中山大学数据科学与计算机学院本科生实验报告

（2017 年秋季学期）

课程名称：手机应用平台开发　　任课教师： 刘宁

年级	15 级	专业（方向）	软件工程计算机应用
学号	15331236	姓名	马朝露
电话	15521122675	Email	1073627941@qq.com
开始日期	2017/11/30	完成日期	2017/11/30

一、 实验题目

数据存储

二、 实现内容

1、 实验目的

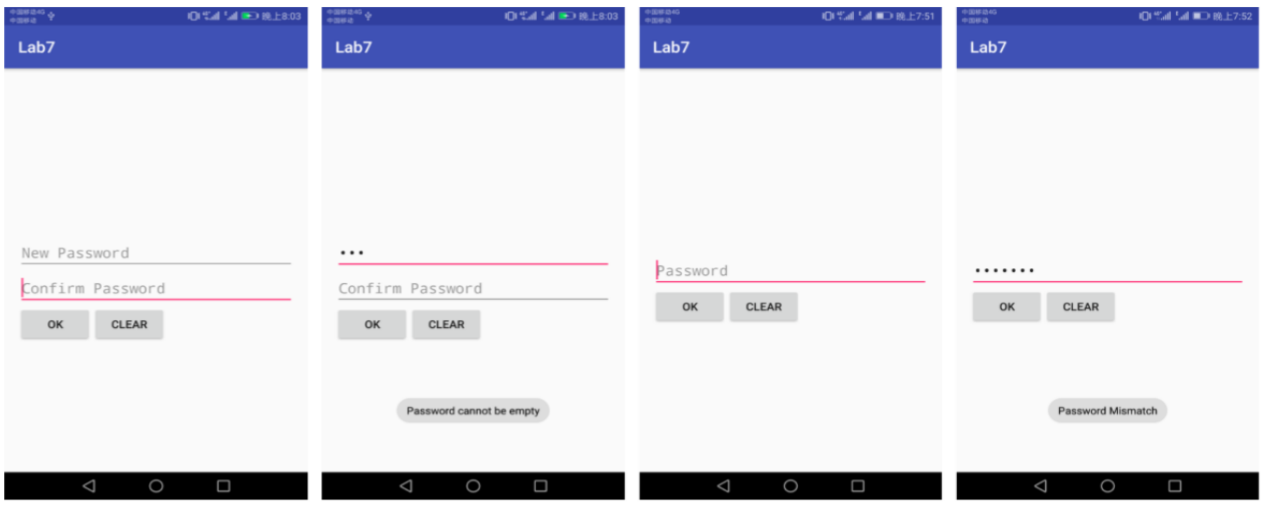
a、 学习 SharedPreferences 的基本使用；

b、 学习 Android 中常见的文件操作方法；

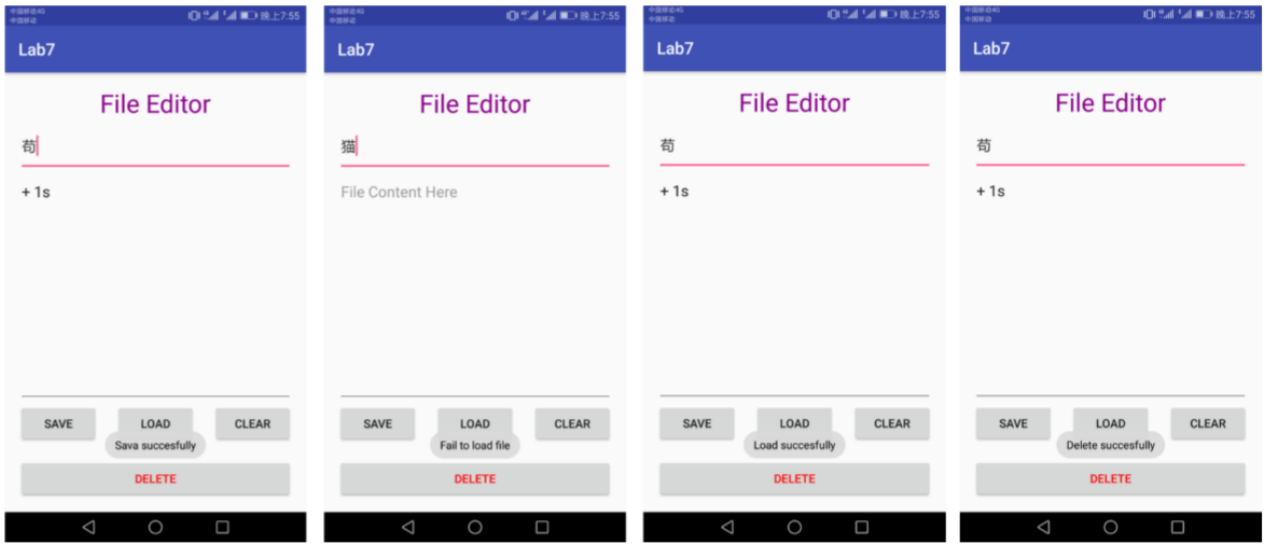
c、 复习 Android 界面编程。

2、 实验内容

从左至右，依次为：初始密码界面、密码为空提示、密码匹配后重新进入界面、密码错误提示。



从左至右，依次为：保存成功提示、写入失败提示、写入成功提示、删除成功提示。



1、 如图所示，本次实验需要实现两个 activity；

2、 首先，需要实现一个密码输入 activity：

a、 如果应用首次启动，则界面呈现出两个输入框，分别为新密码输入和确认密码输入框；

b、 输入框下方有两个按钮：

- OK 按钮，点击之后：
- 若 new password 为空，则弹出密码为空的提示；
- 若 new password 与 comfirm password 不匹配，则弹出不匹配的提示；
- 若密码不为空且互相匹配，则保存密码，进入文件编辑界面。
- CLEAR 按钮，点击之后清除所有输入框的内容。

c、 完成创建密码后，退出应用再进入应用，则只呈现一个密码输入框；

- 点击 OK 按钮后，如果输入的密码与保存的密码不匹配，则弹出 Toast 提示；
- 点击 CLEAR 按钮后，清除密码输入框的内容。

d、 出于学习的目的，我们使用 SharedPreferences 来保存密码，但是在实际应用中我们会用更加 安全的机制来保存这些隐私信息，更多可以参考链接一和链接二。

3、 然后，实现一个文件编辑 activity：

a、 界面底部有两行四个按钮，第一行三个按钮高度一致，顶对齐，按钮水平均匀分布。按钮上方除 了 ActionBar 和 StatusBar 之外的空间由标题和两个 EditText 占据，文件内容编辑的 EditText 需 要占据除去其他控件的全部屏幕空间，且内部文字竖直方向置顶，左对齐；

b、 在文件名输入框内输入文件名，在文件内容编辑区域输入任意内容，点击 SAVE 按钮后能够保存 到指定文件，成功保存后弹出 Toast 提示；

c、 点击 CLEAR 按钮，能够清空文件内容编辑区域内的内容；

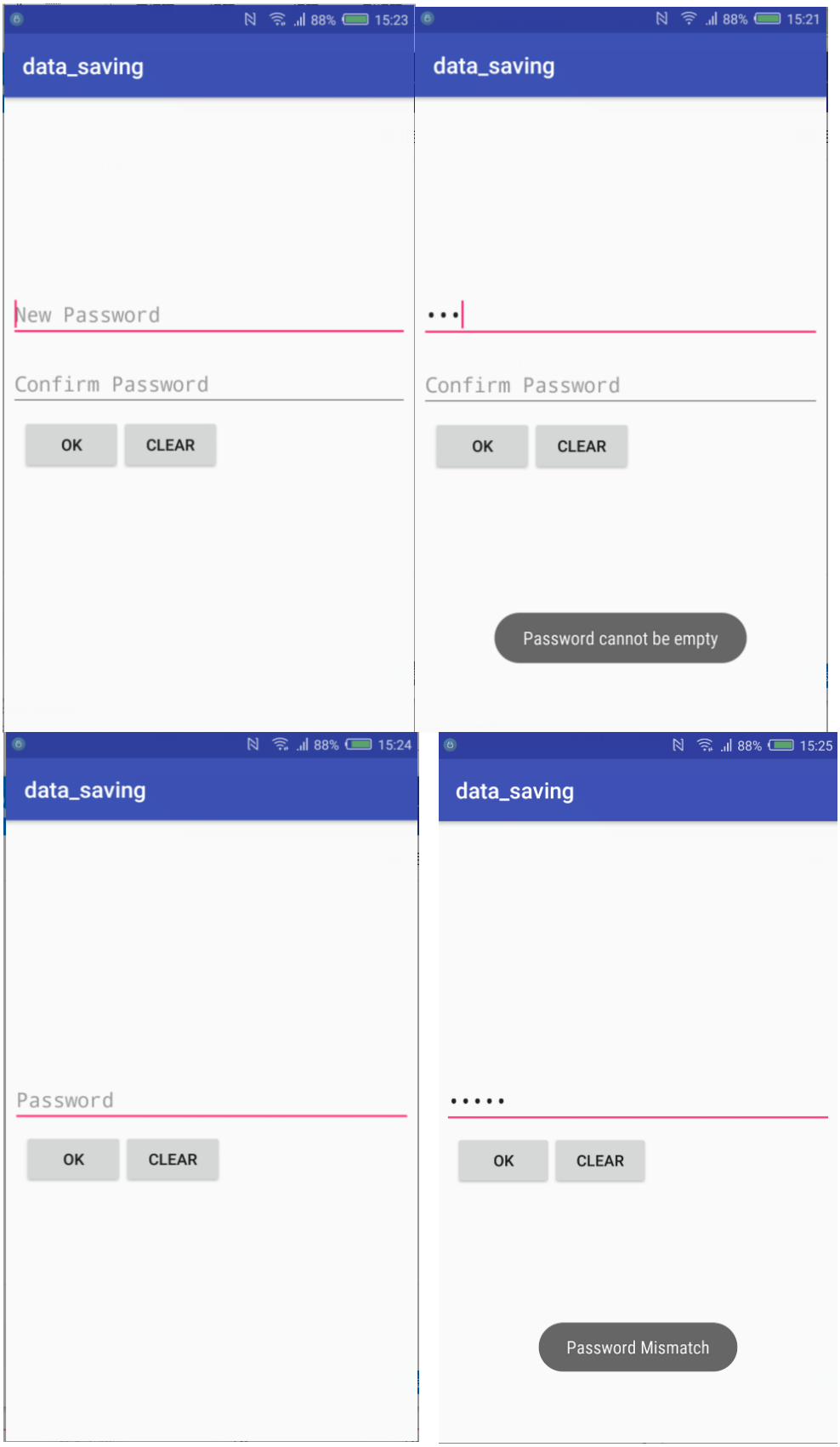
d、 点击 LOAD 按钮，能够按照文件名从内存中读取文件内容，并将文件内容写入到编辑框中。如 果成功导入，则弹出成功的 Toast 提示，如果导入失败（例如：文件不存在） ，则弹出读取失败 的 Toast 提示。

e、 点击 DELETE 按钮，能够按照文件名从内容中删除文件，删除文件后再载入文件，弹出导入失败 的 Toast 提示。

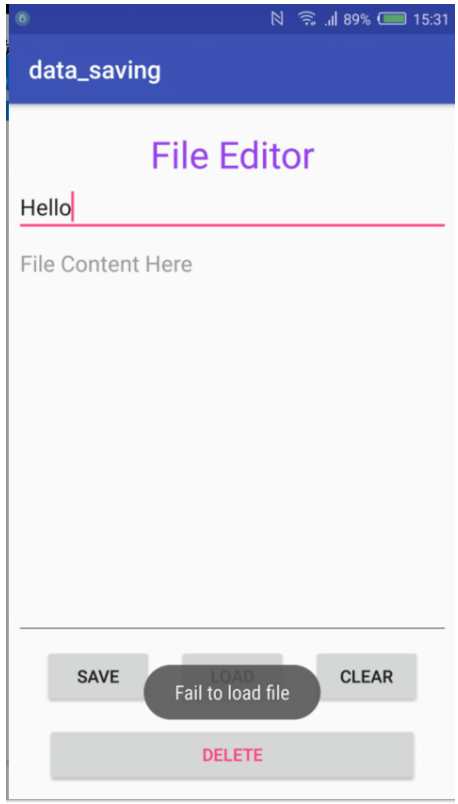
4、 特殊要求：进入文件编辑的 Activity 之后，如果点击返回按钮，则直接返回 Home 界面，不再返回 密码输入界面。

三、 课堂实验结果

（1） 实验截图



从左至右，依次为：初始密码界面、密码为空提示、密码匹配后重新进入界面、密码错误提示。



从左至右，依次为：保存成功提示、写入失败提示、写入成功提示、删除成功提示。

（2） 实验步骤以及关键代码

【1】 界面的设计

保存密码界面：

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.zhaoluma.data_saving.data_saving">

    <EditText
        android:layout_width="350dp"
        android:layout_height="wrap_content"
        android:id="@+id/password"
        android:hint="New Password"
        android:inputType="textPassword"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        android:layout_marginTop="168dp" />
    <EditText
        android:layout_width="350dp"
        android:layout_height="wrap_content"
        android:id="@+id/con_password"
        android:inputType="textPassword"
        app:layout_constraintLeft_toLeftOf="parent"
        android:hint="Confirm Password"
        app:layout_constraintRight_toRightOf="parent"
        android:layout_marginTop="8dp"
        app:layout_constraintTop_toBottomOf="@+id/password"
        app:layout_constraintBottom_toBottomOf="parent"
        android:layout_marginBottom="8dp"
        app:layout_constraintVertical_bias="0.027" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/OK"
        android:text="OK"
        app:layout_constraintBottom_toBottomOf="parent"
        android:layout_marginBottom="230dp"
        android:layout_marginLeft="15dp"
        app:layout_constraintLeft_toLeftOf="parent" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/clear"
        android:text="CLEAR"
        app:layout_constraintBottom_toBottomOf="parent"
        android:layout_marginBottom="230dp"
        android:layout_marginRight="170dp"
        app:layout_constraintRight_toRightOf="parent" />

</android.support.constraint.ConstraintLayout>
```

使用约束布局，这里我使用了 design 来设计界面。

文件编辑界面：

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical" android:layout_width="match_parent"
    android:weightSum="5"
    android:layout_height="match_parent">
    <TextView
        android:layout_marginTop="20dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:text="File Editor"
        android:textColor="@color/colorTitle"
        android:textSize="30dp"/>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="4.8"
        android:orientation="vertical"
        android:weightSum="5">
        <EditText
            android:layout_width="350dp"
            android:layout_height="wrap_content"
            android:id="@+id/filename"
            android:layout_gravity="center_horizontal"
            />
        <EditText
            android:gravity="top"
            android:layout_width="350dp"
            android:layout_height="wrap_content"
            android:layout_weight="4.9"
```

```
            android:id="@+id/filecotent"
            android:layout_gravity="center_horizontal"
            android:hint="File Content Here"/>
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.2"
        android:orientation="vertical">
        <android.support.constraint.ConstraintLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal">
            <Button
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:id="@+id/save"
                android:text="SAVE"
                app:layout_constraintRight_toLeftOf="@+id/load"
                android:layout_marginRight="20dp"
                />
            <Button
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:id="@+id/load"
                android:text="LOAD"
                app:layout_constraintLeft_toRightOf="@+id/save"
                app:layout_constraintRight_toLeftOf="@+id/clear"
                />
            <Button
                android:layout_width="wrap_content"
                 android:layout_height="wrap_content"
                 android:id="@+id/clear"
                 app:layout_constraintLeft_toRightOf="@+id/load"
                 android:text="CLEAR"
                />
        </android.support.constraint.ConstraintLayout>
        <Button
            android:layout_width="300dp"
            android:layout_height="wrap_content"
            android:id="@+id/delete"
            android:layout_gravity="center_horizontal"
            android:text="DELETE"
            android:textColor="@color/colorAccent"
            android:layout_marginTop="15dp"/>
    </LinearLayout>

</LinearLayout>
```

整体使用线性布局，用 weight 来调整每一部分所占比例。两个 edittext 也使用线性布局。三个 button 使用约束布

局，约束布局和一个大 button 使用线性布局。

【2】 SharedPreferences 的使用

```
SharedPreferences sharedPreferences = getSharedPreferences(PREFERENCE_NAME, MODE);
Boolean user_first = sharedPreferences.getBoolean("FIRST",true);

if (user_first) {
    ok.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            SharedPreferences sharedPreferences1 = getSharedPreferences(PREFERENCE_NAME, MODE);
            SharedPreferences.Editor editor = sharedPreferences1.edit();
            if (password.getText().toString().isEmpty() || con_password.getText().toString().isEmpty()) {
                Toast.makeText(data_saving.this, "Password cannot be empty", Toast.LENGTH_SHORT).show();
            } else if (!password.getText().toString().equals(con_password.getText().toString())) {
                Toast.makeText(data_saving.this, "Password Mismatch", Toast.LENGTH_SHORT).show();
            } else if (!password.getText().toString().isEmpty()
                    && password.getText().toString().equals(con_password.getText().toString())){
                editor.putString("password", con_password.getText().toString());
                editor.putBoolean("FIRST", false);
                editor.commit();
                Intent intent = new Intent(data_saving.this, file_editor.class);
                startActivity(intent);
                //Toast.makeText(data_saving.this, "数据成功写入", Toast.LENGTH_SHORT).show();
            }
        }
    });
    clear.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            password.setText("");
            con_password.setText("");
        }
    });
} else {
```

```
        } else {
            password.setVisibility(View.INVISIBLE);
            con_password.setHint("Password");

            ok.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    SharedPreferences sharedPreferences1 = getSharedPreferences(PREFERENCE_NAME, MODE);
                    String p = sharedPreferences1.getString("password", "Default");
                    if (con_password.getText().toString().isEmpty()) {
                        Toast.makeText(data_saving.this, "Password cannot be empty", Toast.LENGTH_SHORT).show();
                    } else if (!con_password.getText().toString().equals(p)) {
                        Toast.makeText(data_saving.this, "Password Mismatch", Toast.LENGTH_SHORT).show();
                    } else if (!con_password.getText().toString().isEmpty() && con_password.getText().toString().equals(p)) {
                        Intent intent = new Intent(data_saving.this, file_editor.class);
                        startActivity(intent);
                        //Toast.makeText(data_saving.this, "Succeed!", Toast.LENGTH_SHORT).show();
                    }
                }
            });
            clear.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    con_password.setText("");
                }
            });
        }

    }
```

首先用其存储一个 boolean 变量，用于判断是否是第一启动程序。如果是第一启动，则有两个输入框，否则只有一个输入框。使用 sharedpreference.edit()保存数据。注意每次使用 edit 后都要提交。使用 getString 等等读取数据。

【3】 文件存储，加载，删除的实现

```
save.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (filename.getText().toString().isEmpty()) {
            Toast.makeText(file_editor.this, "Filename cannot be empty ", Toast.LENGTH_SHORT).show();
        } else {
            FileOutputStream fileOutputStream = null;
            try  {
                fileOutputStream = openFileOutput(filename.getText().toString(), MODE_PRIVATE);
                String str = filecontent.getText().toString();
                fileOutputStream.write(str.getBytes());
                fileOutputStream.flush();
                fileOutputStream.close();
                Log.i("Tag", "Successfully saved file");
                Toast.makeText(file_editor.this, "Save successfully", Toast.LENGTH_SHORT).show();
            } catch (IOException ex) {
                Log.e("TAG", "Fail to save file");
            }
        }
    }
});
```

```
load.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (filename.getText().toString().isEmpty()) {
            Toast.makeText(file_editor.this, "File name cannot be empty", Toast.LENGTH_SHORT).show();
        } else {
            filecontent.setText("");
            FileInputStream fileInputStream = null;
            try {
                fileInputStream = openFileInput(filename.getText().toString());
                byte[] contents = new byte[fileInputStream.available()];
                fileInputStream.read(contents);
                String str = new String(contents);
                filecontent.setText(str);
                Toast.makeText(file_editor.this, "Load successfully", Toast.LENGTH_SHORT).show();

            } catch (IOException ex) {
                Toast.makeText(file_editor.this, "Fail to load file", Toast.LENGTH_SHORT).show();
                Log.e("TAG", "Fail to read file.");
            }
        }
    }
});
```

```
clear.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        filename.setText("");
        filecontent.setText("");
    }
});

delete.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (filename.getText().toString().isEmpty()) {
            Toast.makeText(file_editor.this, "File name cannot be empty", Toast.LENGTH_SHORT).show();
        }else {
            deleteFile(filename.getText().toString());
            Toast.makeText(file_editor.this, "Delete successfully", Toast.LENGTH_SHORT).show();
        }
    }
});
```

openFileOutput()函数用于写入数据，如果指定的文件不存在，则创建一 个新的文件。为了提高文件系统的性能，一般调用 write()函数时，如果写入的数据量较小，系统会把数据保存在数据缓冲区中，等数据量累积到一定程度时再一次性的删除。写入文件中在调用 close()函数关闭文件前，务必要 调用 flush()函数，将缓冲区内所有的数据写入文件。openFileInput()函数用于打开一个与应用程序联系的私有文件输入流，当文件不存在时抛出 FileNotFoundException 异常。上面的两部分代码在实际使用过程中会遇到错误提 示，因为文件操作可能会遇到各种问题而最终导致操作失败，因此代码应该使用 try/catch 捕获可能产 生的异常。

（3） 实验遇到困难以及解决思路

1. 不能够成功保存 sharedPreference

解决方法：

```
m_iSharedPre = getSharedPreferences("MyTest", MODE_PRIVATE);
如果用以下方式写入数据：
        m_iSharedPre.edit().putInt("aa", 100);
        m_iSharedPre.edit().commit();
那么当获取数据的时候
        m_iSharedPre.getInt("aa", 0);
永远返回默认值。
这是为什么呢？？来查看一下edit()函数的说明：
/**
 * Create a new Editor for these preferences, through which you can make
 * modifications to the data in the preferences and atomically commit those
 * changes back to the SharedPreferences object.
 *
 * <p>Note that you <em>must</em> call {@link Editor#commit} to have any
 * changes you perform in the Editor actually show up in the
 * SharedPreferences.
 *
 * @return Returns a new instance of the {@link Editor} interface, allowing
 * you to modify the values in this SharedPreferences object.
 */
Editor edit();
光看第一句就明白了，原来每次调用edit()函数都是创建一个新的Editor对象，真是坑啊！
正确的写法：
        m_iSharedPre = getSharedPreferences("MyTest", MODE_PRIVATE);
        m_iSharedEditor = m_iSharedPre.edit();
        m_iSharedEditor .putInt("aa", 100);
        m_iSharedEditor .commit();
那么当下次再开启程序的时候
        m_iSharedPre.getInt("aa", 0);
返回值就是100.
```

上网查资料得知我每次调用 edit（）函数都是创建一个新的 Editor 对象，所以应该首先声明 edit = sharepreference.edit()。

2. 如何判断是否为第一次启动程序。

   解决方法：使用 sharedpreference 存储一个 boolean 变量，第一次时设为 true，当保存了用户的密码信息后设为 false。每次 oncreate 时判断这个变量的值，如果是 true，则是第一次启动程序，否则为第二次启动程序。

```
SharedPreferences sharedPreferences = getSharedPreferences(PREFERENCE_NAME, MODE);
Boolean user_first = sharedPreferences.getBoolean("FIRST",true);

if (user_first) {
```

3. 如何删除文件：

   解决方法：上网查询，删除内部存储的文件只需 deletefile（），括号内为 string 类型的文件名。

四、 实验思考及感想

1. 这次实验总体来不是特别难，代码量少，老师的 PPT,和 TA 的实验文档十分详尽，理解起来也比较容易。这次实验主要涉及了两个部分，一个是 sharedpreference，一个是安卓文件存储。
2. SharedPreference 是 Android 提供的一种轻量级的数据存储方式，主要用来存储一些简单的配置信息，例如，默认欢迎语，登录用户名和密码等。其以键值对的方式存储,使得我们能很方便进行读取和存入。
3. Android 使用的是基于 Linux 的文件系统。 程序开发人员可以建立和访问程序自身的私有文件，也可以访问保存在资源（res）目录中的原始文件和 XML 文 件，还可以在 SD 卡等外部存储设备中保存与读取文件。这次实验涉及的是建立和访问程序自身的私有文件。
4. 当 Activity 不可见时，如何将其从 activity stack 中除去（按返回键直接返回 Home）。
   在 AndroidManifest.xml 中设置 noHistory 属性。

```
<activity android:name=".data_saving"  android:noHistory="true">
```

5. 更深刻的学会使用 LinearLayout，学会了使用 layout_weight 属性和 weightsum 属性。
   android:weightSum 属性在官方文档中的解释包含下面的内容：“定义 weight 综合的最大值，如果未指定该值，以所有子师徒的 layout_weight 属性的累加值作为总和的最大值。典型案例是：通过指定子视图的 layout_weight 属性为 0.5.并设置 LinearLayout 的 weightSum 属性为 1.0，实现子视图占据可用宽高的 50%”。
   LinearLayout 中子控件可以使用 layout_weight 属性来分配所占的“权重”。
6. 如何根据需要隐藏/显示特定的控件？

```
password.setVisibility(View.INVISIBLE);
```

   设置控件的 visibility 属性。
7. Internal Storage 和 External Storage 的区别以及他们分别适用的场景。
   Internal storage 是属于应用程序的，文件管理器看不见。External storage 在文件浏览器里是可以看见的 。这两个概念都是相对于应用来说的，应该理解为逻辑上的概念,不应理解为物理上的外部 SD 卡和手机或移动设备内存.。一个应用把数据存在 external storage 上时,那么数据成为共有的,所有人都可见的和可用的。存在 internal storage 上时,只有这个应用本身可以看到和使用。很多没有插 S D 卡的设备,系统会虚拟出一部分存储空间用来做公共存储(主要是音乐,文档之类的 media)。
   内部存储：总是可用的
   这里的文件默认只能被我们的 app 所访问。
   当用户卸载 app 的时候，系统会把 internal 内该 app 相关的文件都清除干净。
   Internal 是我们在想确保不被用户与其他 app 所访问的最佳存储区域。
   外部存储：并不总是可用的，因为用户有时会通过 USB 存储模式挂载外部存储器，当取下挂载的这部分后，就无法对其进行访问了。
   是大家都可以访问的，因此保存在这里的文件可能被其他程序访问。
   当用户卸载我们的 app 时，系统仅仅会删除 external 根目录（getExternalFilesDir()）下的相关文件。

External 是在不需要严格的访问权限并且希望这些文件能够被其他 app 所共享或者是允许用户通过电脑访问时的最佳存储区域。

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。