PRACTICAL NO 1

AIM:- a) Write a program using Kotlin to implement control structures and loops.

INTRODUCTION:-
You can use control structures and loops to control the flow of execution of your program. For example, you can use an if statement to check a condition and execute a block of code if the condition is true. You can use a while loop to execute a block of code repeatedly as long as a condition is true. And you can use a do-while loop to execute a block of code at least once, and then continue executing the block as long as a condition is true.
Control structures and loops are essential tools for any Kotlin programmer. By understanding how to use them, you can write more efficient and effective code.

1)If… Else Expression
Code:-
```
fun main(args:Array<String>){
    val age:Int = 10
    if (age>18){
          print(“Adult”)
    }
    else{
          print(“Minor”)
    }
}
```

2)when expression
Code:-
```
Fun Main(Args: Array<String>) {
    Val Day = 2
    Val Result = When (Day) {
          1 -> "Monday"
          2 -> "Tuesday"
          3 -> "Wednesday"
          4 -> "Thursday"
          5 -> "Friday"
          6 -> "Saturday"
          7 -> "Sunday"
          Else -> "Invalid Day."
     }
    Println(Result)
}
```

3)for loop
Code:-
```
fun main(args: Array<String>) {
    var fruits = arrayOf("Orange", "Apple", "Mango", "Banana")
    for (item in fruits) {
          println(item)
    }
}
```

4)while loop

Code:-

```
fun main(args: Array<String>) {
    var i = 5;
    while (i > 0) {
          println(i)
          i--
    }
}
```

5)do… while loop

Code:-

```
fun main(args: Array<String>) {
    var i = 5;
    do{
          println(i)
          i--
    }
    while(i > 0)
}
```

6)break statement

Code:-

```
fun main(args: Array<String>) {
    var i = 0;
    while (i++ < 100) {
          println(i)
          if( i == 3 ){
              break
          }
    }
}
```

7)continue statement

Code:-

```
fun main(args: Array<String>) {
    var i = 0;
    while (i++ < 6) {
          if( i == 3 ){
              continue
          }
          println(i)
    }
}
```

AIM:- b) Write a program to implement object-oriented concepts in Kotlin.

INTRODUCTION:-

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 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. 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.

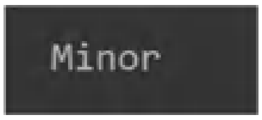
Code:-

```
class myclass{
    private var name:String = “UDEMY”
    fun printme( ){
          print(“The Best Online Education Sites - $name”)
    }
}
fun main (args:Array<String>){
    val obj=myclass( )
    obj.printme( )
}
```

PRACTICAL NO 1

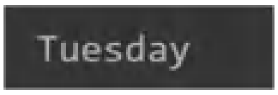
AIM:- a) Write a program using Kotlin to implement control structures and loops.

1)if… else statement
OUTPUT:-



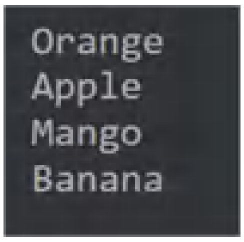
```
Minor
```

2)when statement
OUTPUT:-



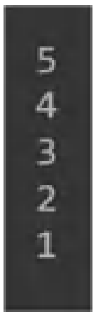
```
Tuesday
```

3)for loop
OUTPUT:-



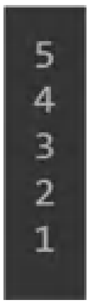
```
Orange
Apple
Mango
Banana
```

4)while loop
OUTPUT:-



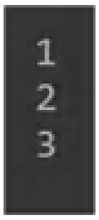
```
5
4
3
2
1
```

5)do… while loop
OUTPUT:-



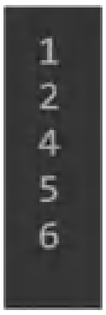
```
5
4
3
2
1
```

6)break statement
OUTPUT:-



```
1
2
3
```

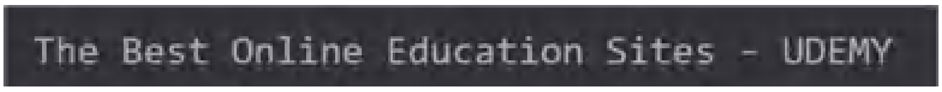
7)continue statement
OUTPUT:-



```
1
2
4
5
6
```

AIM:- b) Write a program to implement object-oriented concepts in Kotlin.

OUTPUT:-



```
The Best Online Education Sites - UDEMY
```

PRACTICAL NO 2

AIM:- a) Create an Android application to design screens using different layouts and UI including Button, Edittext, Textview, Radio Button etc.

b) Write an android application demonstrating response to event/user interaction for

a) Checkbox
b) Radio button
c) Button
d) Spinner

INTRODUCTION:-
In android UI or input controls are the interactive or View components that are used to design the user interface of an application. In android we have a wide variety of UI or input controls available, those are TextView, EditText, Buttons, Checkbox, Progressbar, Spinners, etc.
Generally, in android the user interface of an app is made with a collection of View and ViewGroup objects.
The View is a base class for all UI components in android and it is used to create interactive UI components such as TextView, EditText, Checkbox, Radio Button, etc. and it is responsible for event handling and drawing.
The ViewGroup is a subclass of View and it will act as a base class for layouts and layout parameters. The ViewGroup will provide invisible containers to hold other Views or ViewGroups and to define the layout properties.

CODE:-

←activity.main.xml→

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/name"
        android:layout_width="262dp"
        android:layout_height="51dp"
        android:ems="10"
        android:hint="Name"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.516"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.023" />

    <EditText
        android:id="@+id/mobileNumber"
        android:layout_width="262dp"
        android:layout_height="51dp"
        android:ems="10"
        android:hint="Mobile Num"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.516"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.125" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="119dp"
        android:layout_height="53dp"
        android:gravity="center"
        android:text="Gender"
        android:textColor="#06980C"
        android:textSize="20sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.247" />

    <RadioGroup
        android:id="@+id/radioButtonGroup"
        android:layout_width="340dp"
        android:layout_height="130dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.393">

        <RadioButton
            android:id="@+id/radioButton"
            android:layout_width="141dp"
            android:layout_height="59dp"
            android:text="Male"
            android:textSize="16sp" />

        <RadioButton
            android:id="@+id/radioButton2"
            android:layout_width="141dp"
            android:layout_height="59dp"
            android:text="Female"
            android:textSize="16sp" />
    </RadioGroup>
```

```
<Button
    android:id="@+id/button2"
    android:layout_width="155dp"
    android:layout_height="68dp"
    android:text="Submit"
    android:textSize="20sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.666" />

<CheckBox
    android:id="@+id/checkBox"
    android:layout_width="239dp"
    android:layout_height="56dp"
    android:text="Plzzz Check Term and Condition"
    android:textSize="16sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="@+id/name"
    app:layout_constraintVertical_bias="0.55" />

<TextView
    android:id="@+id/textViewR"
    android:layout_width="268dp"
    android:layout_height="46dp"
    android:gravity="center"
    android:textColor="#EC0E0E"
    android:textSize="20sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.496"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.773" />

<TextView
    android:id="@+id/textViewR2"
    android:layout_width="268dp"
    android:layout_height="46dp"
    android:gravity="center"
    android:textColor="#EA0909"
    android:textSize="20sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.943" />
    <TextView
        android:id="@+id/textViewR3"
        android:layout_width="268dp"
        android:layout_height="46dp"
        android:gravity="center"
        android:textColor="#E83B04"
        android:textSize="20sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.496"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.859" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

←MainActivity.kt→

```
package com.example.prac2

import android.annotation.SuppressLint
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.*

class MainActivity : AppCompatActivity() {
    @SuppressLint("MissingInflatedId")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val Name = findViewById<EditText>(R.id.name)
        val SubminB = findViewById<Button>(R.id.button2)
        val dispM =findViewById<TextView>(R.id.textViewR)
        val dispM2 =findViewById<TextView>(R.id.textViewR2)
        val dispM3 =findViewById<TextView>(R.id.textViewR3)
        val UserNumber = findViewById<EditText>(R.id.mobileNumber)
        val radioGroupB = findViewById<RadioGroup>(R.id.radioButtonGroup)

        radioGroupB.setOnCheckedChangeListener { group, checkedid ->
            val radio1=findViewById<RadioButton>(R.id.radioButton)
            val radio2=findViewById<RadioButton>(R.id.radioButton2)
            if (checkedid ==R.id.radioButton) {
                Toast.makeText(this, radio1.text.toString(),
                    Toast.LENGTH_SHORT).show()
                dispM3.text = radio1.text
            }
            if (checkedid ==R.id.radioButton2) {
                Toast.makeText(this, radio2.text.toString(),
                    Toast.LENGTH_SHORT).show()
                dispM3.text = radio2.text
            }
        }
        SubminB.setOnClickListener{
            dispM.setText("Wellcome:- "+Name.text+"!")
            dispM2.setText(UserNumber.text)
        }
    }
}
```

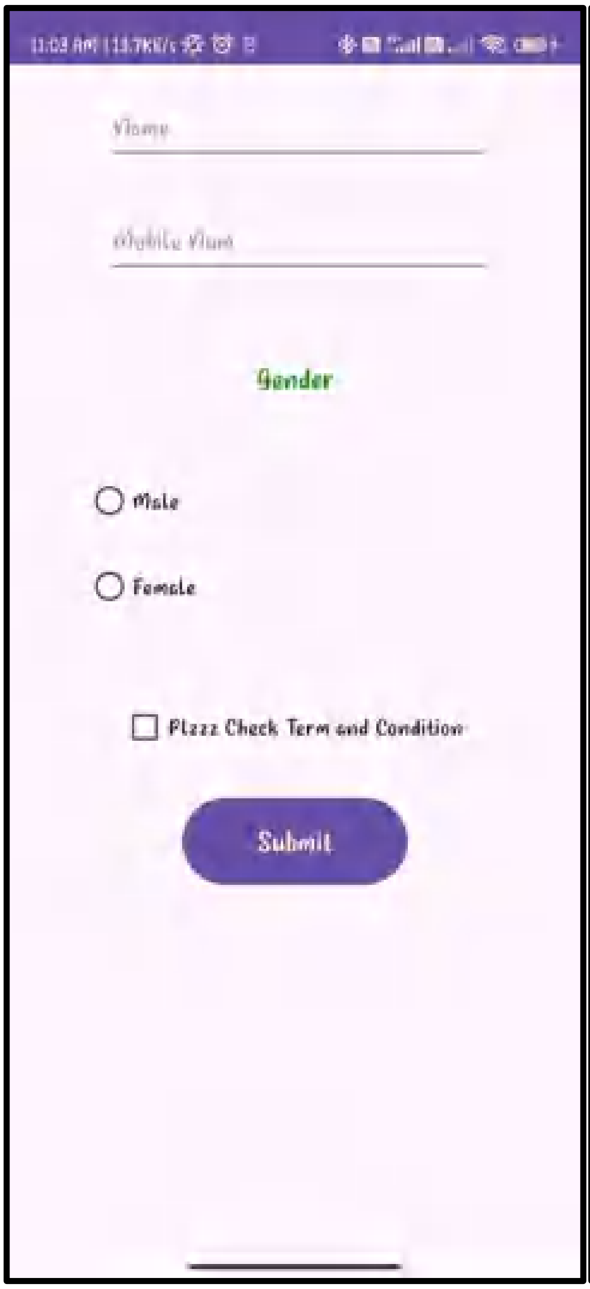
PRACTICAL NO 2

AIM:- a) Create an Android application to design screens using different layouts and UI including Button, Edittext, Textview, Radio Button etc.

b) Write an android application demonstrating response to event/user interaction for

a) Checkbox
b) Radio button
c) Button
d) Spinner

OUTPUT:-



Name

Mobile Num

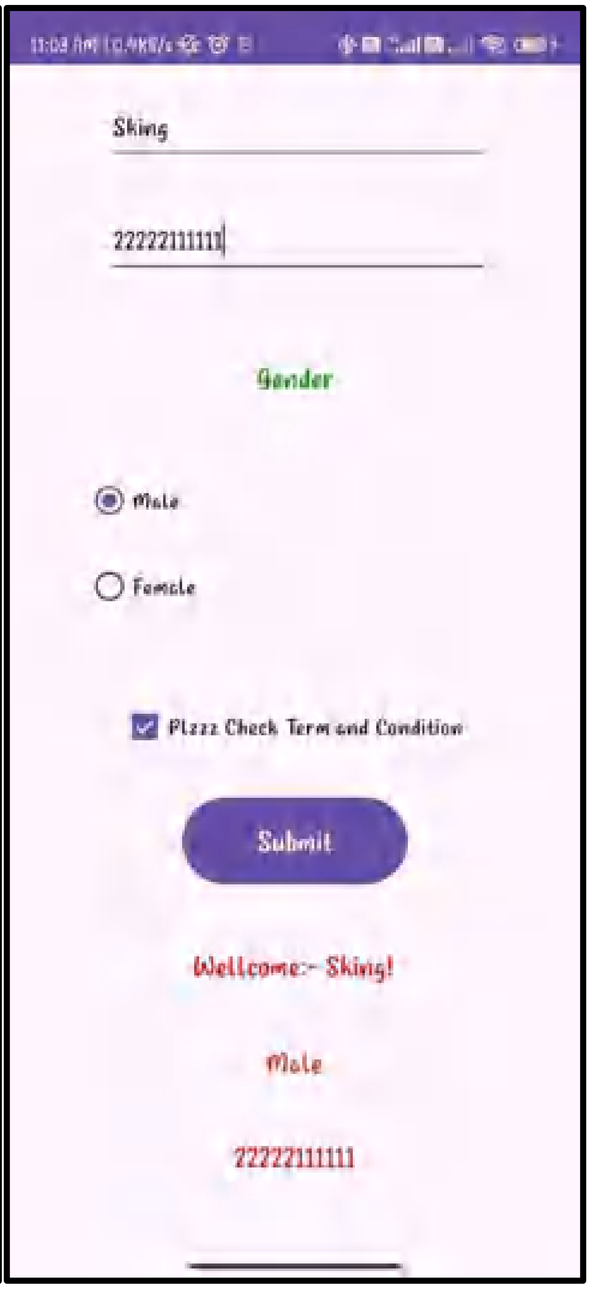
Gender

Male

Female

Plzzz Check Term and Condition

Submit



Sking

22222111111

Gender

Male

Female

Plzzz Check Term and Condition

Submit

Wellcome:- Sking!

Male

22222111111

PRACTICAL NO 3

AIM:- a) Create an application to create Image Flipper and Image Gallery. On click on the image display the information about the image.

INTRODUCTION:-

In Android, ImageView class is used to display an image file in application. Image file is easy to use but hard to master in Android, because of the various screen sizes in Android devices. An android is enriched with some of the best UI design widgets that allows us to build good looking and attractive UI based application. ImageView comes with different configuration options to support different scale types. Scale type options are used for scaling the bounds of an image to the bounds of the imageview. Some of them scaleTypes configuration properties are center, center_crop, fit_xy, fitStart etc. You can read our ScaleType tutorial to learn all details on it.

CODE:-

←activity_main.xml→

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.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=".MainActivity">
  <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:weightSum="3">
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:layout_weight="1"
      android:weightSum="2"
      tools:ignore="MissingConstraints">
      <ImageView
        android:id="@+id/html"
        android:layout_width="125dp"
        android:layout_height="125dp"
        android:layout_gravity="center_horizontal"
        android:src="@drawable/html"
        tools:ignore="MissingConstraints"
        tools:layout_editor_absoluteX="54dp"
        tools:layout_editor_absoluteY="137dp"/>
      <ImageView
        android:id="@+id/python"
        android:layout_width="125dp"
        android:layout_height="125dp"
        android:layout_gravity="center_horizontal"
        android:src="@drawable/python"
        tools:ignore="MissingConstraints"
        tools:layout_editor_absoluteX="225dp"
        tools:layout_editor_absoluteY="137dp"/>
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:weightSum="2"
    tools:ignore="MissingConstraints">
    <ImageView
        android:id="@+id/java"
        android:layout_width="125dp"
        android:layout_height="125dp"
        android:layout_gravity="center_horizontal"
        android:src="@drawable/java"
        tools:ignore="MissingConstraints"
        tools:layout_editor_absoluteX="65dp"
        tools:layout_editor_absoluteY="327dp"/>
    <ImageView
        android:id="@+id/kotlin"
        android:layout_width="125dp"
        android:layout_height="125dp"
        android:layout_gravity="center_horizontal"
        android:src="@drawable/kotlin"
        tools:ignore="MissingConstraints"
        tools:layout_editor_absoluteX="225dp"
        tools:layout_editor_absoluteY="327dp"/>
</LinearLayout>
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:weightSum="2"
    tools:ignore="MissingConstraints">
    <ImageView
        android:id="@+id/ruby"
        android:layout_width="125dp"
        android:layout_height="125dp"
        android:layout_gravity="center_horizontal"
        android:src="@drawable/ruby"
        tools:ignore="MissingConstraints"
        tools:layout_editor_absoluteX="235dp"
        tools:layout_editor_absoluteY="503dp"/>
    <ImageView
        android:id="@+id/go"
        android:layout_width="125dp"
        android:layout_height="125dp"
        android:layout_gravity="center_horizontal"
        android:src="@drawable/go"
        tools:ignore="MissingConstraints"
```

```
            tools:layout_editor_absoluteX="72dp"
            tools:layout_editor_absoluteY="503dp"/>
        </LinearLayout>
    </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

←MainActivity.kt→

```
package com.example.prac3a

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.ImageView
import android.widget.Toast

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

        val html =findViewById<ImageView>(R.id.html)
        val python =findViewById<ImageView>(R.id.python)
        val java =findViewById<ImageView>(R.id.java)
        val kotlin =findViewById<ImageView>(R.id.kotlin)
        val ruby =findViewById<ImageView>(R.id.ruby)
        val go =findViewById<ImageView>(R.id.go)

        html.setOnClickListener{
            Toast.makeText(applicationContext,"YouClickedHTMLLogo",Toast.LENGTH_LONG).show()
        }
        python.setOnClickListener{
            Toast.makeText(applicationContext,"YouClickedPythonLogo",Toast.LENGTH_LONG).show()
        }
        java.setOnClickListener{
            Toast.makeText(applicationContext,"YouClickedJavaLogo",Toast.LENGTH_LONG).show()
        }
        kotlin.setOnClickListener{
            Toast.makeText(applicationContext,"YouClickedKotlinLogo",Toast.LENGTH_LONG).show()
        }
        ruby.setOnClickListener{
            Toast.makeText(applicationContext,"YouClickedRubyLogo",Toast.LENGTH_LONG).show()
        }
        go.setOnClickListener{
            Toast.makeText(applicationContext,"YouClickedGOLogo",Toast.LENGTH_LONG).show()
        }
    }
}
```

AIM:- b) Create an application to use Gridview for shopping cart application.

INTRODUCTION:-

A GridView is a type of AdapterView that displays items in a two-dimensional scrolling grid. Items are inserted into this grid layout from a database or from an array. The adapter is used for displaying this data, setAdapter() method is used to join the adapter with GridView. The main function of the adapter in GridView is to fetch data from a database or array and insert each piece of data in an appropriate item that will be displayed in GridView. This is what the GridView structure looks like. We are going to implement this project using both Java and Kotlin Programming Language for Android.

XML Attributes of GridView

- android:numColumns: This attribute of GridView will be used to decide the number of columns that are to be displayed in Grid.
- android:horizontalSpacing: This attribute is used to define the spacing between two columns of GridView.
- android:verticalSpacing: This attribute is used to specify the spacing between two rows of GridView.

CODE:-

←activity_main.xml→

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

    <GridView
        android:id="@+id/my_grid_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:numColumns="auto_fit"
        android:columnWidth="150dp"
        android:horizontalSpacing="15dp"
        android:verticalSpacing="15dp"/>
</RelativeLayout>
```

←grid_item_list.xml→

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <ImageView
        android:id="@+id/icons"
        android:layout_width="90dp"
        android:layout_height="90dp"
        android:layout_marginLeft="30dp"
        android:layout_marginTop="6dp"
        android:src="@drawable/tel"/>
    <TextView
        android:id="@+id/name_text_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="30dp"
        android:layout_marginTop="10dp"
        android:gravity="center"
        android:text="shoppingcard"
        android:textStyle="bold"/>
</LinearLayout>
```

←MainActivity.kt→

```
package com.example.prac3b

import android.annotation.SuppressLint
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.AdapterView
import android.widget.GridView
import android.widget.Toast

class MainActivity:AppCompatActivity(),AdapterView.OnItemClickListener {
    private var gridView:GridView?=null
    private var arrayList:ArrayList<LanguageItem> ?=null
    private var languageAdapter:LanguageAdapter?=null
    override fun onCreate(savedInstanceState:Bundle?){
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        gridView=findViewById(R.id.my_grid_view)
        arrayList=ArrayList()
        arrayList=setDataList()
        languageAdapter=LanguageAdapter(applicationContext,arrayList!!)
        gridView?.adapter=languageAdapter
        gridView?.onItemClickListener=this
    }
    private fun setDataList():ArrayList<LanguageItem>{
        var arrayList:ArrayList<LanguageItem> = ArrayList()
        arrayList.add(LanguageItem(R.drawable.apple,"Apple:Rs:100"))
        arrayList.add(LanguageItem(R.drawable.butter,"Butter:Rs:250"))
        arrayList.add(LanguageItem(R.drawable.graphs,"Graphs:Rs:75"))
        arrayList.add(LanguageItem(R.drawable.horliks,"horlicks:Rs:505"))
        arrayList.add(LanguageItem(R.drawable.lays,"Lays:Rs:15"))
        arrayList.add(LanguageItem(R.drawable.maggi,"Maggi:Rs:08"))
        arrayList.add(LanguageItem(R.drawable.mango,"Mango:Rs:180"))
        arrayList.add(LanguageItem(R.drawable.orange,"Orange:Rs:60"))
        arrayList.add(LanguageItem(R.drawable.oreo,"Oreo:Rs:15"))
        arrayList.add(LanguageItem(R.drawable.potato,"Potato:Rs:55"))
        arrayList.add(LanguageItem(R.drawable.tomato,"Tomato:Rs:35"))
        arrayList.add(LanguageItem(R.drawable.souce,"Souces:Rs:45"))
        arrayList.add(LanguageItem(R.drawable.soap,"Soap:Rs:30"))
        arrayList.add(LanguageItem(R.drawable.tel,"Oil:Rs:50"))
        arrayList.add(LanguageItem(R.drawable.vegitable,"Vegiteble:Rs:80"))
        arrayList.add(LanguageItem(R.drawable.watermelon,"Watermelon:Rs:200"))
        arrayList.add(LanguageItem(R.drawable.strawberry,"Strawberry:Rs:150"))
        return arrayList
    }
    override fun onItemClick(p0:AdapterView<*>?,p1:View?,p2:Int,p3:Long){
        var languageItem:LanguageItem=arrayList!!.get(p2)
        Toast.makeText(applicationContext,"Addsuccessfully",Toast.LENGTH_SHORT).show()
        Toast.makeText(applicationContext,languageItem.name,Toast.LENGTH_LONG).show()
    }
}
```

←LanguageAdapter.kt→

```
package com.example.prac3b

import android.content.Context
import android.view.View
import android.view.ViewGroup
import android.widget.BaseAdapter
import android.widget.ImageView
import android.widget.TextView

class LanguageAdapter(var context:Context,var arrayList:ArrayList<LanguageItem>):BaseAdapter(){
    override fun getCount():Int{
        return arrayList.size
    }
    override fun getItem(p0:Int):Any{
        return arrayList.get(p0)
    }
    override fun getItemId(p0:Int):Long{
        return p0.toLong()
    }
    override fun getView(p0:Int,p1:View?,p2:ViewGroup?):View{
        var view:View=View.inflate(context,R.layout.grid_item_list,null)
        var icons:ImageView=view.findViewById(R.id.icons)
        var names:TextView=view.findViewById(R.id.name_text_view)
        var languageItem:LanguageItem =arrayList.get(p0)
        icons.setImageResource(languageItem.icons!!)
        names.text=languageItem.name
        return view
    }
}
```

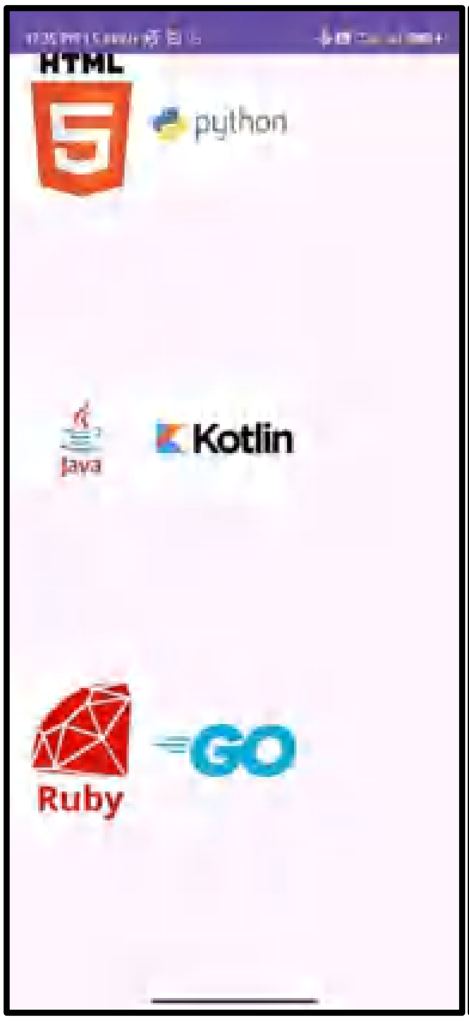
←LanguageItem.kt→

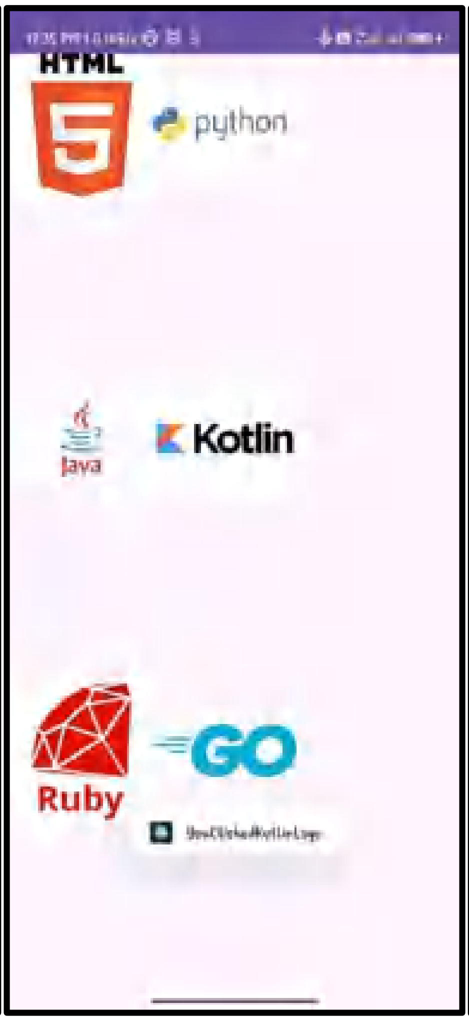
```
package com.example.prac3b

class LanguageItem {

    var icons:Int?=0
    var name:String?=null

    constructor(icons:Int?,name:String?){
        this.icons=icons
        this.name=name
    }
}
```

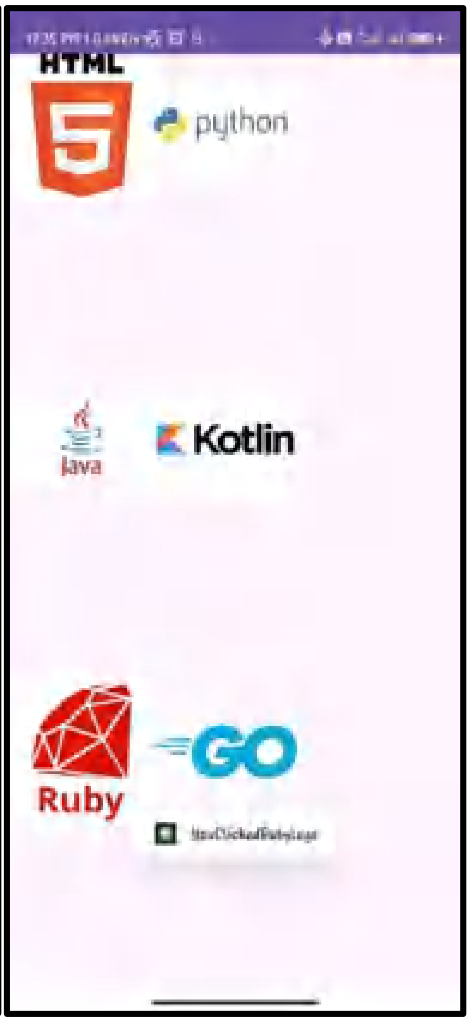
PRACTICAL NO 3

AIM:- a) Create an application to create Image Flipper and Image Gallery. On click on the image display the information about the image.

OUTPUT:-

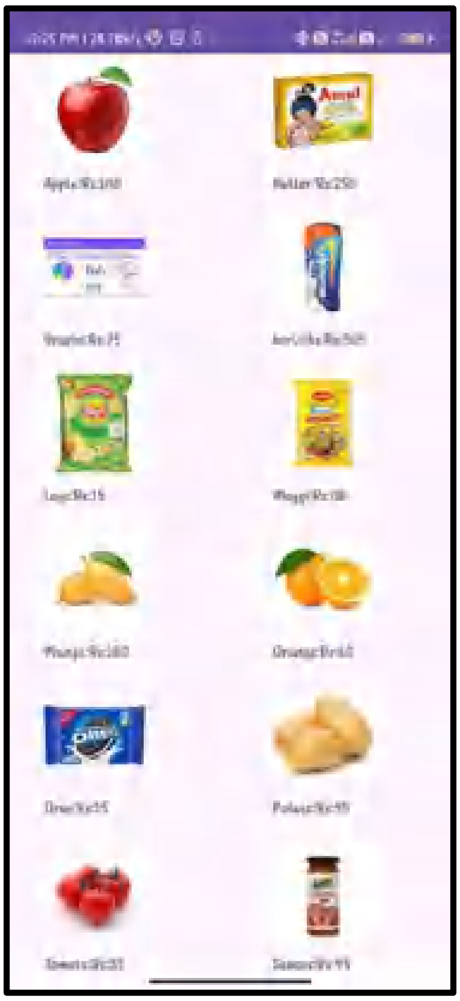


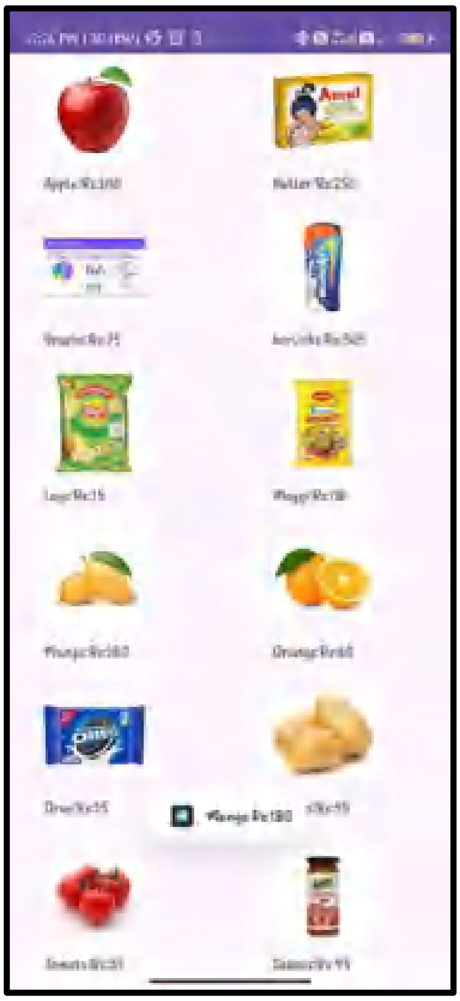


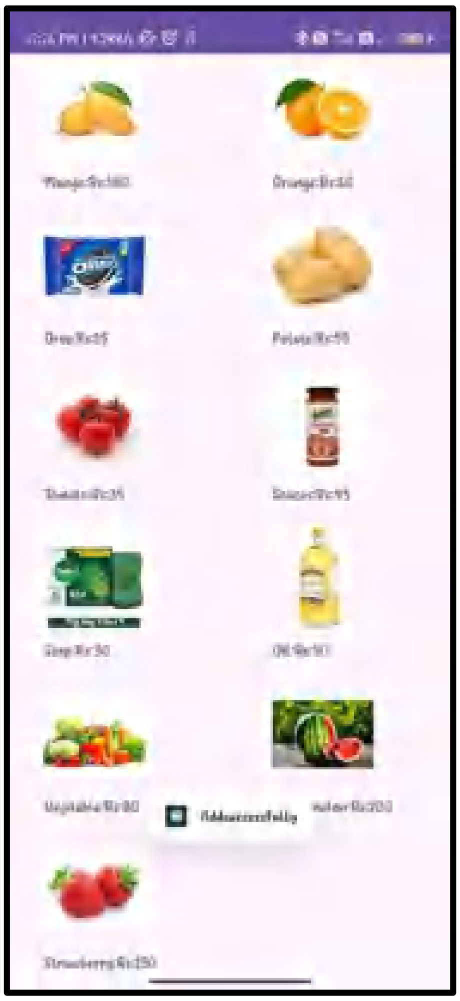


AIM:- b) Create an application to use Gridview for shopping cart application.

OUTPUT:-







PRACTICAL NO 4

AIM:- a) Create an Android application to demonstrate implicit and explicit intents

INTRODUCTION:-

The intent is a messaging object which passes between components like services, content providers, activities, etc. Normally startActivity() method is used for invoking any activity.

There are two types of intents in android
Implicit Intent
Explicit Intentt Intent
Using implicit Intent, components can’t be specified. An action to be performed is declared by implicit intent. Then android operating system will filter out components that will respond to the action.it Intent
Using explicit intent any other component can be specified. In other words, the targeted component is specified by explicit intent. So only the specified target component will be invoked.

CODE:-

←activity_main.xml→

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.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=".MainActivity">
  <TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:layout_marginTop="8dp"
    android:text="Your First Activity"
    android:textSize="18sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintHorizontal_bias="0.501"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.172" />
  <Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:text="click to invoke intent"
```

```
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
        app:layout_constraintVertical_bias="0.77" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

←MainActivity.kt→

```
package com.example.prac4

import android.content.Intent
import android.net.Uri
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Button

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

        val btn = findViewById<Button>(R.id.button)

        btn.setOnClickListener() {
            intent = Intent(Intent.ACTION_VIEW)
            intent.setData(Uri.parse("https://www.javatpoint.com/"))
            startActivity(intent)
            /* intent= Intent(Intent.ACTION_VIEW,
            Uri.parse("https://www.javatpoint.com/"))
            startActivity(intent)*/
        }
    }
}
```

AIM:- b) Create an application to demonstrate shared preferences

INTRODUCTION:-

One of the most Interesting Data Storage options Android provides its users is Shared Preferences. Shared Preferences is the way in which one can store and retrieve small amounts of primitive data as key/value pairs to a file on the device storage such as String, int, float, Boolean that make up your preferences in an XML file inside the app on the device storage. Shared Preferences can be thought of as a dictionary or a key/value pair. For example, you might have a key being “username” and for the value, you might store the user’s username. And then you could retrieve that by its key (here username).

CODE:-

←activity_main.xml→
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.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=".MainActivity">
   <TableLayout
      android:layout_width="368dp"
      android:layout_height="495dp"
      android:layout_marginBottom="8dp"
      android:layout_marginEnd="8dp"
      android:layout_marginStart="8dp"
      android:layout_marginTop="8dp"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent">
      <TableRow>
         <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_column="0"
            android:layout_marginLeft="10sp"
            android:layout_marginStart="10sp"
            android:text="Enter Id"
            android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium" />
         <EditText
            android:id="@+id/editId"
            android:layout_width="201dp"
            android:layout_height="wrap_content"
            android:layout_column="1"
            android:layout_marginLeft="50sp"
            android:layout_marginStart="50sp"
            android:hint="id"
            android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium" />
      </TableRow>
      <TableRow>
         <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_column="0"
            android:layout_marginLeft="10sp"
            android:layout_marginStart="10sp"
```

```xml
            android:text="Enter Name"
            android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium" />
        <EditText
            android:id="@+id/editName"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_column="1"
            android:layout_marginLeft="50sp"
            android:layout_marginStart="50sp"
            android:hint="name"
            android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium" />
    </TableRow>
    <TableRow android:layout_marginTop="60dp">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_column="0"
            android:layout_marginLeft="10sp"
            android:layout_marginStart="10sp"
            android:text="Your Id"
            android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium" />
        <TextView
            android:id="@+id/textViewShowId"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_column="1"
            android:layout_marginLeft="50sp"
            android:layout_marginStart="50sp"
            android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium" />
    </TableRow>
    <TableRow android:layout_marginTop="20dp">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_column="0"
            android:layout_marginLeft="10sp"
            android:layout_marginStart="10sp"
            android:text="Your Name"
            android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium" />
        <TextView
            android:id="@+id/textViewShowName"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_column="1"
            android:layout_marginLeft="50sp"
            android:layout_marginStart="50sp"
            android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium" />
    </TableRow>
```

```
  </TableLayout>
  <LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="16dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:orientation="horizontal"
    android:gravity="center"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent">
    <Button
      android:id="@+id/save"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Save" />
    <Button
      android:id="@+id/view"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="View" />
    <Button
      android:id="@+id/clear"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Clear" />
  </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

←MainActivity.kt→

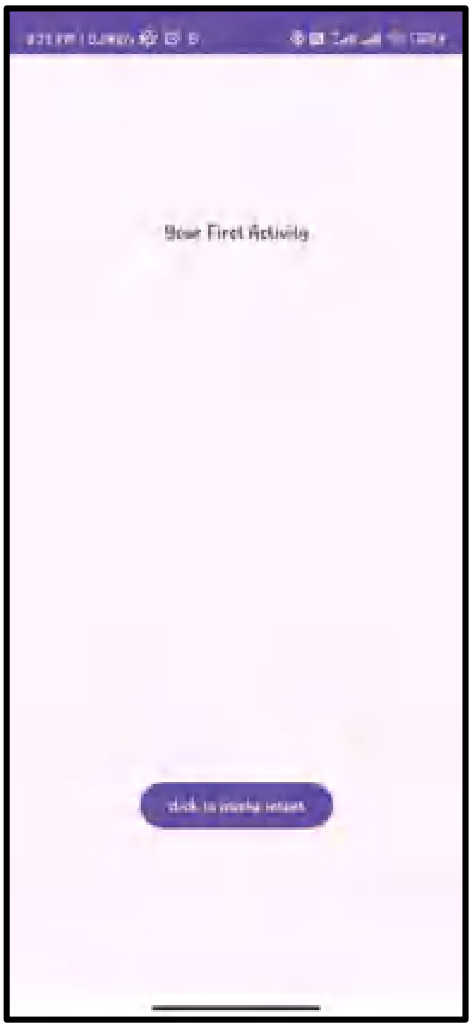
```
package com.example.prac4b

import android.content.Context
import android.content.SharedPreferences
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.Button
import android.widget.EditText
import android.widget.TextView

class MainActivity : AppCompatActivity() {
  private val sharedPrefFile = "kotlinsharedpreference"
  override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
```

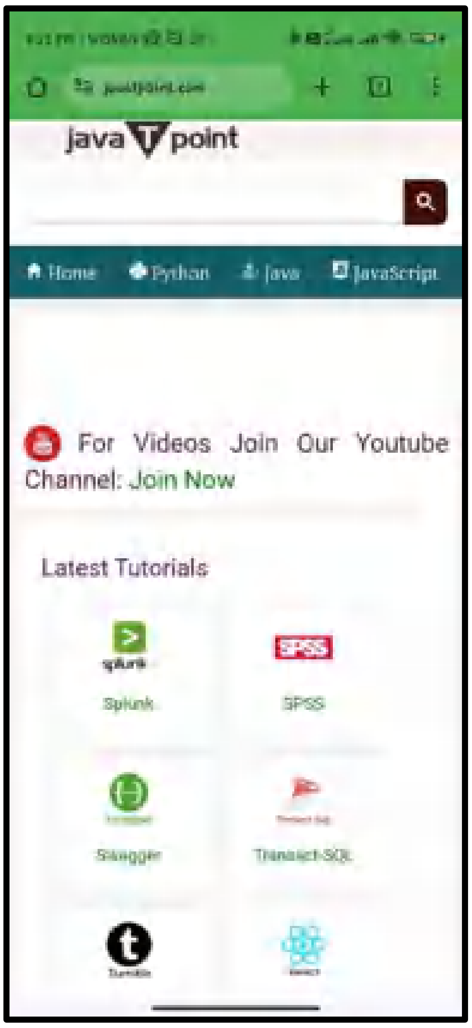
```
        val inputId = findViewById<EditText>(R.id.editId)
        val inputName = findViewById<EditText>(R.id.editName)
        val outputId = findViewById<TextView>(R.id.textViewShowId)
        val outputName = findViewById<TextView>(R.id.textViewShowName)
        val btnSave = findViewById<Button>(R.id.save)
        val btnView = findViewById<Button>(R.id.view)
        val btnClear = findViewById<Button>(R.id.clear)
        val sharedPreferences: SharedPreferences =
            this.getSharedPreferences(sharedPrefFile,
                Context.MODE_PRIVATE)
        btnSave.setOnClickListener(View.OnClickListener {
            val id:Int = Integer.parseInt(inputId.text.toString())
            val name:String = inputName.text.toString()
            val editor:SharedPreferences.Editor = sharedPreferences.edit()
            editor.putInt("id_key",id)
            editor.putString("name_key",name)
            editor.apply()
            editor.commit()
        })
        btnView.setOnClickListener {
            val sharedIdValue = sharedPreferences.getInt("id_key",0)
            val sharedNameValue =
                sharedPreferences.getString("name_key","defaultname")
            if (sharedNameValue != null) {
                if(sharedIdValue.equals(0) && sharedNameValue.equals("defaultname")){
                    outputName.setText("default name: ${sharedNameValue}").toString()
                    outputId.setText("default id: ${sharedIdValue.toString()}")
                }else{
                    outputName.setText(sharedNameValue).toString()
                    outputId.setText(sharedIdValue.toString())
                }
            }
        }
        btnClear.setOnClickListener(View.OnClickListener {
            val editor = sharedPreferences.edit()
            editor.clear()
            editor.apply()
            outputName.setText("").toString()
            outputId.setText("".toString())
        })
    }
}
```

PRACTICAL NO 4

AIM:- a) Create an Android application to demonstrate implicit and explicit intents

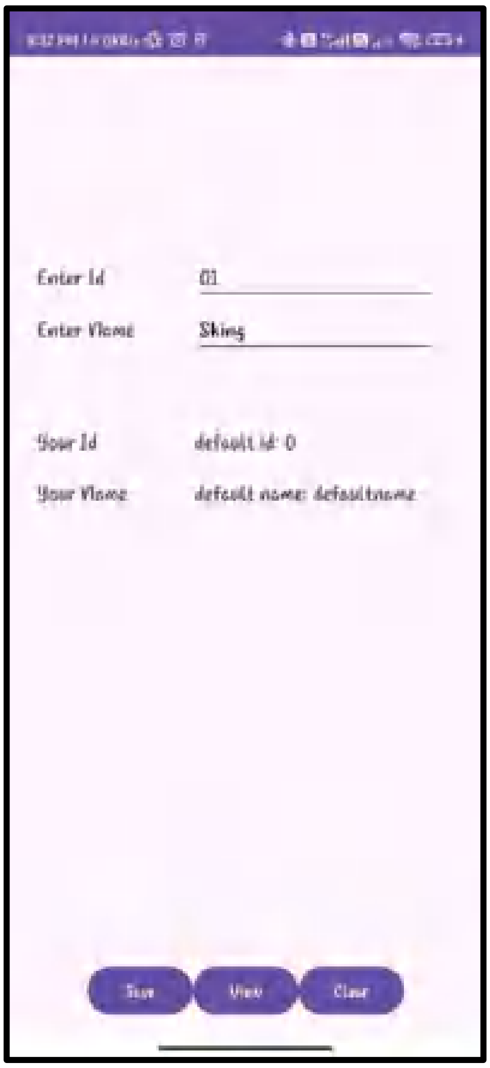
OUTPUT:-



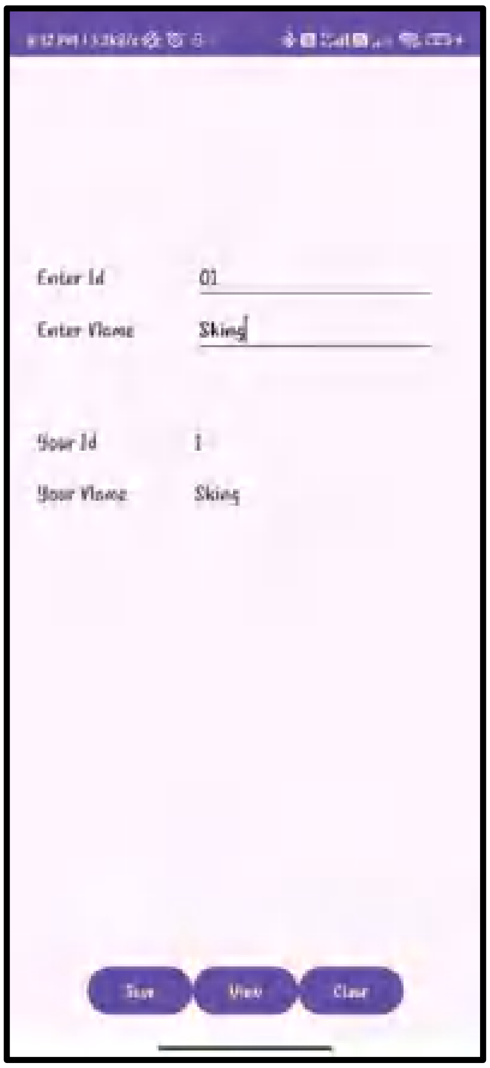


AIM:- b) Create an application to demonstrate shared preferences

OUTPUT:-



```
Enter Id        01
Enter Name      Sking

Your Id         default id: 0
Your Name       default name: defaultname
```



```
Enter Id        01
Enter Name      Sking

Your Id         1
Your Name       Sking
```

PRACTICAL NO 5

AIM:- a) Create an Android application to demonstrate the use of Broadcast listeners.
b) Create an Android application to create and use services.

INTRODUCTION:-
Broadcast in android is the system-wide events that can occur when the device starts, when a message is received on the device or when incoming calls are received, or when a device goes to airplane mode, etc. Broadcast Receivers are used to respond to these system-wide events. Broadcast Receivers allow us to register for the system and application events, and when that event happens, then the register receivers get notified. There are mainly two types of Broadcast Receivers:
Static Broadcast Receivers: These types of Receivers are declared in the manifest file and works even if the app is closed.
Dynamic Broadcast Receivers: These types of receivers work only if the app is active or minimized.

CODE:-

←activity_main.xml→
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.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=".MainActivity">
  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="BroadCast Activity"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

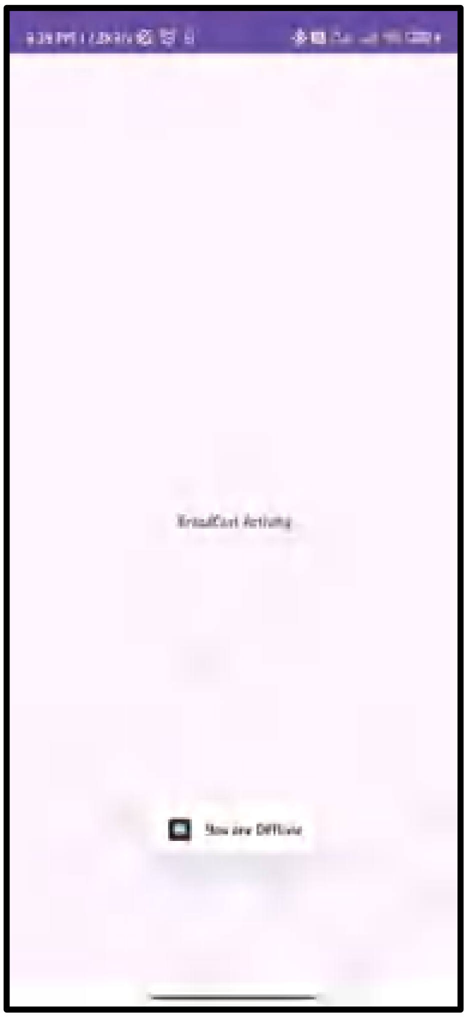
←MainActivity.kt→
```
package com.example.prac5

import android.content.Context
import android.net.ConnectivityManager
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Toast

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

```
        var c = applicationContext.getSystemService(Context.CONNECTIVITY_SERVICE)as ConnectivityManager
        val NetworkInfo = c.activeNetworkInfo

        if(NetworkInfo!=null&&NetworkInfo.isConnected)
        {
            if(NetworkInfo.type==ConnectivityManager.TYPE_MOBILE){
                Toast.makeText(applicationContext, "Connected to Mobile",
                    Toast.LENGTH_LONG).show()
            }
            if(NetworkInfo.type==ConnectivityManager.TYPE_WIFI){
                Toast.makeText(applicationContext, "Connected to WiFi",
                    Toast.LENGTH_LONG).show()
            }
            else{
                Toast.makeText(applicationContext, "You are Offline",
                    Toast.LENGTH_SHORT).show()
            }
        }
    }
}
```

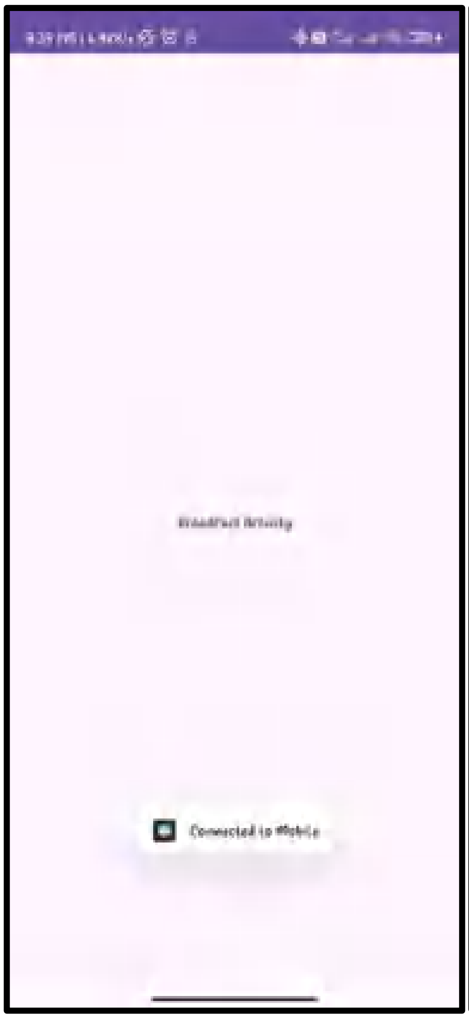
PRACTICAL NO 5

AIM:- a) Create an Android application to demonstrate the use of Broadcast listeners.
b) Create an Android application to create and use services.

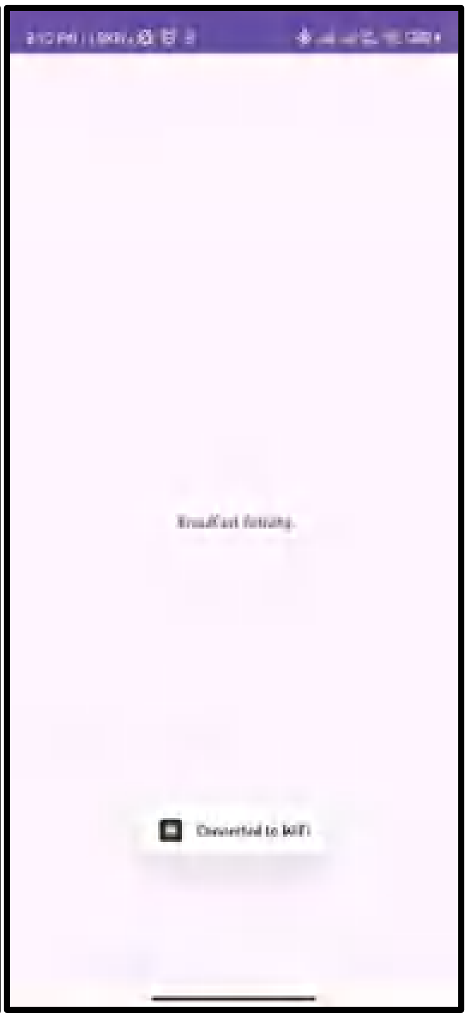
OUTPUT:-


BroadCast Activity

You are Offline


BroadCast Activity

Connected to Mobile


BroadCast Activity

Connected to WiFi

PRACTICAL NO 6

AIM:- a) Create an Android application to demonstrate XML based animation
b) Create an Android application to display canvas and allow the user to draw on it.

INTRODUCTION:-
Animation is the process of adding a motion effect to any view, image, or text. With the help of an animation, you can add motion or can change the shape of a specific view. Animation in Android is generally used to give your UI a rich look and feel.
Animations can add visual cues that notify users about what's going on in your app. They are especially useful when the UI changes state, such as when new content loads or new actions become available. Animations also add a polished look to your app, which gives it a higher quality look and feel.
Android includes different animation APIs depending on what type of animation you want. This documentation provides an overview of the different ways you can add motion to your UI.

CODE:-

←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">
    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_above="@+id/linearLayout"
        android:gravity="center"
        android:text="SKING"
        android:textSize="32sp"
        android:textStyle="bold" />
    <LinearLayout
        android:id="@+id/linearLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:weightSum="2">
            <Button
                android:id="@+id/fade_in"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="Fade In"
                android:textAllCaps="false" />
            <Button
                android:id="@+id/fade_out"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="Fade Out"
                android:textAllCaps="false" />
        </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:weightSum="2">
            <Button
                android:id="@+id/zoom_in"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="Zoom In"
                android:textAllCaps="false" />
            <Button
                android:id="@+id/zoom_out"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="Zoom Out"
                android:textAllCaps="false" />
        </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:weightSum="2">
            <Button
                android:id="@+id/slide_down"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="Slide Down"
                android:textAllCaps="false" />
            <Button
                android:id="@+id/slide_up"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="Slide Up"
            android:textAllCaps="false" />
        </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:weightSum="2">
            <Button
                android:id="@+id/bounce"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="Bounce"
                android:textAllCaps="false" />
            <Button
                android:id="@+id/rotate"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="Rotate"
                android:textAllCaps="false" />
        </LinearLayout>
    </LinearLayout>
</RelativeLayout>
```

←bounce.xml→

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:interpolator="@android:anim/linear_interpolator"
    android:fillAfter="true">
    <translate
        android:fromYDelta="100%"
        android:toYDelta="-20%"
        android:duration="300" />
    <translate
        android:startOffset="500"
        android:fromYDelta="-20%"
        android:toYDelta="10%"
        android:duration="150" />
    <translate
        android:startOffset="1000"
        android:fromYDelta="10%"
        android:toYDelta="0"
        android:duration="100" />
</set>
```

←rotate.xml→
```
<?xml version="1.0" encoding="utf-8"?>
<rotate xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="1000"
    android:fromDegrees="0"
    android:interpolator="@android:anim/linear_interpolator"
    android:pivotX="50%"
    android:pivotY="50%"
    android:startOffset="0"
    android:toDegrees="360" />
```

←slide_in.xml→
```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate
        android:duration="1000"
        android:fromYDelta="-100%"
        android:toYDelta="0" />
</set>
```

←slide_out.xml→
```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate
        android:duration="1000"
        android:fromYDelta="0"
        android:toYDelta="-100%" />
</set>
```

←fade_in.xml→
```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:interpolator="@android:anim/linear_interpolator">
    <alpha
        android:duration="1000"
        android:fromAlpha="0.1"
        android:toAlpha="1.0" />
</set>
```

←fade_out.xml→
```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:interpolator="@android:anim/linear_interpolator">
    <alpha
        android:duration="1000"
        android:fromAlpha="1.0"
        android:toAlpha="0.1" />
</set>
```

🡐zoom_in.xml🡒
```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
   android:fillAfter="true">
   <scale xmlns:android="http://schemas.android.com/apk/res/android"
      android:duration="1000"
      android:fromXScale="1"
      android:fromYScale="1"
      android:pivotX="50%"
      android:pivotY="50%"
      android:toXScale="1.5"
      android:toYScale="1.5">
   </scale>
</set>
```

🡐zoom_out.xml🡒
```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
   android:fillAfter="true" >
   <scale
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:duration="1000"
      android:fromXScale="1.0"
      android:fromYScale="1.0"
      android:pivotX="50%"
      android:pivotY="50%"
      android:toXScale="0.5"
      android:toYScale="0.5" >
   </scale>
</set>
```

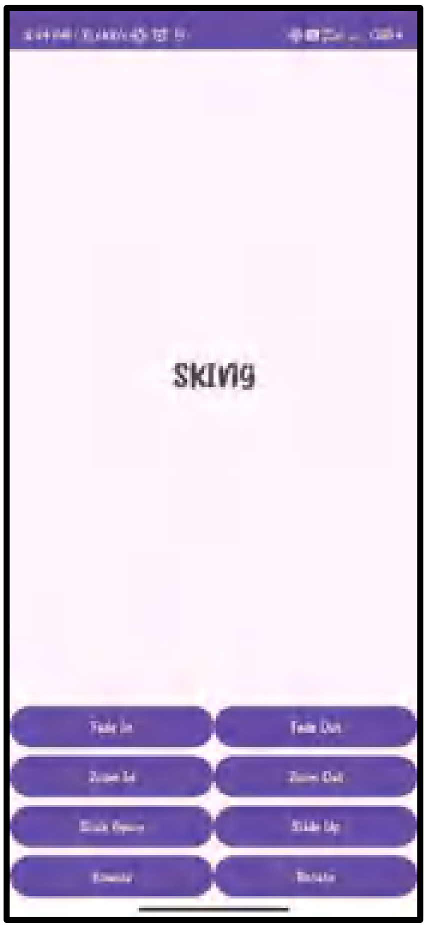
🡐MainActivity.kt🡒
```
package com.example.prac6

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.os.Handler
import android.view.View
import android.view.animation.AnimationUtils
import android.widget.TextView
import java.util.Collections.rotate as rotate1
class MainActivity : AppCompatActivity() {
   override fun onCreate(savedInstanceState: Bundle?) {
      super.onCreate(savedInstanceState)
      setContentView(R.layout.activity_main)
      val fade_in = findViewById<View>(R.id.fade_in)
      val fade_out = findViewById<View>(R.id.fade_out)
      val zoom_in = findViewById<View>(R.id.zoom_in)
      val zoom_out = findViewById<View>(R.id.zoom_out)
      val slide_down = findViewById<View>(R.id.slide_down)
      val slide_up = findViewById<View>(R.id.slide_up)
      val bounce = findViewById<View>(R.id.bounce)
      val rotate = findViewById<View>(R.id.rotate)
      fade_in.setOnClickListener {
         val textView = findViewById<TextView>(R.id.textView)
         textView.visibility = View.VISIBLE
         val animationFadeIn = AnimationUtils.loadAnimation(this, R.anim.fade_in)
         textView.startAnimation(animationFadeIn)
      }
      fade_out.setOnClickListener {
         val animationFadeOut = AnimationUtils.loadAnimation(this, R.anim.fade_out)
         val textView = findViewById<TextView>(R.id.textView)
         textView.startAnimation(animationFadeOut)
         Handler().postDelayed({
            textView.visibility = View.GONE
         }, 1000)
      }
      zoom_in.setOnClickListener {
         val animationZoomIn = AnimationUtils.loadAnimation(this, R.anim.zoom_in)
         val textView = findViewById<TextView>(R.id.textView)
         textView.startAnimation(animationZoomIn)
      }
      zoom_out.setOnClickListener {
         val animationZoomOut = AnimationUtils.loadAnimation(this,
            R.anim.zoom_out)
         val textView = findViewById<TextView>(R.id.textView)
         textView.startAnimation(animationZoomOut)
      }
      slide_down.setOnClickListener {
         val animationSlideDown = AnimationUtils.loadAnimation(this,
            R.anim.slide_in)
         val textView = findViewById<TextView>(R.id.textView)
         textView.startAnimation(animationSlideDown)
      }
      slide_up.setOnClickListener {
         val animationSlideUp = AnimationUtils.loadAnimation(this, R.anim.slide_out)
         val textView = findViewById<TextView>(R.id.textView)
         textView.startAnimation(animationSlideUp)
      }
      bounce.setOnClickListener {
         val animationBounce = AnimationUtils.loadAnimation(this, R.anim.bounce)
         val textView = findViewById<TextView>(R.id.textView)
         textView.startAnimation(animationBounce)
      }
      rotate.setOnClickListener {
         val animationRotate = AnimationUtils.loadAnimation(this, R.anim.rotate)
```

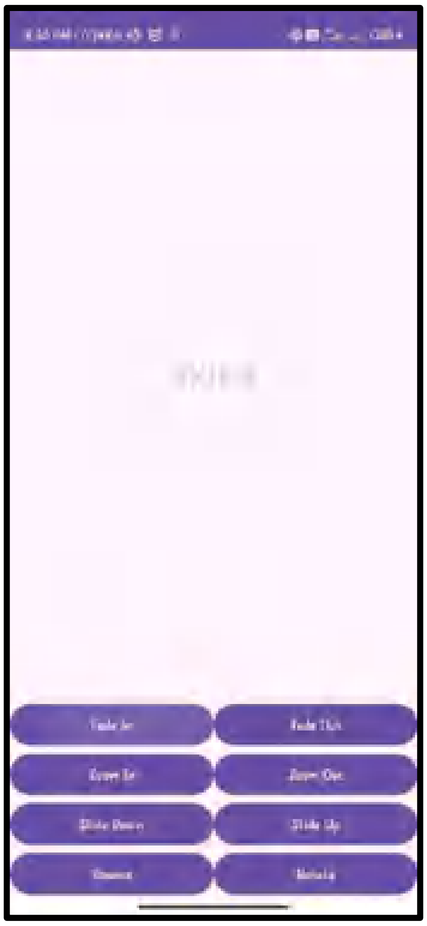
```
            val textView = findViewById<TextView>(R.id.textView)
            textView.startAnimation(animationRotate)
        }
    }
}
```

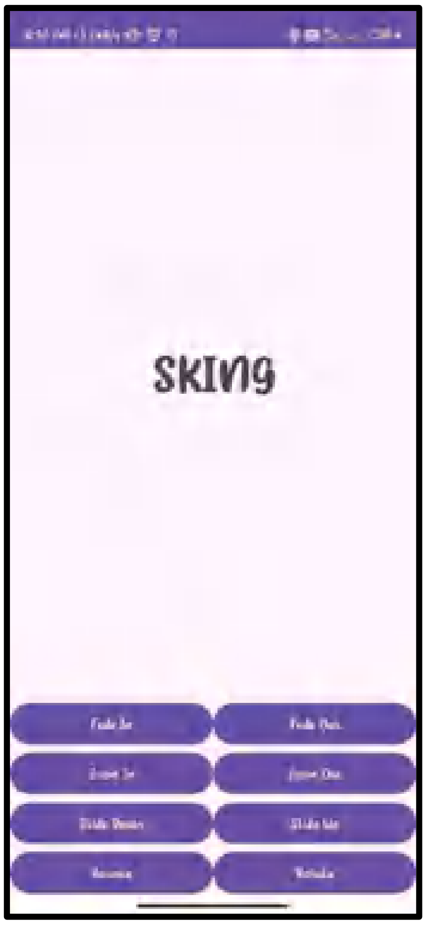
PRACTICAL NO 6

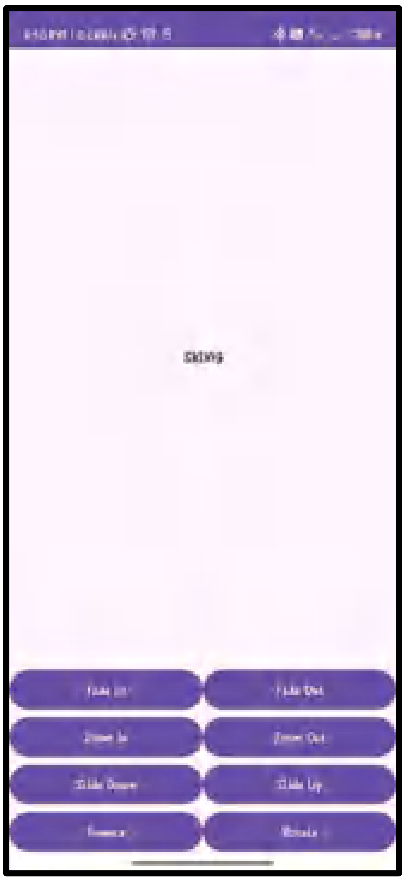
AIM:- a) Create an Android application to demonstrate XML based animation
b) Create an Android application to display canvas and allow the user to draw on it.

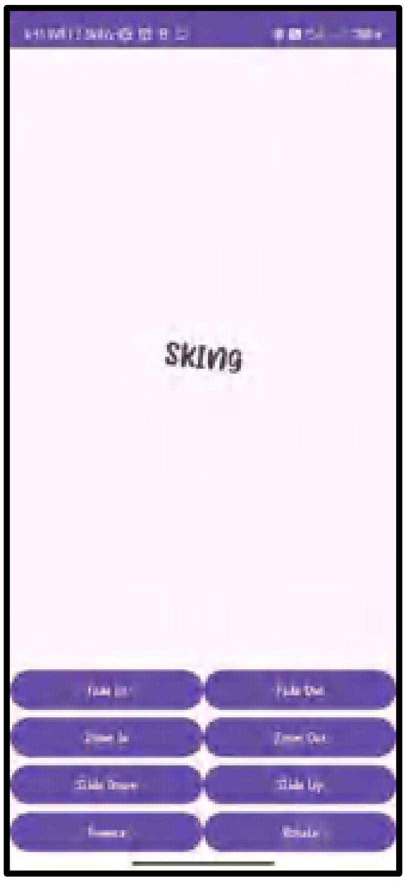
OUTPUT:-











PRACTICAL NO 7

AIM:- a) Create a media player application in android that plays audio. Implement play, pause, and loop features.

INTRODUCTION:-

The Android multimedia framework includes support for playing variety of common media types, so that you can easily integrate audio, video and images into your applications. You can play audio or video from media files stored in your application's resources (raw resources), from standalone files in the filesystem, or from a data stream arriving over a network connection, all using MediaPlayer APIs.

This document shows you how to use MediaPlayer to write a media-playing application that interacts with the user and the system in order to obtain good performance and a pleasant user experience.

Alternatively, you might want to use ExoPlayer, which is a customizable open source library that supports high-performance features not available in MediaPlayer

CODE:-

←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/headingText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="32dp"
    android:text="MEDIA PLAYER"
    android:textSize="18sp"
    android:textStyle="bold"/>
  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/headingText"
    android:layout_marginTop="16dp"
    android:gravity="center_horizontal">
    <Button
      android:id="@+id/stopButton"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginEnd="8dp"
      android:text="STOP"
      android:textColor="@android:color/white"/>
    <Button
      android:id="@+id/playButton"
```

```
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:layout_marginEnd="8dp"
          android:text="PLAY"
          android:textColor="@android:color/white"/>
        <Button
          android:id="@+id/pauseButton"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:text="PAUSE"
          android:textColor="@android:color/white"/>
    </LinearLayout>
</RelativeLayout>
```

←MainActivity.kt→

```
package com.example.prac7a

import android.media.MediaPlayer
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Button

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

        //createaninstanceofmediplayerforaudioplayback
        val mediaPlayer: MediaPlayer=MediaPlayer.create(applicationContext,R.raw.music)

        //registerallthebuttonsusingtheirappropriateIDs
        val bPlay: Button=findViewById(R.id.playButton)
        val bPause: Button=findViewById(R.id.pauseButton)
        val bStop: Button=findViewById(R.id.stopButton)

        //handlethestartbuttonto
        //starttheaudioplayback
        bPlay.setOnClickListener{
          //startmethodisusedtostart
          //playingtheaudiofile
          mediaPlayer.start()
        }
        //handlethepausebuttontoputthe
        //MediaPlayerinstanceatthePausestate

        bPause.setOnClickListener{
          //pause()methodcanbeusedto
          //pausethemediaplyerinstance
          mediaPlayer.pause()
        }

        //handlethestopbuttontostopplaying
        //andpreparethemediaplayerinstance
        //forthenextinstanceofplay
        bStop.setOnClickListener{
          //stop()methodisusedtocompletely
          //stopplayingthemediaplayerinstance
          mediaPlayer.stop()
          //afterstoppingthemediaplayerinstance
          //itisagainneedtobeprepared
          //forthenextinstanceofplayback
          mediaPlayer.prepare()
        }
    }
}
```

AIM:- b) Create an Android application to use a camera and capture image/video and display them on the screen.

INTRODUCTION:-
The Camera class is used to set image capture settings, start/stop preview, snap pictures, and retrieve frames for encoding for video. This class is a client for the Camera service, which manages the actual camera hardware.
To access the device camera, you must declare the android.Manifest.permission#CAMERA permission in your Android Manifest. Also be sure to include the <uses-feature> manifest element to declare camera features used by your application.
This class is not thread-safe, and is meant for use from one event thread. Most long-running operations (preview, focus, photo capture, etc) happen asynchronously and invoke callbacks as necessary. Callbacks will be invoked on the event thread open(int) was called from. This class's methods must never be called from multiple threads at once.

CODE:-

←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">

    <!-- add Camera Button to open the Camera -->
    <Button
        android:id="@+id/camera_button"
        android:layout_width="100dp"
        android:layout_height="50dp"
        android:layout_marginStart="150dp"
        android:text="Camera" />

    <!-- add ImageView to display the captured image -->
    <ImageView
        android:id="@+id/click_image"
        android:layout_width="350dp"
        android:layout_height="450dp"
        android:layout_marginStart="30dp"
        android:layout_marginTop="70dp"
        android:layout_marginBottom="10dp" />
</RelativeLayout>
```

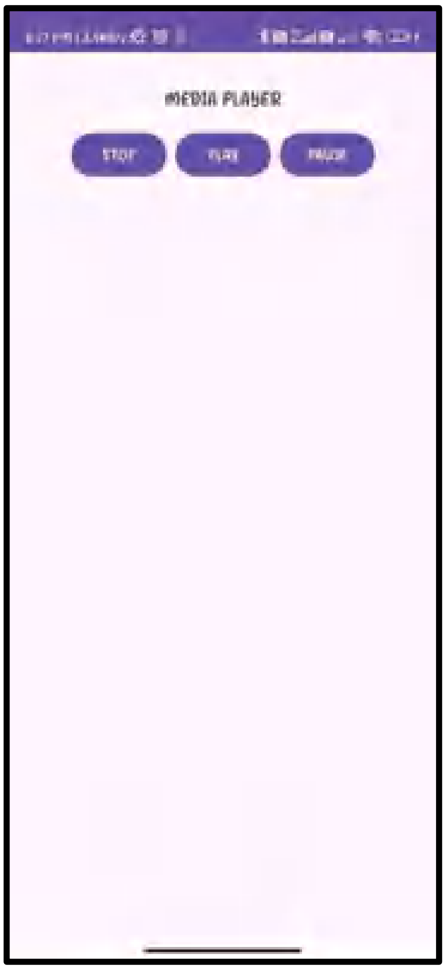
←MainActivity.kt→

```
package com.example.prac7b

import android.content.Intent
import android.graphics.Bitmap
import android.os.Bundle
import android.provider.MediaStore
import android.view.View
import android.widget.Button
import android.widget.ImageView
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {
    // Define the button and imageview type variable
    private lateinit var cameraOpenId: Button
    lateinit var clickImageId: ImageView
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        // By ID we can get each component which id is assigned in XML file get Buttons and imageview.
        cameraOpenId = findViewById(R.id.camera_button)
        clickImageId = findViewById(R.id.click_image)

        // Camera_open button is for open the camera and add the setOnClickListener in this button
        cameraOpenId.setOnClickListener(View.OnClickListener { v: View? ->
            // Create the camera_intent ACTION_IMAGE_CAPTURE it will open the camera for capture the image
            val cameraIntent = Intent(MediaStore.ACTION_IMAGE_CAPTURE)
            // Start the activity with camera_intent, and request pic id
            startActivityForResult(cameraIntent, pic_id)
        })
    }

    // This method will help to retrieve the image
    override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
        super.onActivityResult(requestCode, resultCode, data)
        // Match the request 'pic id with requestCode
        if (requestCode == pic_id) {
            // BitMap is data structure of image file which store the image in memory
            val photo = data!!.extras!!["data"] as Bitmap?
            // Set the image in imageview for display
            clickImageId.setImageBitmap(photo)
        }
    }

    companion object {
        // Define the pic id
        private const val pic_id = 123
    }
}
```

PRACTICAL NO 7

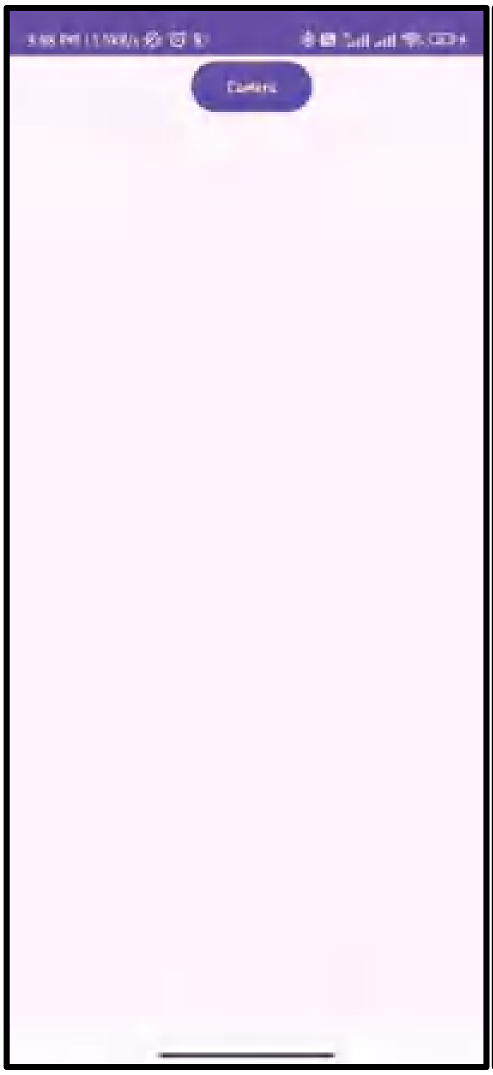
AIM:- a) Create a media player application in android that plays audio. Implement play, pause, and loop features.

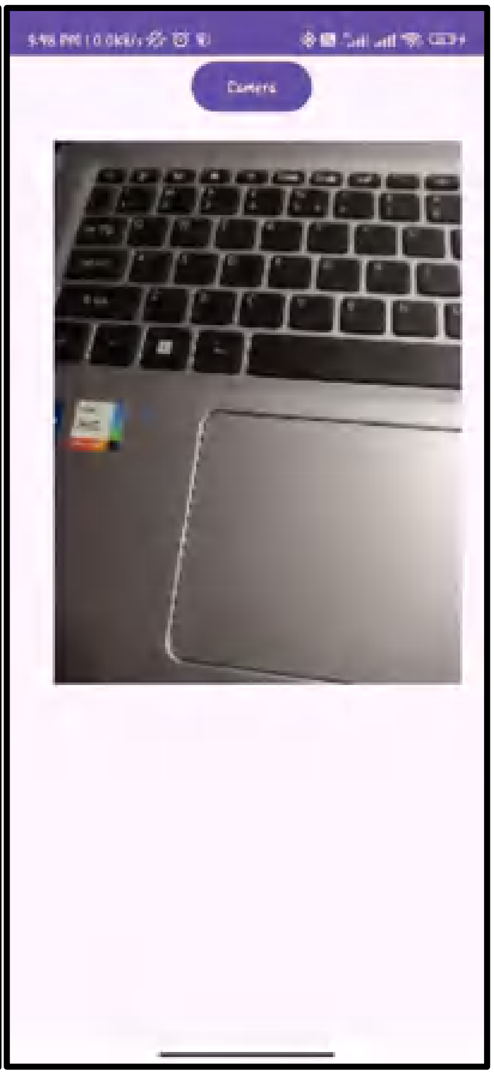
OUTPUT:-



AIM:- b) Create an Android application to use a camera and capture image/video and display them on the screen.

OUTPUT:-





PRACTICAL NO 8

AIM:- a) Create an android application to implement Asynctask and threading concepts.

INTRODUCTION:-

Kotlin offers a feature called coroutines to help with asynchronous programming. Coroutines are lightweight threads that can be suspended and resumed, making them ideal for long-running tasks that don't need to block the main thread.
To use coroutines, you first need to define a suspend function. A suspend function is a function that can be suspended and resumed. You can suspend a function by calling the suspend keyword.
Once you have defined a suspend function, you can start it using the async keyword. The async keyword will start the coroutine in the background and return a Deferred object. The Deferred object represents the result of the coroutine.
You can use the await keyword to wait for the result of a coroutine. The await keyword will suspend the current coroutine until the result of the other coroutine is available.

CODE:-

←activity_main.xml→

```
<?xml version = "1.0" encoding = "utf-8"?>
<LinearLayout xmlns:android = "http://schemas.android.com/apk/res/android"
   xmlns:tools = "http://schemas.android.com/tools"
   android:id = "@+id/rootview"
   android:layout_width = "match_parent"
   android:layout_height = "match_parent"
   android:orientation = "vertical"
   android:background = "#c1c1c1"
   android:gravity = "center_horizontal"
   tools:context = ".MainActivity">
   <Button
      android:id = "@+id/asyncTask"
      android:text = "Download"
      android:layout_width = "wrap_content"
      android:layout_height = "wrap_content" />
   <ImageView
      android:id = "@+id/image"
      android:layout_width = "300dp"
      android:layout_height = "300dp" />
</LinearLayout>
```

←MainActivity.kt→

```
package com.example.prac8a
import android.app.ProgressDialog
import android.graphics.Bitmap
import android.graphics.BitmapFactory
import android.net.wifi.WifiConfiguration.AuthAlgorithm.strings
import android.os.AsyncTask
import android.os.Bundle
import android.support.v7.app.AppCompatActivity
import android.widget.Button
import android.widget.ImageView
import java.io.IOException
import java.io.InputStream
import java.net.HttpURLConnection
import java.net.URL
enum class AsyncTaskExample {
}

class MainActivity : AppCompatActivity() {
    var ImageUrl: URL? = null
    var `is`: InputStream? = null
    var bmImg: Bitmap? = null
    var imageView: ImageView? = null
    var p: ProgressDialog? = null
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val button = findViewById<Button>(R.id.asyncTask)
        imageView = findViewById(R.id.image)
        button.setOnClickListener {
            val asyncTask: AsyncTaskExample = this.AsyncTaskExample() {
            }
        }
        abstract class AsyncTaskExample :
            AsyncTask<String?, String?, Bitmap?>() {
            override fun onPreExecute() {
                super.onPreExecute()
                p = ProgressDialog(this@MainActivity)
                p!!.setMessage("Please wait...It is downloading")
                p!!.isIndeterminate = false
                p!!.setCancelable(false)
                p!!.show()
            }

            protected override fun doInBackground(vararg p0: String?): Bitmap? {
                try {
                    ImageUrl = URL(strings[0])
                    val conn = ImageUrl!!.openConnection() as HttpURLConnection
                    conn.doInput = true
                    conn.connect()
                    `is` = conn.inputStream
                    val options = BitmapFactory.Options()
                    options.inPreferredConfig = Bitmap.Config.RGB_565
                    bmImg = BitmapFactory.decodeStream(`is`, null, options)
                } catch (e: IOException) {
                    e.printStackTrace()
                }
                return bmImg
            }

            override fun onPostExecute(bitmap: Bitmap?) {
                super.onPostExecute(bitmap)
                if (imageView != null) {
                    p!!.hide()
                    imageView!!.setImageBitmap(bitmap)
                } else {
                    p!!.show()
                }
            }
        }
    }

    private fun AsyncTaskExample(function: () -> Unit) {
        "TODO(\"Not yet implemented\")"
    }
}
```

AIM:- b) Create an Android application to demonstrate the different types of menus.

1. Pop-up Menu
2. Context Menu
3. Option Menu

INTRODUCTION:-

In Android, there are three types of menus available to define a set of options and actions in the Android apps. The lists of menus in Android applications are the following:

Android options menu
Android context menu
Android popup menu

In Android, the context menu is like a floating menu and arises when the user has long-pressed or clicked on an item and is beneficial for implementing functions that define the specific content or reference frame effect. The Android context menu is alike to the right-click menu in Windows or Linux. In the Android system, the context menu provides actions that change a specific element or context frame in the user interface and one can provide a context menu for any view. The context menu will not support any object shortcuts and object icons.

CODE:-

←activity_main.xml→

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.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=".MainActivity">
    <include layout="@layout/content_main" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

←content_main.xml→

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context=".MainActivity"
    tools:showIn="@layout/activity_main">
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:text="Click Me:)"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

←strings.xml→

```
<resources>
    <string name="app_name">PopupMenu</string>
    <string name="action_settings">Settings</string>
    <string name="action_cricket">Cricket</string>
    <string name="action_football">Football</string>
    <string name="action_hockey">Hockey</string>
</resources>
```

←popup_menu.xml→

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/action_crick"
        android:title="@string/action_cricket"
        app:showAsAction="never"/>
    <item
```

```
        android:id="@+id/action_ftbal"
        android:title="@string/action_football"
        app:showAsAction="never"/>
    <item
        android:id="@+id/action_hockey"
        android:title="@string/action_hockey"
        app:showAsAction="never"/>
</menu>
```

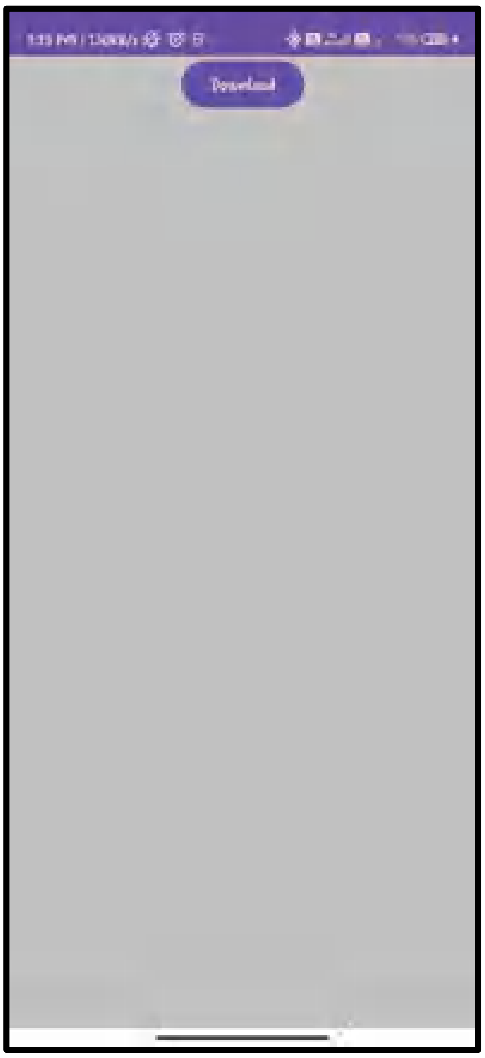
←MainActivity.ky→

```
package com.example.prac8b

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Button
import android.widget.PopupMenu
import android.widget.Toast

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

        val button = findViewById<Button>(R.id.button)

        button.setOnClickListener {
            val popupMenu: PopupMenu = PopupMenu(this,button)
            popupMenu.menuInflater.inflate(R.menu.popup_menu,popupMenu.menu)
            popupMenu.setOnMenuItemClickListener(PopupMenu.OnMenuItemClickListener {
                    item -> when(item.itemId) {
                R.id.action_crick ->
                    Toast.makeText(this@MainActivity, "You Clicked : " + item.title,
                        Toast.LENGTH_SHORT).show()
                R.id.action_ftbal ->
                    Toast.makeText(this@MainActivity, "You Clicked : " + item.title,
                        Toast.LENGTH_SHORT).show()
                R.id.action_hockey ->
                    Toast.makeText(this@MainActivity, "You Clicked : " + item.title,
                        Toast.LENGTH_SHORT).show()
            }
                true
            })
            popupMenu.show()
        }
    }
}
```

PRACTICAL NO 8

AIM:- a) Create an android application to implement Asynctask and threading concepts.

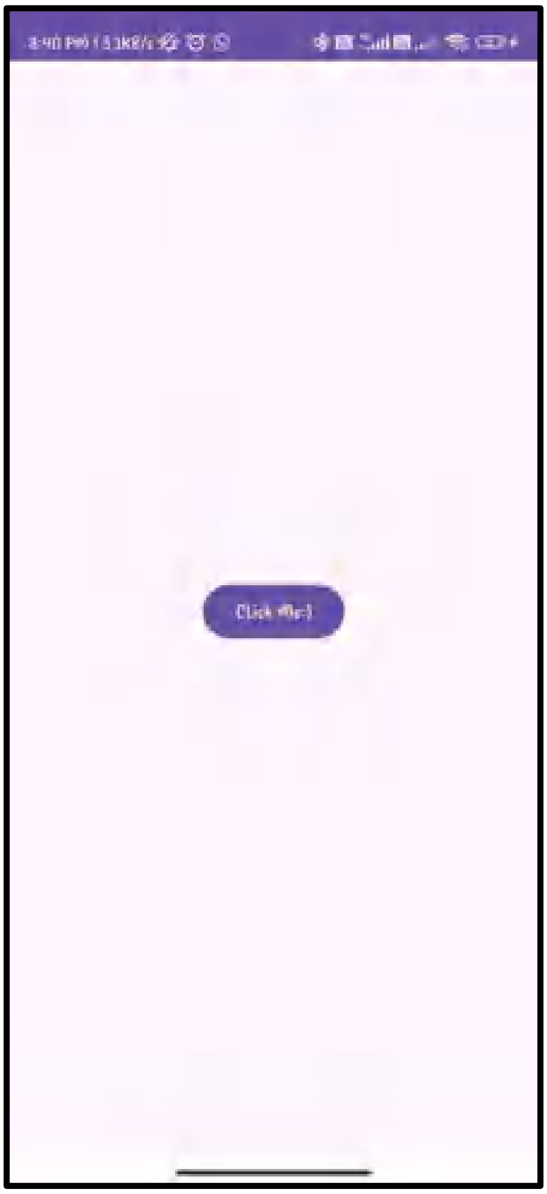
OUTPUT:-

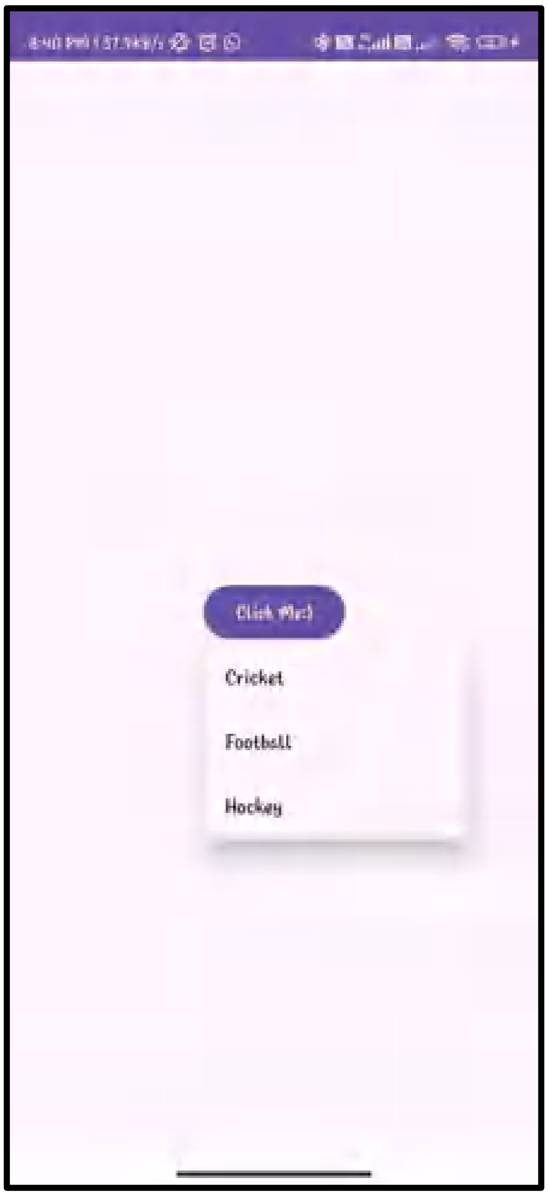


AIM:- b) Create an Android application to demonstrate the different types of menus.

1. Pop-up Menu
2. Context Menu
3. Option Menu

OUTPUT:-




Click Me:)
Cricket
Football
Hockey