width="match_parent"
           android:layout_height="?attr/actionBarSize">

        <android.support.v7.widget.Toolbar
              android:id="@+id/toolbar"
              app:titleTextColor="@color/colorAccent"
              app:title="Shopping List"
              android:background="@android:color/white"
              android:layout_width="match_parent"
              android:layout_height="?attr/actionBarSize">

        </android.support.v7.widget.Toolbar>
```

```
    </android.support.design.widget.AppBarLayout>


    <android.support.v4.widget.SwipeRefreshLayout
        android:id="@+id/swipeRefreshLayout"
        app:layout_behavior="@string/appbar_scrolling_view_behavior"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

      <android.support.v7.widget.RecyclerView
          android:id="@+id/products_recyclerview"
          android:layout_width="match_parent"
          android:layout_height="match_parent"/>

    </android.support.v4.widget.SwipeRefreshLayout>

    <android.support.design.widget.FloatingActionButton
        android:id="@+id/showBasket"
        android:src="@drawable/ic_shopping_basket"
        android:tint="@android:color/white"
        android:layout_margin="16dp"
        android:layout_gravity="bottom|end"
        app:fabSize="normal"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

  </android.support.design.widget.CoordinatorLayout>
```

The above layout contains a recycler view which itself is wrapped in a SwipeRefreshLayout widget. We also add a button for adding items to our shopping cart, but this will be handled in the second part of the tutorial.

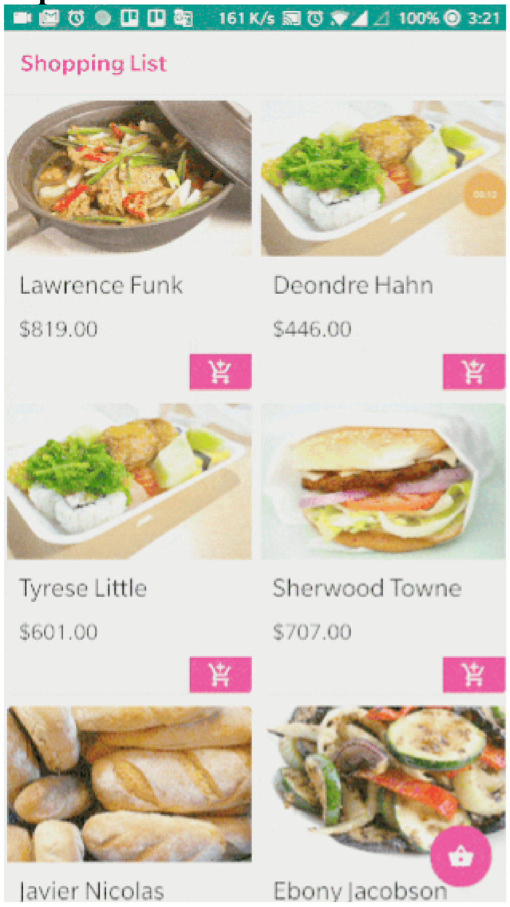
Next, move on to your src/main/MainActvity.kt file, and amend like the following:

```
 //..app/src/main/java/yourPackage/MainActivity.kt

  import android.content.Intent
  import android.support.v7.app.AppCompatActivity
  import android.os.Bundle
  import android.support.v4.content.ContextCompat
  import android.support.v7.widget.StaggeredGridLayoutManager
  import android.util.Log
  import android.widget.Toast
  import kotlinx.android.synthetic.main.activity_main.*
  import retrofit2.Call
  import retrofit2.Response

  class MainActivity : AppCompatActivity() {

    private lateinit var apiService: APIService
    private lateinit var productAdapter: ProductAdapter

    private var products = listOf<Product>()
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        setContentView(R.layout.activity_main)

        setSupportActionBar(toolbar)
        apiService = APIConfig.getRetrofitClient(this).create(APIService::class.java)

        swipeRefreshLayout.setColorSchemeColors(ContextCompat.getColor(this,
R.color.colorPrimary))

        swipeRefreshLayout.isRefreshing = true

        // assign a layout manager to the recycler view
        products_recyclerview.layoutManager = StaggeredGridLayoutManager(2,
StaggeredGridLayoutManager.VERTICAL)

        getProducts()

    }

    fun getProducts() {
        apiService.getProducts().enqueue(object : retrofit2.Callback<List<Product>> {
            override fun onFailure(call: Call<List<Product>>, t: Throwable) {

                print(t.message)
                Log.d("Data error", t.message)
                Toast.makeText(this@MainActivity, t.message,
Toast.LENGTH_SHORT).show()

            }

            override fun onResponse(call: Call<List<Product>>, response:
Response<List<Product>>) {

                swipeRefreshLayout.isRefreshing = false
                products = response.body()!!

                productAdapter = ProductAdapter(this@MainActivity, products)

                products_recyclerview.adapter = productAdapter
                productAdapter.notifyDataSetChanged()

            }

        })
    }

  }
```

We declared an APIService instance, a ProductAdapter instance we'll initialize later. Next, we initialized the list to hold the products: private var products = listOf<Product>().

Then, in the onCreate method, we initialized our APIService instance, configured our swipe refresh layout and made it refreshing, and assigned a proper layout to our recycler view.In the getProducts methods, we made an API call to fetch our products, if everything gets well we first disable the swipe refresh layout, then assign the result to our products list, initialised our product adapter, assigned the adapter to the recycler view, and tell the adapter data its state has changed. Otherwise, we just logged the error for debugging purpose.

O/p



PRACTICAL-4

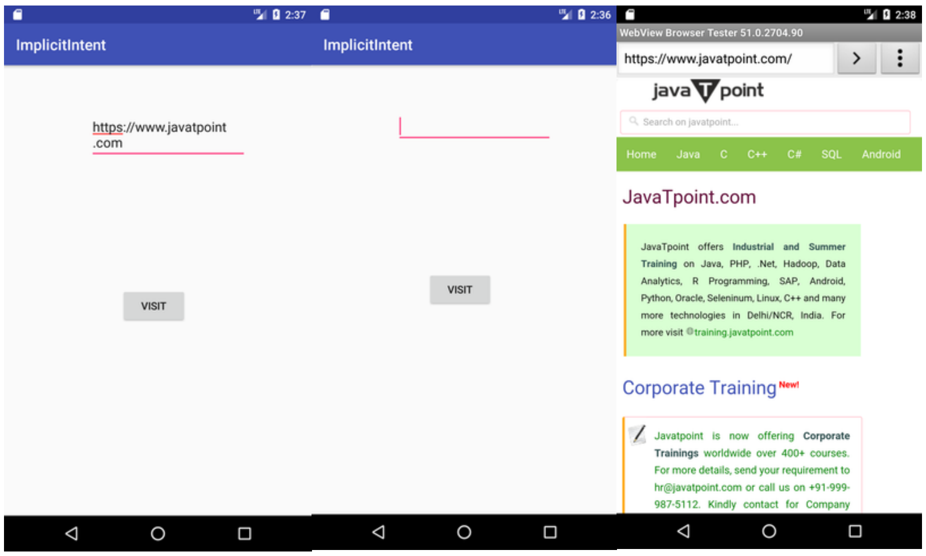
AIM :
i. Create an Android application to demonstrate implicit and explicit intents
ii. Create an application to demonstrate shared preferences

i. Create an Android application to demonstrate implicit and explicit intents.
Android Intent is the message that is passed between components such as activities, content providers, broadcast receivers, services etc.
It is generally used with startActivity() method to invoke activity, broadcast receivers etc.
The dictionary meaning of intent is intention or purpose. So, it can be described as the intention to do action.
There are two types of intents in android: implicit and explicit.
1) Implicit Intent
Implicit Intent doesn't specifiy the component. In such case, intent provides information of available components provided by the system that is to be invoked.
2) Explicit Intent
Explicit Intent specifies the component. In such case, intent provides the external class to be invoked.
implicit intent that displays a web page.
activity_main.xml
```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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="example.javatpoint.com.implicitintent.MainActivity">
    <EditText
      android:id="@+id/editText"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginEnd="8dp"
      android:layout_marginStart="8dp"
      android:layout_marginTop="60dp"
      android:ems="10"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintHorizontal_bias="0.575"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent" />
    <Button
      android:id="@+id/button"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginRight="8dp"
      android:layout_marginLeft="156dp"
      android:layout_marginTop="172dp"
      android:text="Visit"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintHorizontal_bias="0.0"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toBottomOf="@+id/editText" />
</android.support.constraint.ConstraintLayout>
```

MainActivity.java

```
import android.content.Intent;
import android.net.Uri;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity {

    Button button;
    EditText editText;

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

        button = findViewById(R.id.button);
        editText =  findViewById(R.id.editText);

        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String url=editText.getText().toString();
                Intent intent=new Intent(Intent.ACTION_VIEW, Uri.parse(url));
                startActivity(intent);
            }
        });
    }
}
```

O/p



JavaTpoint.com

JavaTpoint offers Industrial and Summer Training on Java, PHP, .Net, Hadoop, Data Analytics, R Programming, SAP, Android, Python, Oracle, Seleninum, Linux, C++ and many more technologies in Delhi/NCR, India. For more visit training.javatpoint.com

Corporate Training New!

Javatpoint is now offering Corporate Trainings worldwide over 400+ courses. For more details, send your requirement to hr@javatpoint.com or call us on +91-999-987-5112. Kindly contact for Company

Android explicit example that calls one activity from another and vice versa.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>

<android.support.constraint.ConstraintLayout
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="example.javatpoint.com.explicitintent.FirstActivity">

    <TextView

      android:layout_width="wrap_content"

      android:layout_height="wrap_content"

      android:layout_marginEnd="8dp"

      android:layout_marginStart="8dp"

      android:layout_marginTop="8dp"

      android:text="First Activity"

      app:layout_constraintBottom_toBottomOf="parent"

      app:layout_constraintEnd_toEndOf="parent"

      app:layout_constraintHorizontal_bias="0.454"

      app:layout_constraintLeft_toLeftOf="parent"

      app:layout_constraintRight_toRightOf="parent"

      app:layout_constraintStart_toStartOf="parent"

      app:layout_constraintTop_toTopOf="parent"

      app:layout_constraintVertical_bias="0.06" />

    <Button

      android:id="@+id/button"

      android:layout_width="wrap_content"

      android:layout_height="wrap_content"
```

```
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="392dp"
        android:onClick="callSecondActivity"
        android:text="Call second activity"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
  </android.support.constraint.ConstraintLayout>
```

MainActivityOne.java

```
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
public class FirstActivity extends AppCompatActivity {
    @Override
   protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_first);
   }
   public void callSecondActivity(View view){
        Intent i = new Intent(getApplicationContext(), SecondActivity.class);
        i.putExtra("Value1", "Android By Javatpoint");
        i.putExtra("Value2", "Simple Tutorial");
        // Set the request code to any code you like, you can identify the
        // callback via this code
        startActivity(i);
   }  }
```

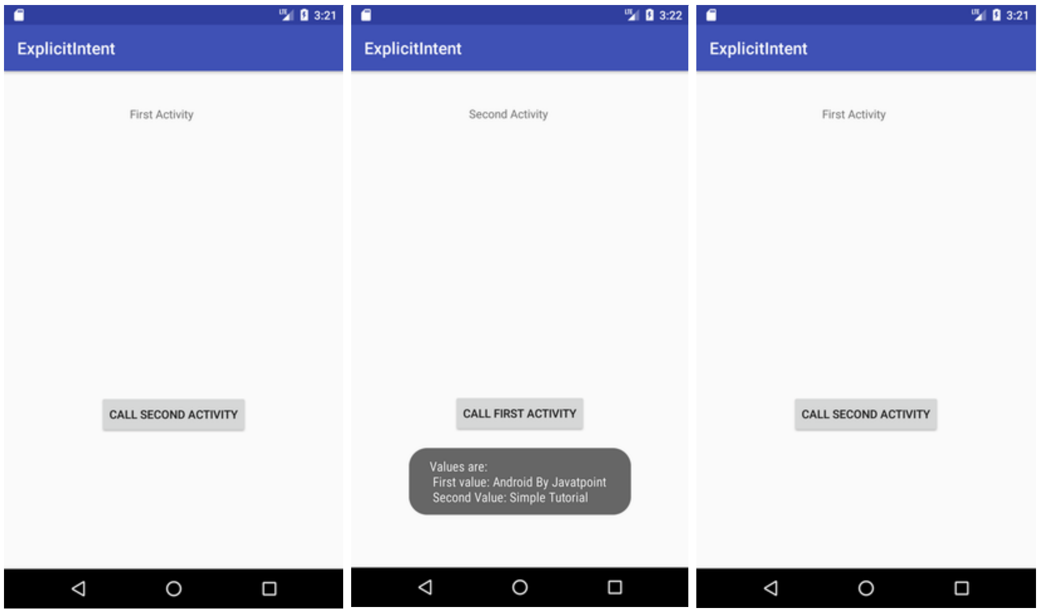
activitytwo_main.xml

```
<?xml version="1.0" encoding="utf-8"?>

<android.support.constraint.ConstraintLayout
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="example.javatpoint.com.explicitintent.SecondActivity">


    <TextView

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_marginEnd="8dp"

        android:layout_marginStart="8dp"

        android:layout_marginTop="8dp"

        android:text="Second Activity"

        app:layout_constraintBottom_toBottomOf="parent"

        app:layout_constraintEnd_toEndOf="parent"

        app:layout_constraintHorizontal_bias="0.454"

        app:layout_constraintLeft_toLeftOf="parent"

        app:layout_constraintRight_toRightOf="parent"

        app:layout_constraintStart_toStartOf="parent"

        app:layout_constraintTop_toTopOf="parent"

        app:layout_constraintVertical_bias="0.06" />

      <Button

        android:id="@+id/button"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="392dp"
        android:onClick="callFirstActivity"
        android:text="Call first activity"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</android.support.constraint.ConstraintLayout>
```

MainActivityTwo.java

```
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;
 public class SecondActivity extends AppCompatActivity {
    @Override
   protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
        Bundle extras = getIntent().getExtras();
        String value1 = extras.getString("Value1");
        String value2 = extras.getString("Value2");
        Toast.makeText(getApplicationContext(),"Values are:\n First value: "+value1+
                "\n Second Value: "+value2, Toast.LENGTH_LONG).show();
   }
   public void callFirstActivity(View view){
        Intent i = new Intent(getApplicationContext(), FirstActivity.class);
        startActivity(i);

    }

}
```

Output:



```
Values are:
First value: Android By Javatpoint
Second Value: Simple Tutorial
```

ii. Create an application to demonstrate shared preferences

Shared Preferences are suitable for different situations. For example, when the user's settings need to be saved or to store data that can be used in different activities within the app. As you know, onPause() will always be called before your activity is placed in the background or destroyed, So for the data to be saved persistently, it's preferred to save it in onPause(), which could be restored in onCreate() of the activity. The data stored using shared preferences are kept private within the scope of the application. However, shared preferences are different from that activity's instance state.

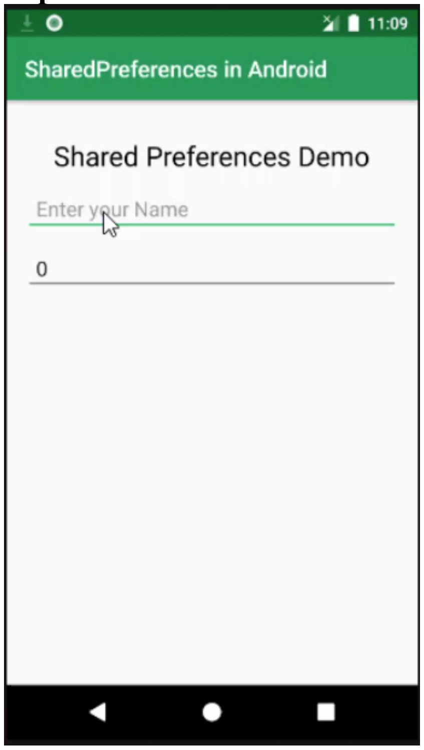
activity_main.xml

```
<?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"
        tools:ignore="HardcodedText">

        <TextView
                android:id="@+id/textview"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_centerHorizontal="true"
                android:layout_marginTop="32dp"
                android:text="Shared Preferences Demo"
                android:textColor="@android:color/black"
                android:textSize="24sp" />

        <!--EditText to take the data from the user and save the data in SharedPreferences-->
        <EditText
                android:id="@+id/edit1"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_below="@+id/textview"
                android:layout_marginStart="16dp"
                android:layout_marginTop="8dp"
                android:layout_marginEnd="16dp"
                android:hint="Enter your Name"
                android:padding="10dp" />

        <!--EditText to take the data from the user and save the data in SharedPreferences-->
        <EditText
                android:id="@+id/edit2"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_below="@+id/edit1"
                android:layout_marginStart="16dp"
                android:layout_marginTop="8dp"
                android:layout_marginEnd="16dp"
                android:hint="Enter your Age"
                android:inputType="number"
                android:padding="10dp" />
</RelativeLayout>
```

MainActivity.kt

```
import android.os.Bundle
import android.widget.EditText
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {
        private lateinit var name: EditText
        private lateinit var age: EditText

        override fun onCreate(savedInstanceState: Bundle?) {
                super.onCreate(savedInstanceState)
                setContentView(R.layout.activity_main)
                name = findViewById(R.id.edit1)
                age = findViewById(R.id.edit2)
        }

        // Fetch the stored data in onResume() Because this is what will be called when the app
opens again
        override fun onResume() {
                super.onResume()
                // Fetching the stored data from the SharedPreference
                val sh = getSharedPreferences("MySharedPref", MODE_PRIVATE)
                val s1 = sh.getString("name", "")
                val a = sh.getInt("age", 0)
                // Setting the fetched data in the EditTexts
                name.setText(s1)
                age.setText(a.toString())
        }

        // Store the data in the SharedPreference in the onPause() method
        // When the user closes the application onPause() will be called and data will be stored
        override fun onPause() {
                super.onPause()
                // Creating a shared pref object with a file name "MySharedPref" in private mode
                val sharedPreferences = getSharedPreferences("MySharedPref",
MODE_PRIVATE)
                val myEdit = sharedPreferences.edit()

                // write all the data entered by the user in SharedPreference and apply
                myEdit.putString("name", name.text.toString())
                myEdit.putInt("age", age.text.toString().toInt())
                myEdit.apply()
        }
}
```

O/p



PRACTICAL-5

AIM :

i. Create an Android application to design screens using different layouts and UI including Button, Edittext, Textview, Radio Button etc.
ii. Write an android application demonstrating response to event/user interaction for
a. Checkbox
b. Radio button
c. Button
d. Spinner

Create a
Open Android Studio and create a new project with an empty activity. Wait for Android Studio to finish creating your project, and then open app > res > layout > activity_main.xml. This file defines the layout for the user interface (UI).

A UI in Android is defined in XML files. The easiest way to build a UI in Android Studio is with the Android Studio Layout Editor. The Layout Editor writes the XML for you as you drag and drop views to build your layout.

In this project, we will use the ConstraintLayout. A ConstraintLayout is a layout that provides a flexible way for creating views.

Add a Title to the Layout
In the Palette panel, click Text and add it to the design. You might need to move it around so that it stays at the top of the layout.
This TextView control will display the form's description and purpose to the user. This control displays a string resource called @string/member_title, which must be defined within the /res/values/strings.xml string resource file.

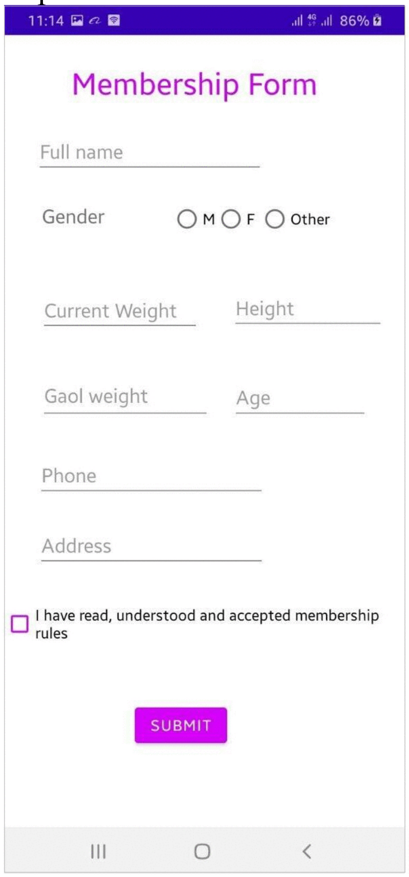
```
<resources>
   <string name="app_name">LEXO</string>
   <string name="title">Membership Form</string>
</resources>
 <TextView
      android:id="@+id/title"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginTop="28dp"
      android:text="@string/title"
      android:textColor="#D500F9"
      android:textSize="28sp"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintHorizontal_bias="0.436"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent" />
<TextView
      android:id="@+id/gender"
      android:layout_width="83dp"
      android:layout_height="32dp"
      android:layout_marginTop="172dp"
      android:text="Gender"
      android:textSize="18sp"
      app:layout_constraintEnd_toEndOf="parent"
```

```
        app:layout_constraintHorizontal_bias="0.118"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <RadioGroup
        android:id="@+id/radioGroup"
        android:layout_width="205dp"
        android:layout_height="32dp"
        android:layout_marginTop="172dp"
        android:orientation="horizontal"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.834"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">
        <RadioButton
            android:id="@+id/radioButton"
            android:layout_width="45dp"
            android:layout_height="33dp"
            android:text="M"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            android:onClick="radioButtonhandler"/>
        <RadioButton
            android:id="@+id/female"
            android:layout_width="45dp"
            android:layout_height="33dp"
            android:text="F"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.655"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            android:onClick="radioButtonhandler"/>
        <RadioButton
            android:id="@+id/other"
            android:layout_width="96dp"
            android:layout_height="33dp"
            android:text="Other"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.949"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            android:onClick="radioButtonhandler"/>
      </RadioGroup>
    <EditText
        android:id="@+id/current_weight"
        android:layout_width="164dp"
        android:layout_height="46dp"
        android:layout_marginTop="260dp"
        android:ems="10"
        android:hint="@string/currrent_weight"
        android:inputType="number"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.151"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <EditText
        android:id="@+id/height"
        android:layout_width="157dp"
        android:layout_height="44dp"
        android:layout_marginTop="260dp"
        android:ems="10"
        android:hint="@string/height"
        android:inputType="number"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.919"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
  <EditText
        android:id="@+id/goal_weight"
        android:layout_width="175dp"
        android:layout_height="52dp"
        android:layout_marginTop="344dp"
        android:ems="10"
        android:hint="@string/goal_weight"
        android:inputType="number"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.156"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <EditText
        android:id="@+id/age"
        android:layout_width="140dp"
        android:layout_height="48dp"
        android:layout_marginTop="348dp"
        android:ems="10"
        android:hint="Age"
        android:inputType="number"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.863"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
  <EditText
        android:id="@+id/Phone"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="428dp"
        android:ems="10"
        android:hint="@string/phone"
        android:inputType="phone"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.194"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
 <EditText
        android:id="@+id/address"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="500dp"
        android:ems="10"
        android:hint="@string/address"
        android:inputType="text"
        android:maxLines="2"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.194"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <CheckBox
        android:id="@+id/conditions"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="580dp"
        android:backgroundTint="#D500F9"
        android:buttonTint="#D500F9"
        android:text="@string/conditions"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.964"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="684dp"
        android:backgroundTint="#D500F9"
        android:onClick="submitbuttonHandler"
        android:text="SUBMIT"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.425"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <RadioGroup
        android:id="@+id/radioGroup"
        android:layout_width="205dp"
        android:layout_height="32dp"
        android:layout_marginTop="172dp"
        android:orientation="horizontal"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.834"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">
        <RadioButton
            android:id="@+id/radioButton"
            android:layout_width="45dp"
            android:layout_height="33dp"
            android:text="M"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            android:onClick="radioButtonhandler"/>
        <RadioButton
            android:id="@+id/female"
            android:layout_width="45dp"
            android:layout_height="33dp"
            android:text="F"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.655"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            android:onClick="radioButtonhandler"/>
        <RadioButton
            android:id="@+id/other"
            android:layout_width="96dp"
            android:layout_height="33dp"
            android:text="Other"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.949"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            android:onClick="radioButtonhandler" />
    </RadioGroup>
```

Open MainActivity.java and use the findViewById() method to return an instance of the full name on the onCreate() method.

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        EditText nameEditText = (EditText) findViewById(R.id.names);
        String fullName = nameEditText.getText().toString();
    }
public void radioButtonhandler(View view) {

    // Decide what happens when a user clicks on a button
    }

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        EditText nameEditText = (EditText) findViewById(R.id.names);
        String fullName = nameEditText.getText().toString();
        EditText currentWeightEditText = (EditText) findViewById(R.id.current_weight);
        String currentWeight = currentWeightEditText.getText().toString();
        EditText heightEditText = (EditText) findViewById(R.id.height);
        String Height = heightEditText.getText().toString();
        EditText goalWeightEditText = (EditText) findViewById(R.id.gol_weight);
        String GoalWeight = goalWeightEditText.getText().toString();
        EditText ageEditText = (EditText) findViewById(R.id.age);
        String age = ageEditText.getText().toString();
        EditText phoneEditText = (EditText) findViewById(R.id.names);
        String phone = phoneEditText.getText().toString();
        EditText addressEditText = (EditText) findViewById(R.id.names);
        String address = addressEditText.getText().toString();
    }

//initiate a check box
CheckBox conditionsCheckBox = (CheckBox)
findViewById(R.id.conditions_CheckBox);
//check current state of the check box
Boolean checkBoxState = conditionsCheckBox.isChecked();

public void submitbuttonHandler(View view) {
    //Decide what happens when the user clicks the submit button
    }
```

O/p


11:14
86%
Membership Form
Full name
Gender
M
F
Other
Current Weight
Height
Gaol weight
Age
Phone
Address
I have read, understood and accepted membership rules
SUBMIT