Q-1 Assign “Red” color from resource as button background & assign “Send” string from resource to text value of button in activity_main.xml file. Create two Colors (Red, Blue), three strings (Sign In, Registration, Send) in corresponding xml files of resource. (note:- write only xml files, use Linear Layout for root in activity_main.xml file)

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:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@color/Red"
        android:text="@string/send"
        />

</LinearLayout>
```

strings.xml

```
<resources>
    <string name="sign_in">Sign In</string>
    <string name="reg">Registration</string>
    <string name="send">Send</string>
</resources>
```

colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="Red">#FF0000</color>
    <color name="Blue">#0000FF</color>
</resources>
```

Q-2 List out basic types of Animation. Create tween Animation on ImageView with image (name as ceitclass.png) according to below instruction:

1. **Rotate from center point of ImageView from 0 degree to 180 degree for 3000ms duration.**
2. **After completion of rotation, scale size of ImageView to double from center point of ImageView for 3000ms duration.**

Write down necessary XML (only necessary tages) & kotlin files (Only necessary methods) for program.

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"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    tools:context=".MainActivity">

    <ImageView
        android:id="@+id/ceitclass"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/ceitclass"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"/>

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

image_anim_rotate.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:interpolator="@android:anim/linear_interpolator">
    <rotate
        android:fromDegrees="0"
        android:toDegrees="180"
        android:pivotX="50%"
        android:pivotY="50%"
        android:duration="3000"/>
</set>
```

image_anim_scale.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
android:interpolator="@android:anim/linear_interpolator">
    <scale
        android:fromXScale="0.0"
        android:fromYScale="0.0"
        android:toXScale="2.0"
        android:toYScale="2.0"
        android:pivotX="50%"
        android:pivotY="50%"
        android:duration="3000"/>
</set>
```

MainActivity.kt

```
package com.example.madmid2q1

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.animation.Animation
import android.view.animation.AnimationUtils
import android.widget.ImageView

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        var iView = findViewById<ImageView>(R.id.ceitclass)
        var animRotate =
AnimationUtils.loadAnimation(applicationContext,R.anim.image_anim_rotate)
        animRotate.setAnimationListener(object :Animation.AnimationListener{
            override fun onAnimationStart(p0: Animation?) {}

            override fun onAnimationRepeat(p0: Animation?) {}

            override fun onAnimationEnd(p0: Animation?) {
                var animScale =
AnimationUtils.loadAnimation(applicationContext,R.anim.image_anim_scale)
                iView.startAnimation(animScale)
            }
        })
        iView.startAnimation(animRotate)
    }
}
```

Q-3 Write procedure (steps) to display list in ListView using string array <string-array name=”month_list”>.

Step1: Create string array in strings.xml file

```
<resources>
    <string-array name="month_list">
        <item>Jan</item>
        <item>Feb</item>
        <item>Mar</item>
        <item>Apr</item>
        <item>May</item>
        <item>Jun</item>
        <item>Jul</item>
        <item>Aug</item>
        <item>Sep</item>
        <item>Oct</item>
        <item>Nov</item>
        <item>Dec</item>
    </string-array>
</resources>
```

Step2: Create ListView inside activity_main.xml file and assign id of that ListView

```
<?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"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    tools:context=".MainActivity">
    <ListView
        android:id="@+id/list_view1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toTopOf="parent"
        android:layout_marginHorizontal="20dp"
        android:layout_marginVertical="40dp"/>

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

Step3: Create one layout file that consist layout of item that available inside ListView.

```
<?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">
    <TextView
        android:id="@+id/text_view1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:padding="10dp"
        />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Step4: Bind Item with ArrayAdapter and then bind with ListView.

```
package com.example.madmid2q1

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.ArrayAdapter
import android.widget.ListView

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        var lView = findViewById<ListView>(R.id.list_view1)
        var monthArray = resources.getStringArray(R.array.month_list)
        val monAdapter = ArrayAdapter(
            this,
            R.layout.list_view_item, R.id.text_view1, monthArray
        )
        lView.adapter=monAdapter
    }
}
```

Q-4: Create Class "MyBroadcastReceiver" to explain BroadcastReceiver. Write down code to register MyBroadcastReceiver in Android Manifest file.

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.

Once a matching intent is broadcasted, the receiver wakes up and receives a call to the onReceive () method. In this method, the receiver would **process the data in the intent, perform any necessary actions, then return**. Once that method returns, the receiver goes back to sleep until another matching intent is broadcasted.

Creating the Broadcast Receiver:

```
package com.example.madmid2q1

import android.content.BroadcastReceiver
import android.content.Context
import android.content.Intent

class MyBroadcastReceiver : BroadcastReceiver() {

    override fun onReceive(context: Context, intent: Intent) {
    }
}
```

Register BroadcastReceiver in Android Manifest file:

```
<application
   android:icon="@drawable/ic_launcher"
   android:label="@string/app_name"
   android:theme="@style/AppTheme" >
   <receiver android:name="MyBroadcastReceiver">

      <intent-filter>
         <action android:name="android.intent.action.BOOT_COMPLETED">
         </action>
      </intent-filter>

   </receiver>
</application>
```