中山大学移动信息工程学院本科生实验报告

（2017 年秋季学期）

课程名称：移动应用开发　　任课教师： 郑贵锋

年级	15 级	专业（方向）	移动
学号	15352385	姓名	杨与瑕
电话	15013045984	Email	913219727@qq.com
开始日期	2017.9.26	完成日期	2.17.9.28

一．实验题目

基本 UI 界面设计

二．实现内容

实现一个 Android 应用，界面呈现如下效果：

要求：

（1）该界面为应用启动后看到的第一个界面

（2）各控件的要求如下：

要求只用一个 ConstraintLayout 实现整个布局；

标题字体大小 20sp，与顶部距离 20dp，居中；图片与标题的间距为 20dp，居中；输入框整体距屏幕右边间距 20dp，上下两栏间距 20dp，内容（包括提示内容）如图所示，内容字体大小 18sp；

学号对应的 EditText 只能输入数字，密码对应的 EditText 输入方式为密码；

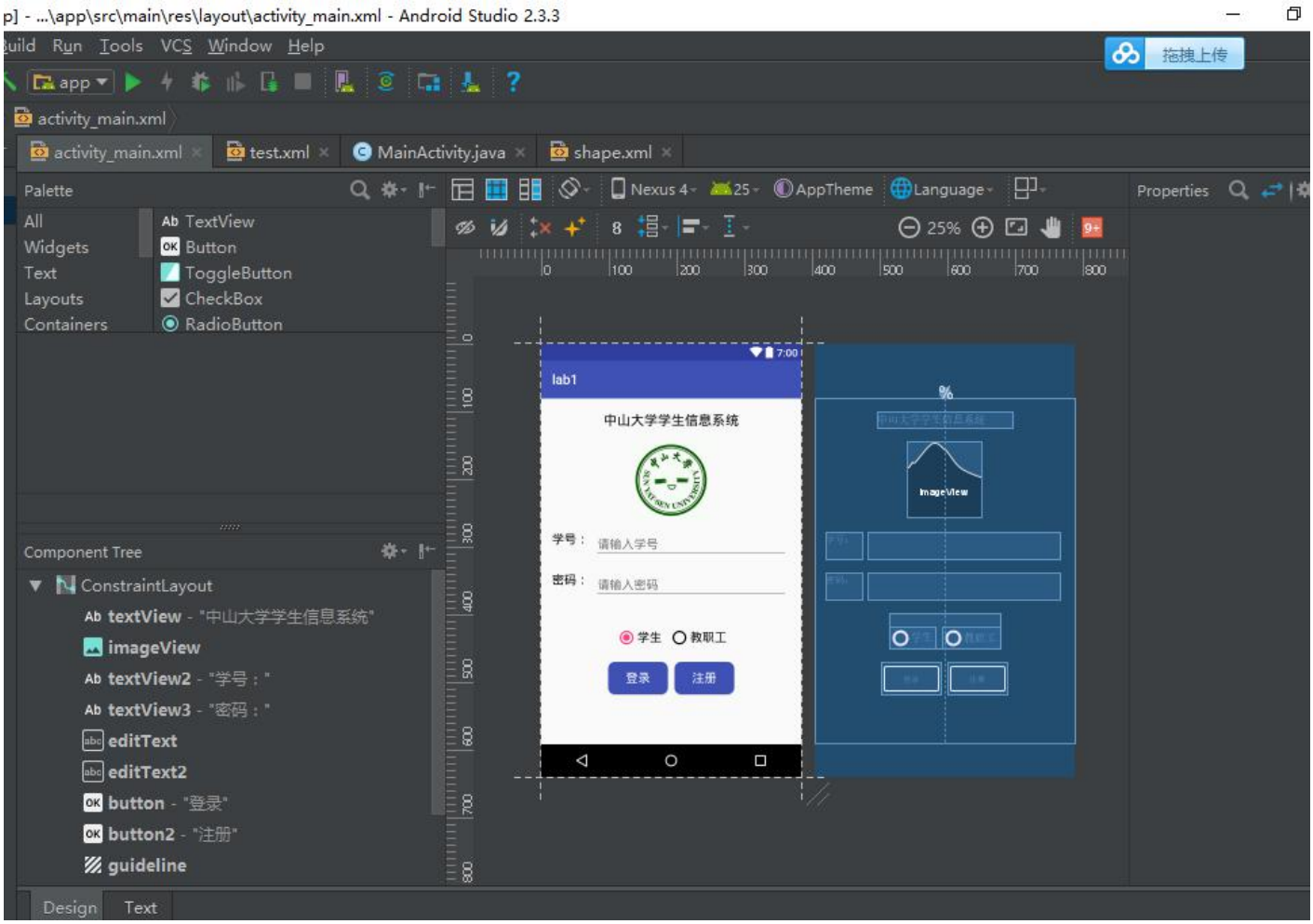
两个单选按钮整体居中，字体大小 18sp，间距 10dp，默认选中的按钮为第一个；

两个按钮整体居中，与上方控件间距 20dp，按钮间的间距 10dp，文字大小 18sp。按钮背景框左右边框与文字间距 10dp，上下边框与文字间距 5dp，圆角半径 10dp，背景色为#3F51B5

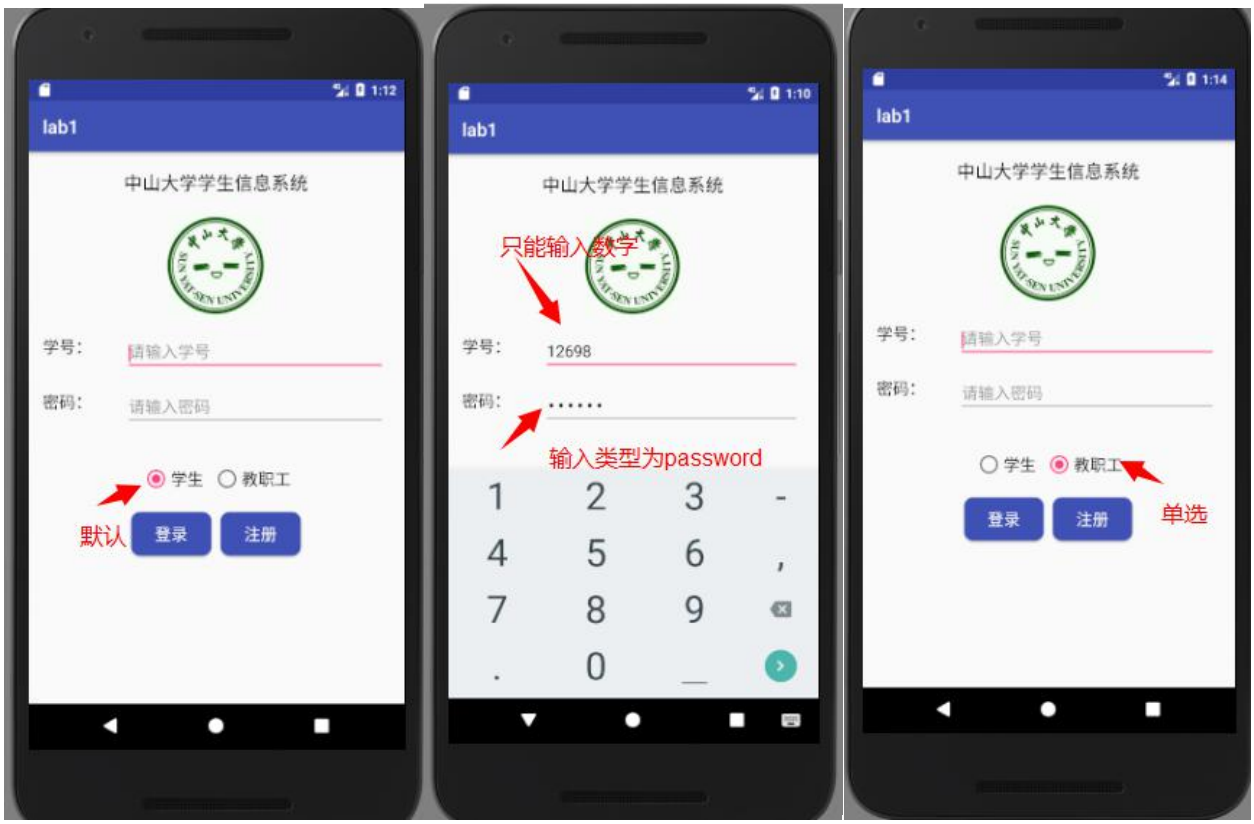
（3）使用的布局和控件：ConstraintLayout、TextView、EditText、Button、ImageView、RadioGroup、RadioButton

三．课堂实验结果

（1）实验截图



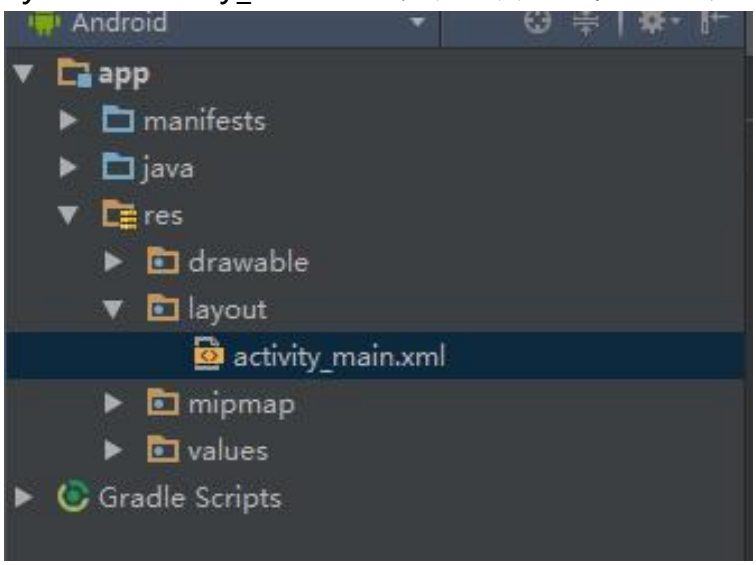
虚拟机运行结果：



（2）实验步骤以及关键代码

按照实验内容所给的界面从上到下进行实验。

首先 layout 布局是使用要求的一个 Constraint_Layout 约束布局。打开 res->layout->activity_main.xml，默认就是约束布局。无需调整。

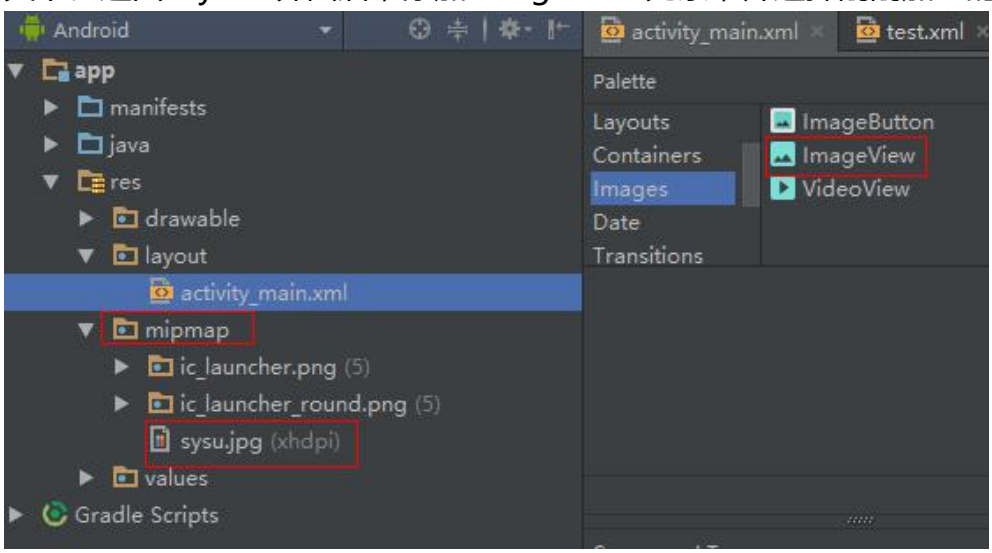


初始时已经有一个程序 textview，先选中这个 textview 左右用弹簧连接到屏幕的左右边界，再将顶部连接到屏幕的顶部，切换到代码界面，修改相关参数，如边界距离，字体内容，字体大小等。

```
<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="中山大学学生信息系统"
    android:textColor="#000000"
    android:textSize="20sp"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    android:layout_marginTop="20dp" />
```

这里的居中是指垂直居中，此时左右边界与 constraintLayout(父容器)边界对齐，则为居中。
注意：在约束布局中使用 gravity=“center”无效果

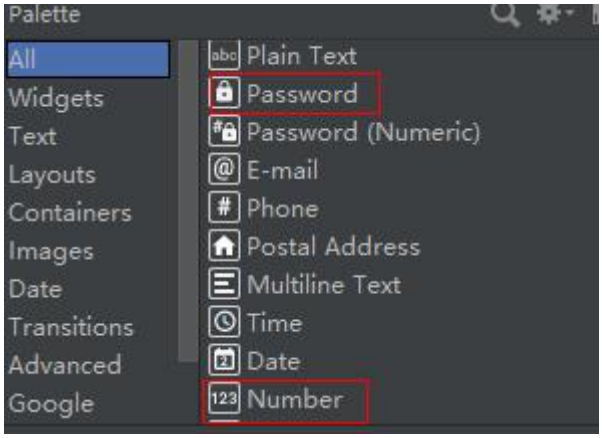
接着添加图片元素到 res->mipmap 中，理论课上讲过可以直接 ctrl c+v 添加到相应文件夹下。返回 layout 界面后，添加 imageview 元素，并选择刚刚加入的图片。



同样，左右两端拉弹簧对齐父容器边界，左右居中。顶部连接到上一个 textview。然后在代码界面确定相关参数。

```
<ImageView
    android:id="@+id/imageView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:srcCompat="@mipmap/sysu"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    android:layout_marginTop="20dp"
    app:layout_constraintTop_toBottomOf="@+id/textView" />
```

接下来是输入框部分，由于只能使用一个布局文件，所以直接创建两个 textview 和两个 exitview。实验要求学号的输入形式只能为数字，选择“Numeber”，密码的输入类型选择“Password”。



把四个控件拉到可视化界面之后，分为文本和输入两个部分来拉弹簧。首先是两个 textview 连接在一起并且分别连接到左边界，上面的 textview 连接到 imageview 的底部，（控件无需用四个弹簧来控制位置，最少只需要两个弹簧分别控制 X 和 Y 的位置即可）。然后是两个输入框连接在一起并且分别连接到右边界，上面的输入框连接到 imageview 的底部。（无需把 textview 和对应的 exitview 连接起来，因为两者的位置都已经可以确定了。）

```
<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="40dp"
    android:text="学号:"
    android:textColor="#000000"
    android:textSize="18sp"
    android:layout_marginTop="20dp"
    app:layout_constraintTop_toBottomOf="@+id/imageView"
    app:layout_constraintLeft_toLeftOf="parent"
    android:layout_marginLeft="16dp"
    android:layout_marginStart="16dp" />
<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="40dp"
    android:text="密码:"
    android:textColor="#000000"
    android:textSize="18sp"
    android:layout_marginTop="20dp"
    app:layout_constraintTop_toBottomOf="@+id/textView2"
    android:layout_marginLeft="16dp"
    app:layout_constraintLeft_toLeftOf="parent"
    android:layout_marginStart="16dp" />
```

```
<EditText
    android:id="@+id/editText"
    android:layout_width="285dp"
    android:layout_height="40dp"
    android:ems="10"
    android:hint="请输入学号"
    android:textSize="18sp"
    android:inputType="number"
    android:layout_marginTop="20dp"
    app:layout_constraintTop_toBottomOf="@+id/imageView"
    android:layout_marginRight="20dp"
    app:layout_constraintRight_toRightOf="parent"
    android:layout_marginEnd="20dp" />
<EditText
    android:id="@+id/editText2"
    android:layout_width="285dp"
    android:layout_height="40dp"
    android:ems="10"
    android:hint="请输入密码"
    android:textSize="18sp"
    android:inputType="textPassword"
    android:layout_marginRight="20dp"
    app:layout_constraintRight_toRightOf="parent"
    android:layout_marginTop="20dp"
    app:layout_constraintTop_toBottomOf="@+id/editText"
    android:layout_marginEnd="20dp" />
```

为了视图效果，需要拉长输入框，对应修改参数即可。

接下来的部分是单选键 Radio，为了实现 Radio 的互斥性，需要将两个单选键 RadioButton 放在一个组里面即 RadioGroup。

因为两个单选键是一个整体，所以居中设置和之前的 imageview 一样把 RadioGroup 左右边界和父容器对齐，上方则用弹簧和之前的 exitview（密码输入框）连接，在代码中设置精准的参数。

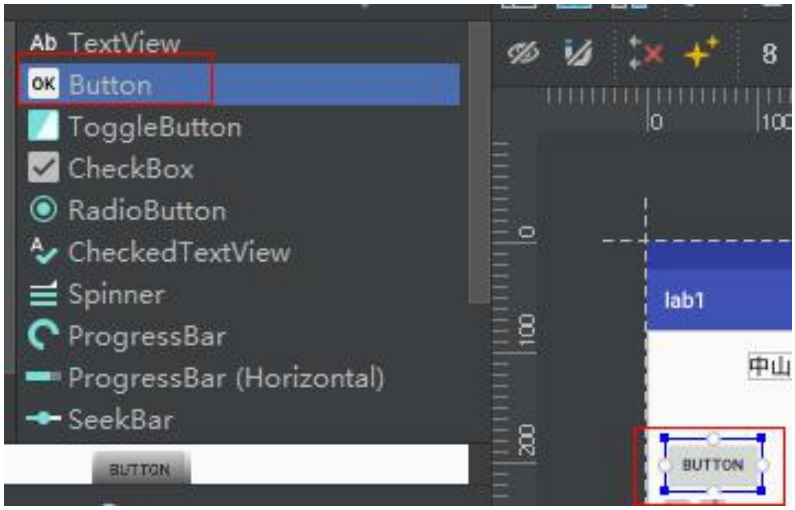
```
<RadioGroup
    android:id="@+id/radioGroup"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:orientation="horizontal"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/editText2">
```

这里的 orientation 是指单选键的一个排列模式，此时为水平的。

```
    <RadioButton
        android:id="@+id/radioButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginRight="10dp"
        android:layout_marginTop="20dp"
        android:text="学生"
        android:textSize="18sp"
        android:checked="true"
        app:layout_constraintLeft_toLeftOf="@+id/radioGroup"
        app:layout_constraintRight_toLeftOf="@+id/radioButton2"
        app:layout_constraintTop_toBottomOf="@+id/editText2" />
    <RadioButton
        android:id="@+id/radioButton2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:text="教职工"
        android:textSize="18sp"
        app:layout_constraintRight_toRightOf="@+id/radioGroup"
        app:layout_constraintTop_toBottomOf="@+id/editText2" />
</RadioGroup>
```

设置为默认选项

最后是 Button 的设计。选择 Button 这个控件拉到可视化编辑界面中，发现 Button 的形状需要调整，所以另外在 res->drawable 中新建一个 xml 文件来设置 Button 的 background。




```xml
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <solid android:color="#3F51B5" /><!— 填充的颜色 —>
    <corners android:radius="10dp" /><!—圆角边框—>
    <padding
        android:left="10dp"
        android:top="5dp"
        android:right="10dp"
        android:bottom="5dp"
        />
    <!— padding: Button里面的文字与Button边界的间隔 —>
</shape>
```

然后是要让两个按钮整体居中并且之间间隔 10dp。这里有两种方法，我自己做出来了第一种，就是增加一根 guideline 在屏幕中间，然后两个按钮的分别和其左右两边连接，并且设置距离一样，同时两个按钮都要和上方的 RadiosGroup 连接来确定 Y 距离。

```xml
<android.support.constraint.Guideline
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/guideline"
    android:orientation="vertical"
    app:layout_constraintGuide_percent="0.5"
    tools:layout_editor_absoluteY="0dp"
    tools:layout_editor_absoluteX="192dp" />
```

垂直模式
并且居中

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="登录"
    android:textSize="18sp"
    android:textColor="#FFFFFF"
    android:background="@drawable/shape"
    app:layout_constraintTop_toBottomOf="@+id/radioGroup"
    android:layout_marginTop="20dp"
    app:layout_constraintRight_toLeftOf="@+id/guideline"
    android:layout_marginRight="5dp" />
<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="注册"
    android:textSize="18sp"
    android:textColor="#FFFFFF"
    android:background="@drawable/shape"
    app:layout_constraintTop_toBottomOf="@+id/radioGroup"
    android:layout_marginTop="20dp"
    android:layout_marginLeft="5dp"
    app:layout_constraintLeft_toLeftOf="@+id/guideline" />
```

所有控件都添加完毕，直接使用虚拟机运行即可得到结果图像。

注：应用开启第一个画面即为此画面，main_activity 中无需再设计多余代码，默认的即可。

（3）实验遇到困难以及解决思路

①因为实验文档中写的比较细节了，实验做起来也很流畅。只是开始的时候，以为是一定要把一个控件的上下左右连接起来的，但是后来观察代码（下图），当你在可视区域拖动的时候，代码中会出现 tool 帮你判断但是当你加了一些约束弹簧以后，这两句话就会消失。结论就是其实约束布局它所需要的其实就是用”参照“关系去锁定控件的位置，即 X 和 Y 坐标，所以一个控件只需在两个方向上都有参照对象即可锁定位置。

```
tools:layout_editor_absoluteX="16dp"
tools:layout_editor_absoluteY="98dp"
```

②最开始的时候我想尝试用第二种方法也就是链法来控制后面两个按钮，但没有成功，之后找 TA 询问了，才发现自己的代码出现了问题。修改过后也成功得到了结果。

首先选中两个 Button，右键选择“center Horizontally”生成链条之后，修改链条模式为“packed”之后添加一个 margin 语句设置两者之间的距离。

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="登录"
    android:textSize="18sp"
    android:textColor="#FFFFFF"
    android:background="@drawable/shape"
    app:layout_constraintTop_toBottomOf="@+id/radioGroup"
    android:layout_marginTop="20dp"
    app:layout_constraintHorizontal_chainStyle="packed"
    app:layout_constraintRight_toLeftOf="@+id/button2"
    android:layout_marginRight="10dp"
    app:layout_constraintLeft_toLeftOf="parent"
    />
```

四．课后实验结果

（1）使用 linearLayout 和其他 layout 重做实验。因为在理论课上老师讲了布局的相关知识，除了最新的 constraintLayout 之外还有其余的经典布局，所以我就尝试用其他的布局做了一下实验，其中还包括了布局的嵌套。每个布局都有自己的特性，这次实验使用多种布局嵌套会更方便和规范。过程比较简单，直接贴代码：

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

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

   <TextView
      android:textColor="#000000"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:paddingTop="20dp"
      android:textSize="20sp"
      android:gravity="center"
      android:text="中山大学学生信息系统"/>

   <ImageView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:paddingTop="20dp"
      android:layout_gravity="center"
      android:src="@mipmap/sysu"/>

   <TableLayout
      android:layout_width="match_parent"
      <EditText
         style="@style/textStyle"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:ems="20"
         android:inputType="textPassword"
         android:hint="请输入密码"/>
      </TableRow>

   <RadioGroup
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="10dp"
      android:orientation="horizontal"
      android:gravity="center">
      <RadioButton
         style="@style/textStyle"
         android:id="@+id/id0"
         android:checked="true"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="学生"/>

      <RadioButton
         style="@style/textStyle"
         android:id="@+id/id1"
```

```
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp">
    <TableRow
        android:paddingLeft="20dp"
        android:paddingRight="20dp">
        <TextView
            android:id="@+id/username"
            style="@style/textStyle"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textColor="#000000"
            android:text="学号"/>

        <EditText
            style="@style/textStyle"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="20"
            android:hint="请输入学号"/>
    </TableRow>

    <TableRow
        android:paddingLeft="20dp"
        android:paddingRight="20dp"
        android:layout_marginTop="10dp">
        <TextView
            style="@style/textStyle"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textColor="#000000"
            android:gravity="right"
            android:text="密码:"/>
```

```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="10dp"
            android:text="教职工"/>

        </RadioGroup>
    </TableLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:orientation="horizontal"
        android:gravity="center">
        <Button
            style="@style/textStyle"
            android:id="@+id/id4"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="@drawable/button"
            android:textColor="#FFFFFF"
            android:text="登录 />

        <Button
            style="@style/textStyle"
            android:id="@+id/id5"
            android:layout_marginLeft="10dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="@drawable/button"
            android:textColor="#FFFFFF"
            android:text="注册"/>

    </LinearLayout>

</LinearLayout>
```

（2）创新部分：本来一开始是想要实现点击”登陆”后能够实现页面跳转，但是时间上来不及了，也没啥其他好的想法就简单的给整个页面加了一个渐变色的 background，实现非常简单，重新编写一个 res->drawable 下的文件。

查看了一下颜色表，发现有很多很好看的颜色。

五．实验思考及感想

（1）实验思考：不同布局的对比

线性布局	此布局里面可以放多个控件，但是一行/列只能放一个控件
表单布局	与 TableRow 配合使用，此布局里面可以放多个控件，但是一行（列）只能放一个控件。子元素放入到行与列中，不显示行、列或是单元格边界线，单元格不能横跨行，如 HTML 中一样。
相对布局	让子元素指定它们相对于其他元素的位置(通过 ID 来指定)或相对于父布局对象。跟约束布局类似，使用RelativeLayout 布局的时候，减少程序运行时动态改变控件布局，因为 RelativeLayout 布局里面的属性之间，很容易冲突。
绝对布局	指明子元素确切的屏幕(X,Y)坐标，(0,0)是左上角，下移或右移时，坐标值增加 。(但是不推荐，因为使用要求严格)
约束布局	根据布局中的其他元素或视图, 确定 View 在屏幕中的位置, 受到三类约束, 即其他视图, 父容器(parent), 基准线(Guideline)。 ConstraintLayout 的基本使用方式就是这些, 兼顾 LinearLayout 与 RelativeLayout 的优点, 非常适合构建复杂布局, 降低布局的层级, 加快渲染速度

（2）实验感想

本次实验操作都比较流畅，详细阅读过实验文档后基本上就没什么困难了。今年新出的 constraintLayout 的确有它强大的地方，可视化的编辑其实在大一的时候用 APPInventor 已经接触过了，所以界面的设计并不陌生，Android studio 自动生成代码之后，控件的定位也变得更加的准确了，整体操作比较流畅，整个实验中比较难搞定的就是链条的部分，自己还需要多多摸索。其次就是使用了两种方法来构造界面，虽然 constraintLayout 可视化约束的感觉比较强烈，控件是比较独立存在的，而在第二种中，各种布局的嵌套使得相关的控件能够形成模块更加规范，代码的编辑也更加直接明了，以后的实验可能会多采取各种布局嵌套的方式来完成界面的一个设计。

作业要求：

1. 命名要求: 学号_姓名_实验编号，例如 15330000_林 XX_lab1。
2. 实验报告提交格式为 pdf。
3. 实验内容不允许抄袭，我们要进行代码相似度对比。如发现抄袭，按 0 分处理。