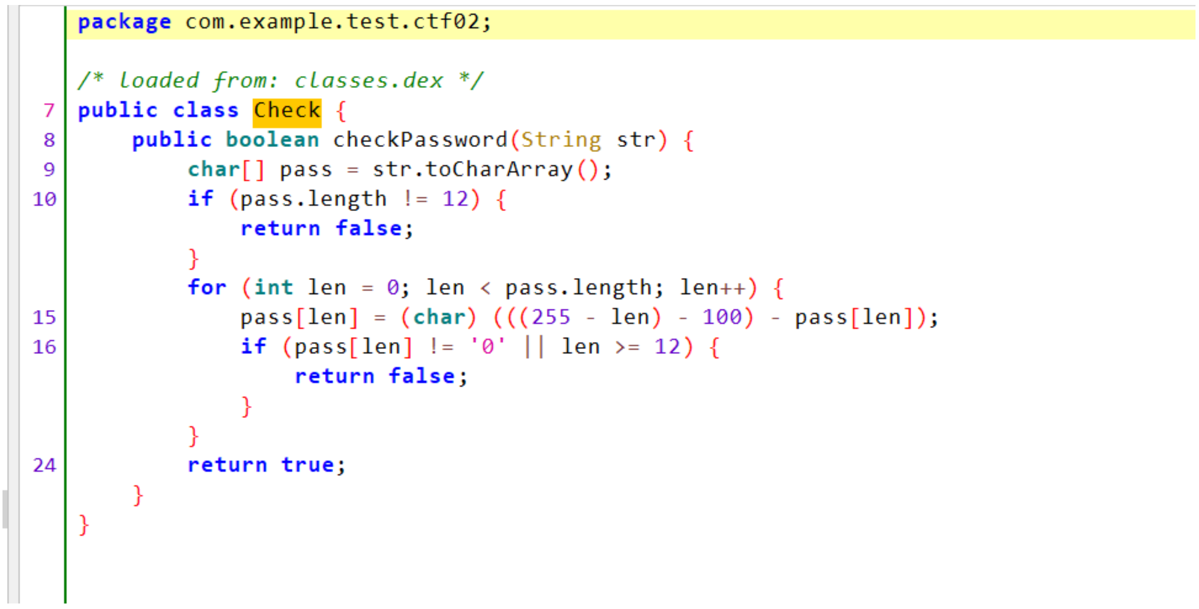
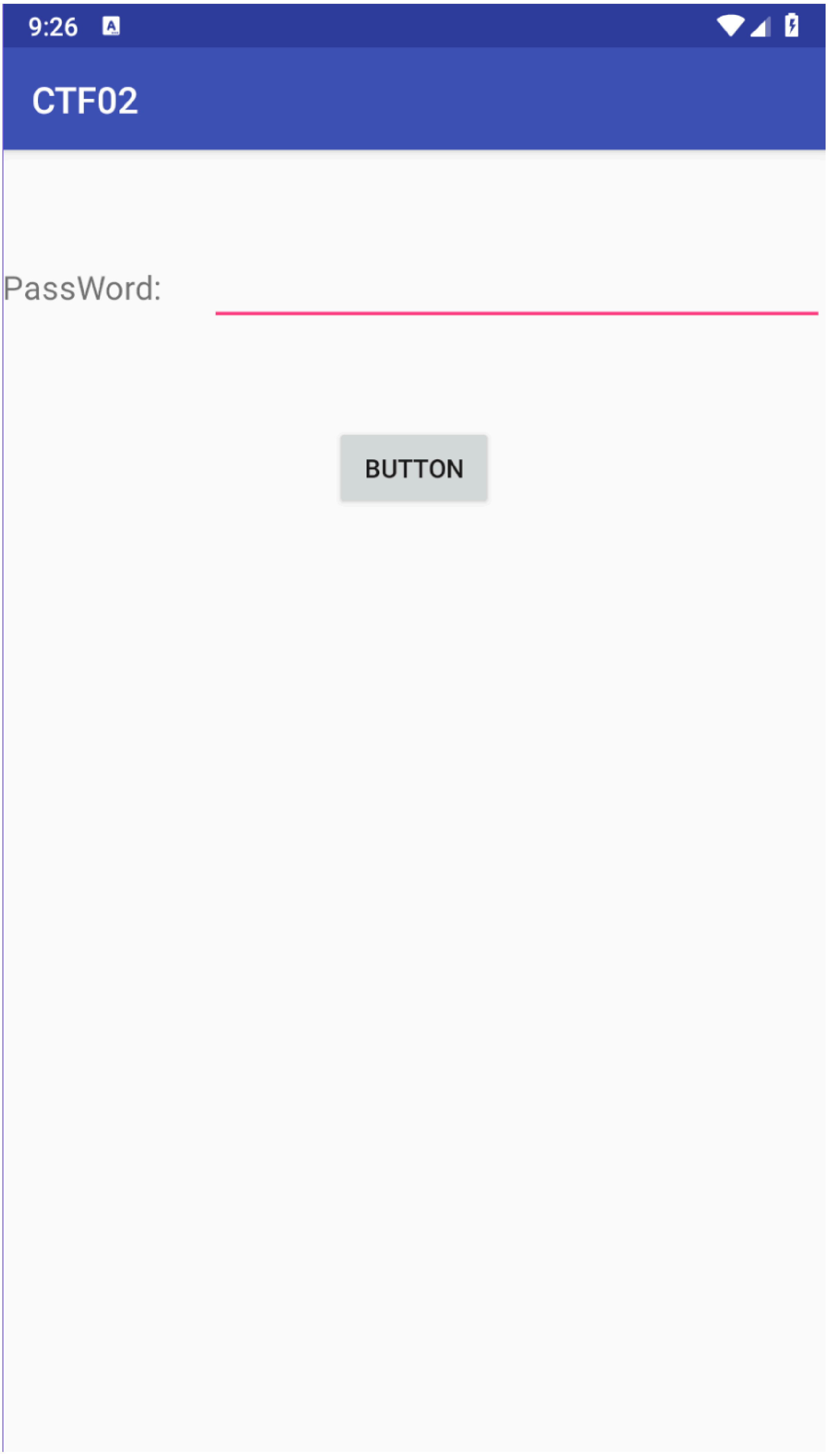
polarctf RE刷题记录（困难的安卓）

test



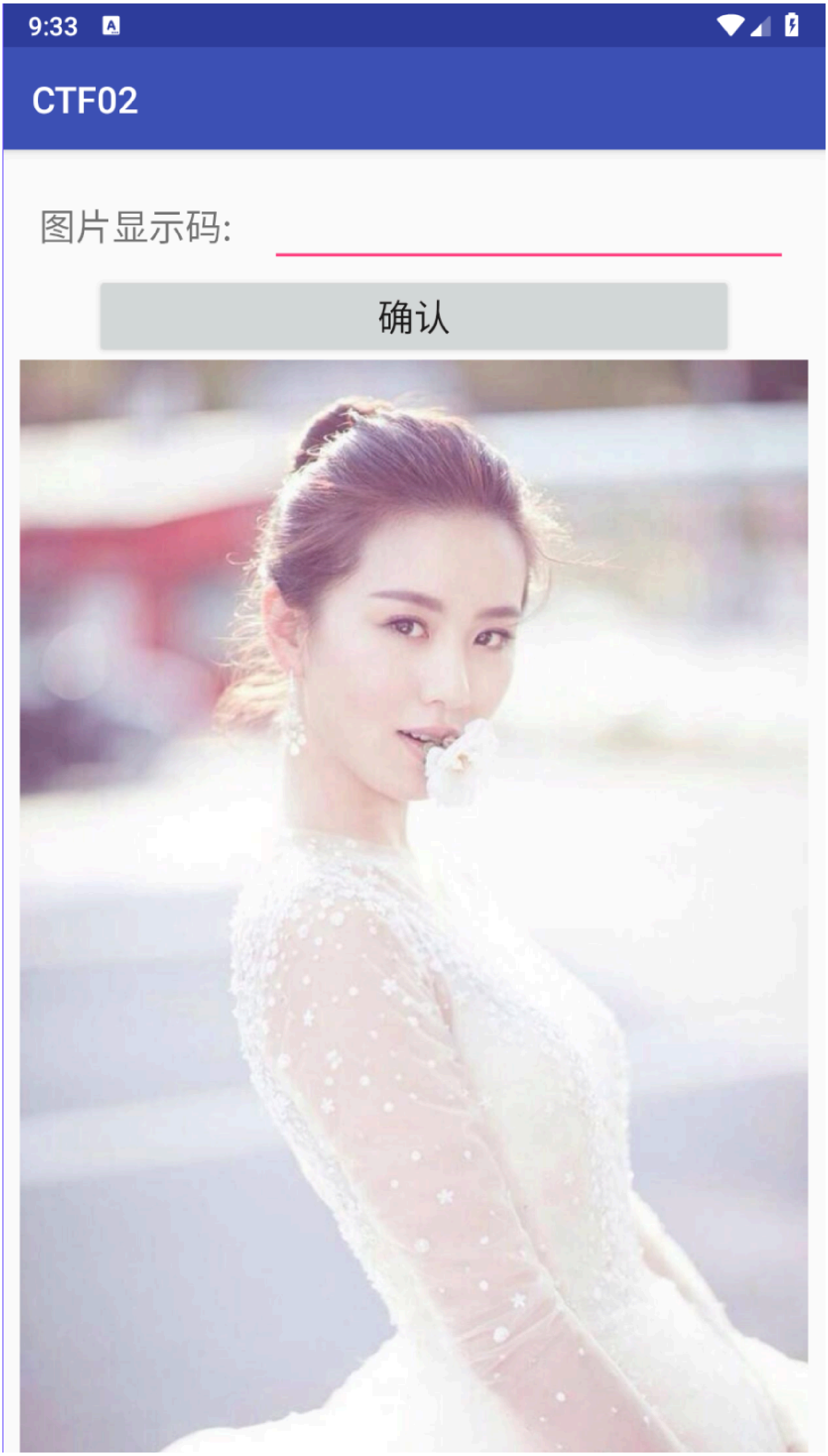
```java
package com.example.test.ctf02;

/* loaded from: classes.dex */
public class Check {
    public boolean checkPassword(String str) {
        char[] pass = str.toCharArray();
        if (pass.length != 12) {
            return false;
        }
        for (int len = 0; len < pass.length; len++) {
            pass[len] = (char) (((255 - len) - 100) - pass[len]);
            if (pass[len] != '0' || len >= 12) {
                return false;
            }
        }
        return true;
    }
}
```

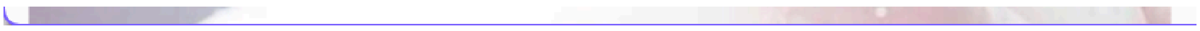

9:26
CTF02
PassWord:
BUTTON

使用jadx打开，可分析出password逻辑为一个12位的字符串，让他在经过

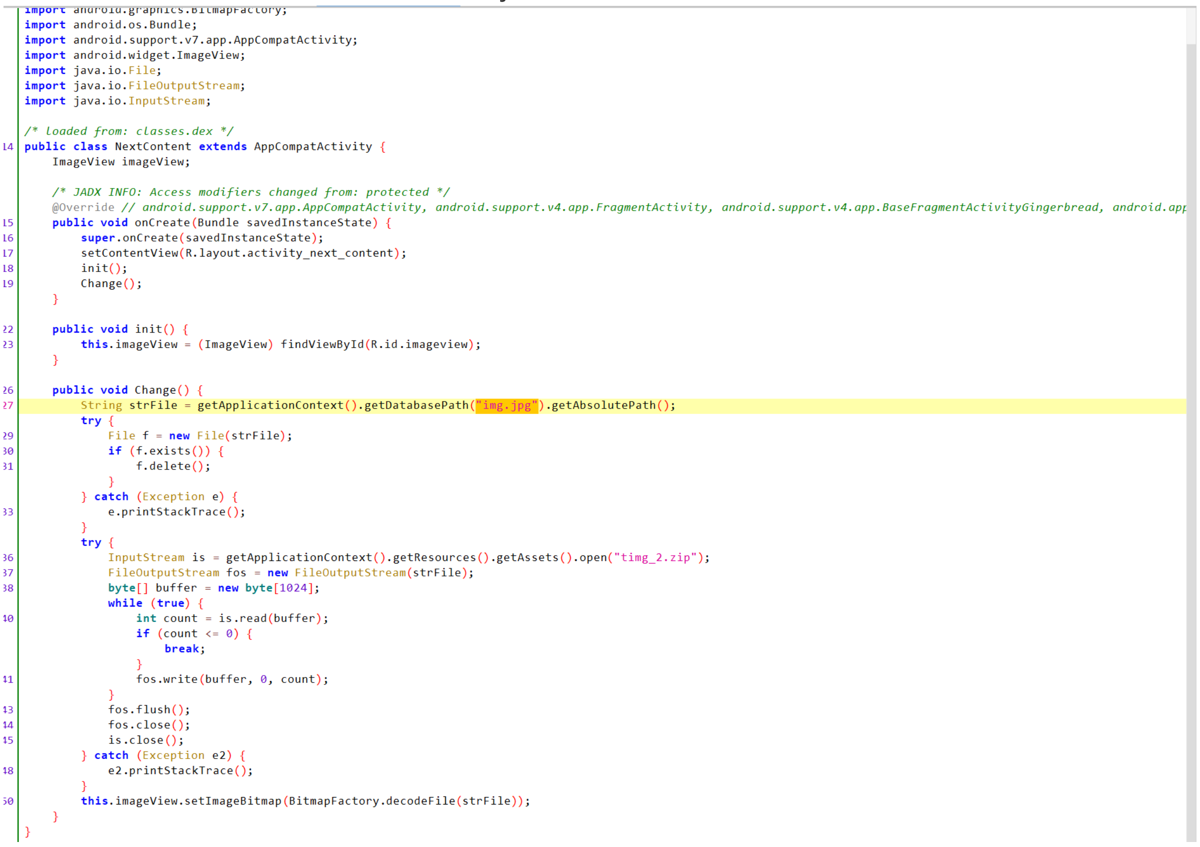
```
for (int len = 0; len < pass.length; len++) {
    pass[len] = (char) (((255 - len) - 100) - pass[len]);
```

后每一位都等于'0'（就是ascII码的48），让ai推，或者写代码就可以推出字符串为kjihgfedcba`，输入


9:33
CTF02
图片显示码:
确认



嗯？图片显示码？难道还要misc？于是又去jadx里，

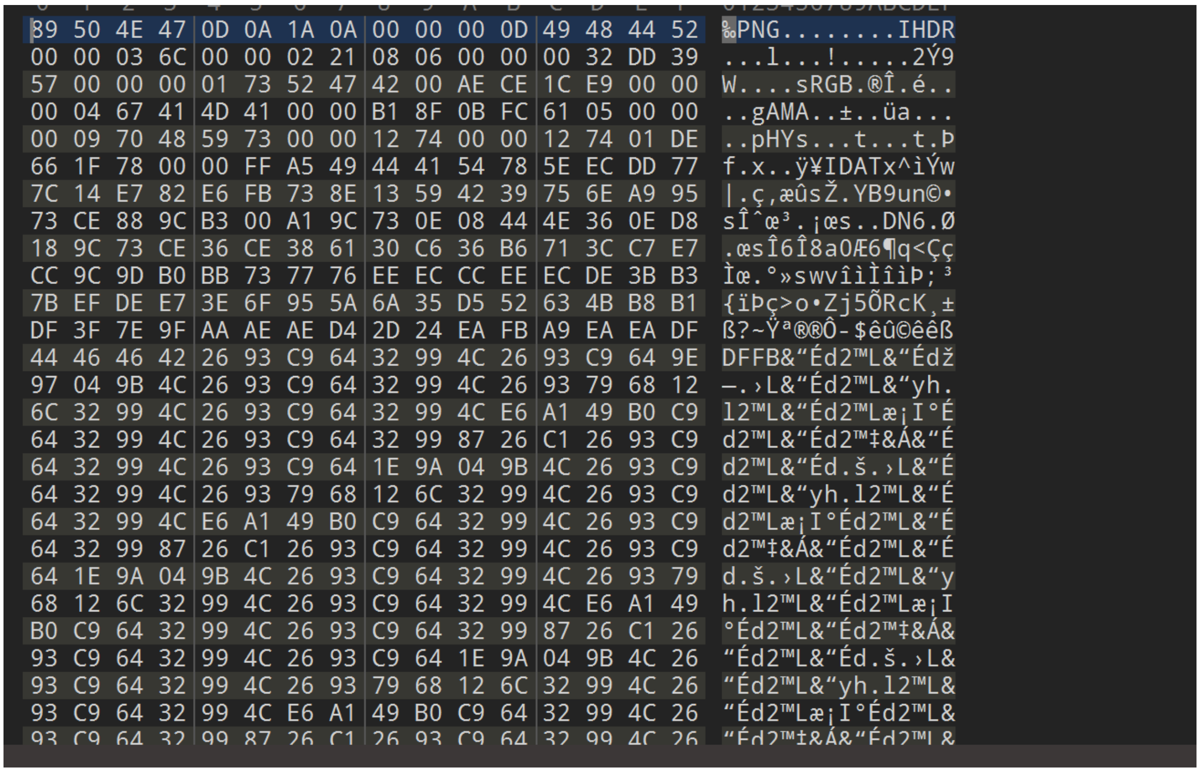


```
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.widget.ImageView;
import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;

/* loaded from: classes.dex */
public class NextContent extends AppCompatActivity {
    ImageView imageView;

    /* JADX INFO: Access modifiers changed from: protected */
    @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, android.support.v4.app.BaseFragmentActivityGingerbread, android.app
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_next_content);
        init();
        Change();
    }

    public void init() {
        this.imageView = (ImageView) findViewById(R.id.imageview);
    }

    public void Change() {
        String strFile = getApplicationContext().getDatabasePath("img.jpg").getAbsolutePath();
        try {
            File f = new File(strFile);
            if (f.exists()) {
                f.delete();
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        try {
            InputStream is = getApplicationContext().getResources().getAssets().open("timg_2.zip");
            FileOutputStream fos = new FileOutputStream(strFile);
            byte[] buffer = new byte[1024];
            while (true) {
                int count = is.read(buffer);
                if (count <= 0) {
                    break;
                }
                fos.write(buffer, 0, count);
            }
            fos.flush();
            fos.close();
            is.close();
        } catch (Exception e2) {
            e2.printStackTrace();
        }
        this.imageView.setImageBitmap(BitmapFactory.decodeFile(strFile));
    }
}
```

逻辑为从assets中读取timg_2.zip，将其内容写入img.jpg，3最后通过ImageView显示该文件，嗯，总算可以解包了，在apk路径命令行中运行

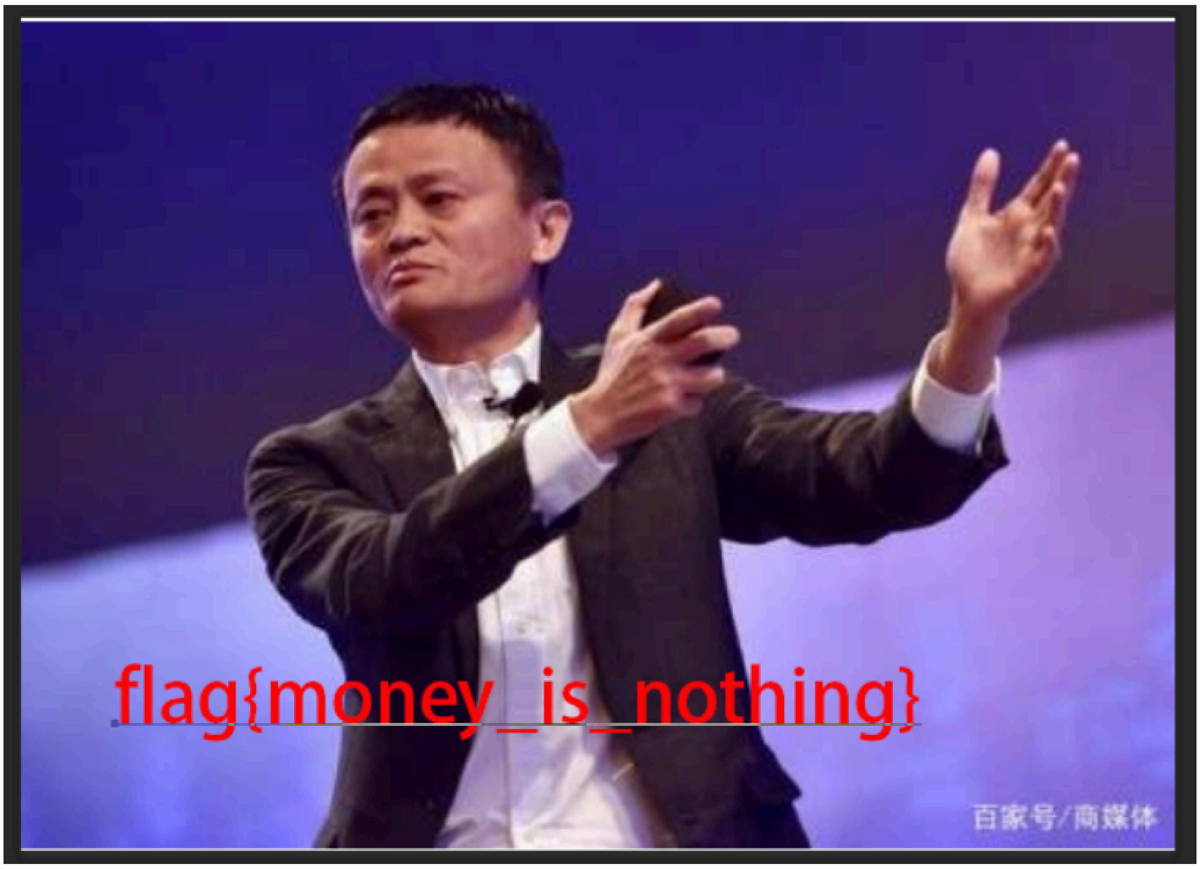
```
apktool d test.apk
```

生成的文件夹中assets里面的timg压缩包打不开，最开始我以为是要用其他方式比如7zip里的#之类的提取方式（gal玩多了就是这样），然后想着既然是生成了图片，那会不会是要改后缀，于是拖入010，



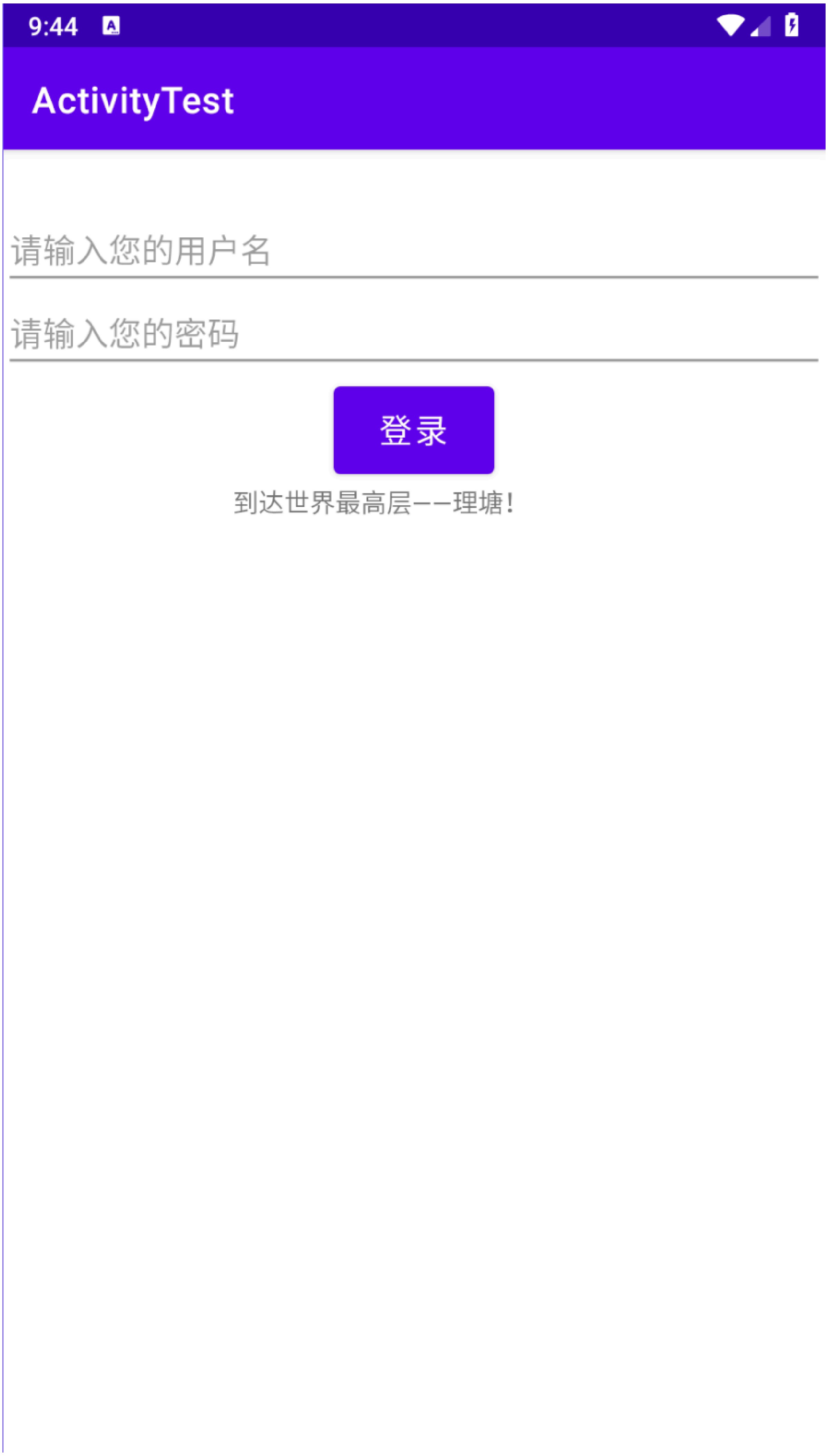
```
89 50 4E 47 0D 0A 1A 0A 00 00 00 0D 49 48 44 52  ‰PNG........IHDR
00 00 03 6C 00 00 02 21 08 06 00 00 00 32 DD 39  ...l...!.....2Ý9
57 00 00 00 01 73 52 47 42 00 AE CE 1C E9 00 00  W....sRGB.®Î.é..
00 04 67 41 4D 41 00 00 B1 8F 0B FC 61 05 00 00  ..gAMA..±..üa...
00 09 70 48 59 73 00 00 12 74 00 00 12 74 01 DE  ..pHYs...t...t.Þ
66 1F 78 00 00 FF A5 49 44 41 54 78 5E EC DD 77  f.x..ÿ¥IDATx^ìÝw
7C 14 E7 82 E6 FB 73 8E 13 59 42 39 75 6E A9 95  |.ç‚æûsŽ.YB9un©•
73 CE 88 9C B3 00 A1 9C 73 0E 08 44 4E 36 0E D8  sÎˆœ³.¡œs..DN6.Ø
18 9C 73 CE 36 CE 38 61 30 C6 36 B6 71 3C C7 E7  .œsÎ6Î8a0Æ6¶q<Çç
CC 9C 9D B0 BB 73 77 76 EE EC CC EE EC DE 3B B3  Ìœ.°»swvîìÌîìÞ;³
7B EF DE E7 3E 6F 95 5A 6A 35 D5 52 63 4B B8 B1  {ïÞç>o•Zj5ÕRcK¸±
DF 3F 7E 9F AA AE AE D4 2D 24 EA FB A9 EA EA DF  ß?~Ÿª®®Ô-$êû©êêß
44 46 46 42 26 93 C9 64 32 99 4C 26 93 C9 64 9E  DFFB&“Éd2™L&“Édž
97 04 9B 4C 26 93 C9 64 32 99 4C 26 93 79 68 12  —.›L&“Éd2™L&“yh.
6C 32 99 4C 26 93 C9 64 32 99 4C E6 A1 49 B0 C9  l2™L&“Éd2™Læ¡I°É
64 32 99 4C 26 93 C9 64 32 99 87 26 C1 26 93 C9  d2™L&“Éd2™‡&Á&“É
64 32 99 4C 26 93 C9 64 1E 9A 04 9B 4C 26 93 C9  d2™L&“Éd.š.›L&“É
64 32 99 4C 26 93 79 68 12 6C 32 99 4C 26 93 C9  d2™L&“yh.l2™L&“É
64 32 99 4C E6 A1 49 B0 C9 64 32 99 4C 26 93 C9  d2™Læ¡I°Éd2™L&“É
64 32 99 87 26 C1 26 93 C9 64 32 99 4C 26 93 C9  d2™‡&Á&“Éd2™L&“É
64 1E 9A 04 9B 4C 26 93 C9 64 32 99 4C 26 93 79  d.š.›L&“Éd2™L&“y
68 12 6C 32 99 4C 26 93 C9 64 32 99 4C E6 A1 49  h.l2™L&“Éd2™Læ¡I
B0 C9 64 32 99 4C 26 93 C9 64 32 99 87 26 C1 26  °Éd2™L&“Éd2™‡&Á&
93 C9 64 32 99 4C 26 93 C9 64 1E 9A 04 9B 4C 26  “Éd2™L&“Éd.š.›L&
93 C9 64 32 99 4C 26 93 79 68 12 6C 32 99 4C 26  “Éd2™L&“yh.l2™L&
93 C9 64 32 99 4C E6 A1 49 B0 C9 64 32 99 4C 26  “Éd2™Læ¡I°Éd2™L&
93 C9 64 32 99 87 26 C1 26 93 C9 64 32 99 4C 26  “Éd2™‡&Á&“Éd2™L&
```

将zip后缀改为png，得到



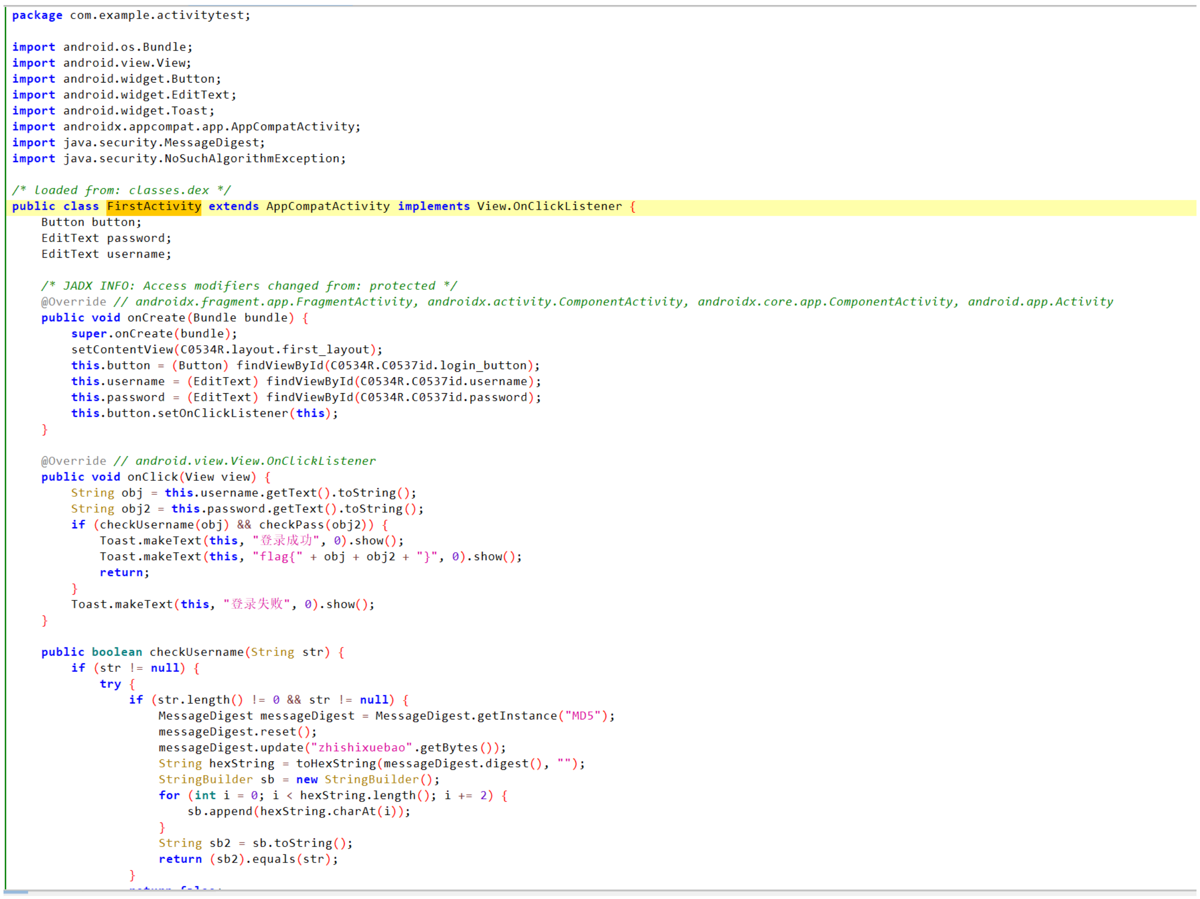
这个flag挺有意思

app-login

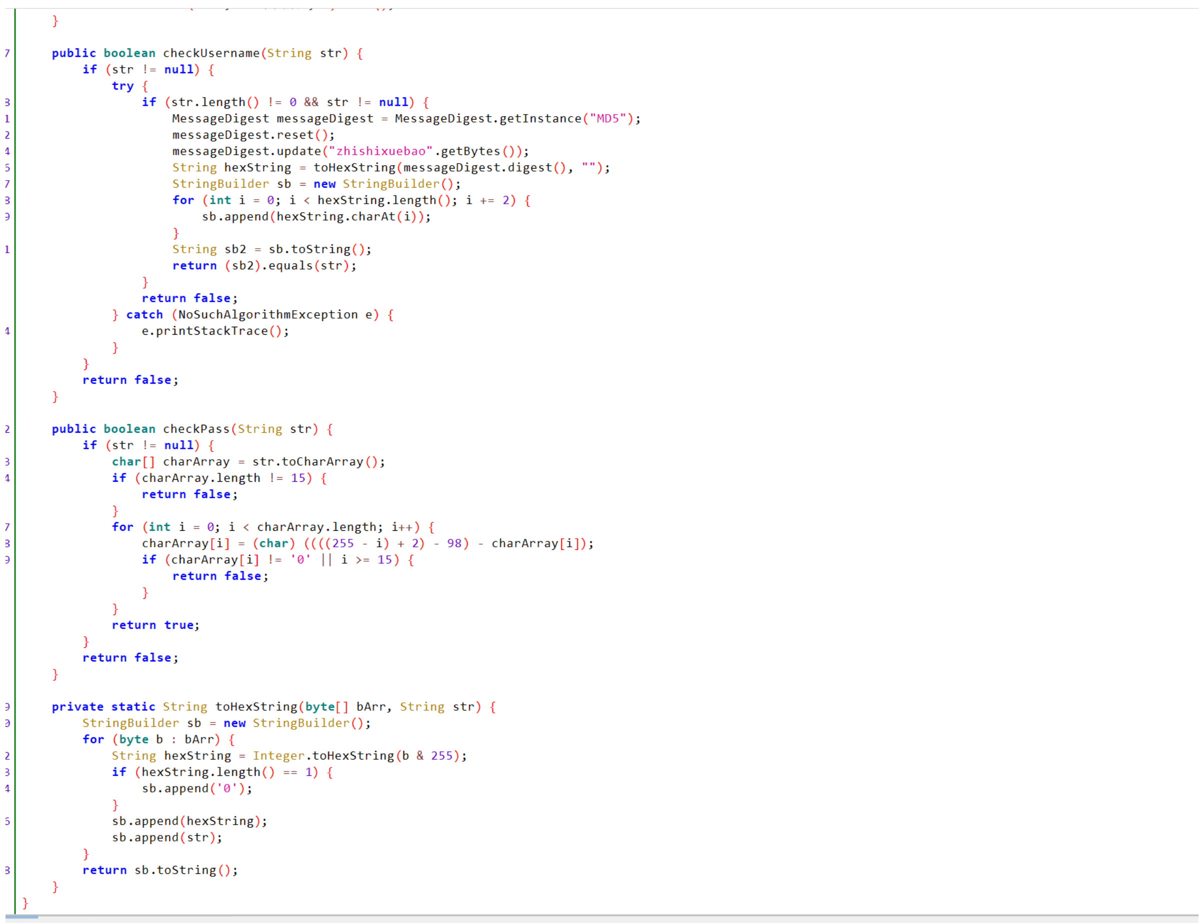


到达世界最高层——理塘！

依旧典型的login，一眼顶针要用jadx



```java
package com.example.activitytest;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

/* loaded from: classes.dex */
public class FirstActivity extends AppCompatActivity implements View.OnClickListener {
    Button button;
    EditText password;
    EditText username;

    /* JADX INFO: Access modifiers changed from: protected */
    @Override // androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, androidx.core.app.ComponentActivity, android.app.Activity
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(C0534R.layout.first_layout);
        this.button = (Button) findViewById(C0534R.C0537id.login_button);
        this.username = (EditText) findViewById(C0534R.C0537id.username);
        this.password = (EditText) findViewById(C0534R.C0537id.password);
        this.button.setOnClickListener(this);
    }

    @Override // android.view.View.OnClickListener
    public void onClick(View view) {
        String obj = this.username.getText().toString();
        String obj2 = this.password.getText().toString();
        if (checkUsername(obj) && checkPass(obj2)) {
            Toast.makeText(this, "登录成功", 0).show();
            Toast.makeText(this, "flag{" + obj + obj2 + "}", 0).show();
            return;
        }
        Toast.makeText(this, "登录失败", 0).show();
    }

    public boolean checkUsername(String str) {
        if (str != null) {
            try {
                if (str.length() != 0 && str != null) {
                    MessageDigest messageDigest = MessageDigest.getInstance("MD5");
                    messageDigest.reset();
                    messageDigest.update("zhishixuebao".getBytes());
                    String hexString = toHexString(messageDigest.digest(), "");
                    StringBuilder sb = new StringBuilder();
                    for (int i = 0; i < hexString.length(); i += 2) {
                        sb.append(hexString.charAt(i));
                    }
                    String sb2 = sb.toString();
                    return (sb2).equals(str);
                }
```



```java
    }

    public boolean checkUsername(String str) {
        if (str != null) {
            try {
                if (str.length() != 0 && str != null) {
                    MessageDigest messageDigest = MessageDigest.getInstance("MD5");
                    messageDigest.reset();
                    messageDigest.update("zhishixuebao".getBytes());
                    String hexString = toHexString(messageDigest.digest(), "");
                    StringBuilder sb = new StringBuilder();
                    for (int i = 0; i < hexString.length(); i += 2) {
                        sb.append(hexString.charAt(i));
                    }
                    String sb2 = sb.toString();
                    return (sb2).equals(str);
                }
                return false;
            } catch (NoSuchAlgorithmException e) {
                e.printStackTrace();
            }
        }
        return false;
    }

    public boolean checkPass(String str) {
        if (str != null) {
            char[] charArray = str.toCharArray();
            if (charArray.length != 15) {
                return false;
            }
            for (int i = 0; i < charArray.length; i++) {
                charArray[i] = (char) ((((255 - i) + 2) - 98) - charArray[i]);
                if (charArray[i] != '0' || i >= 15) {
                    return false;
                }
            }
            return true;
        }
        return false;
    }

    private static String toHexString(byte[] bArr, String str) {
        StringBuilder sb = new StringBuilder();
        for (byte b : bArr) {
            String hexString = Integer.toHexString(b & 255);
            if (hexString.length() == 1) {
                sb.append('0');
            }
            sb.append(hexString);
            sb.append(str);
        }
        return sb.toString();
    }
}
```

大概逻辑是

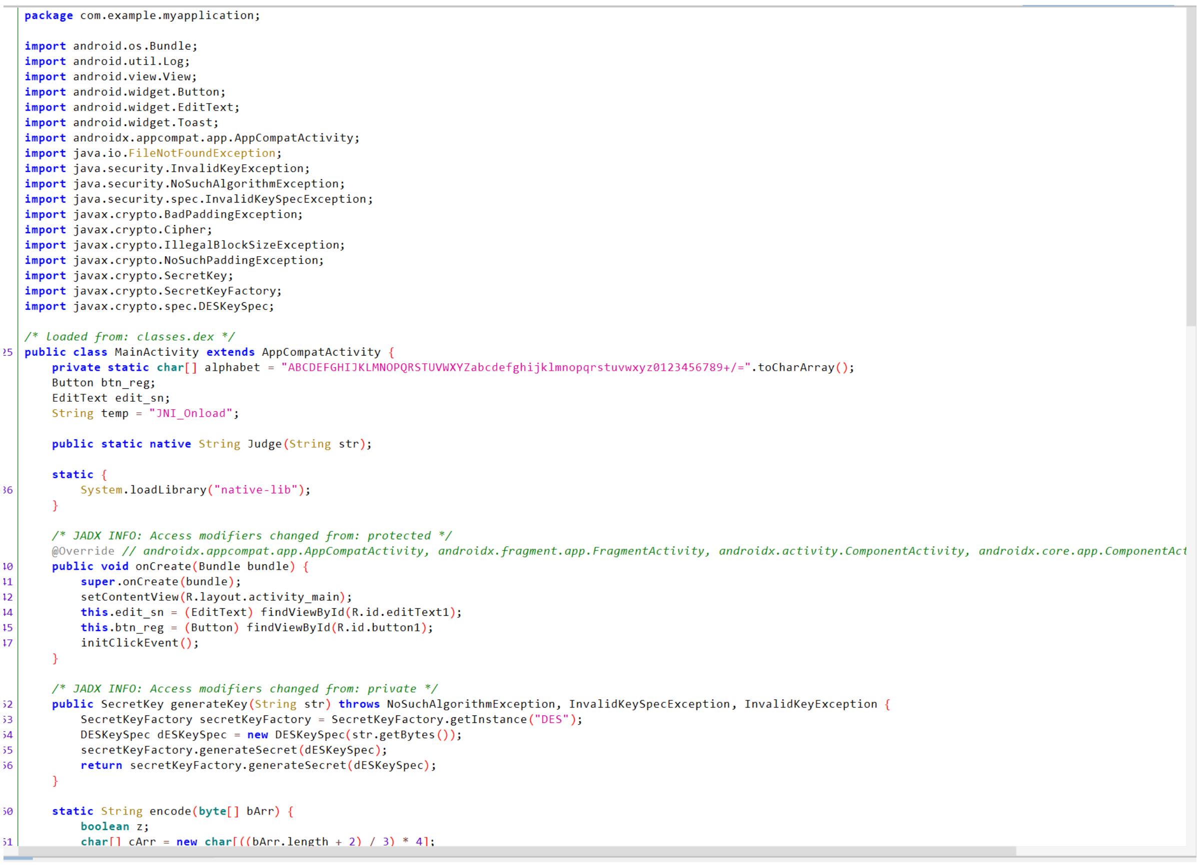
```
解密原理
用户名：
对字符串 "zhishixuebao"计算 MD5 哈希值。
将 MD5 的字节数组转换为十六进制字符串。
取该十六进制字符串的奇数位字符（索引从 0 开始，取偶数索引字符，即第1、3、5...位）。
得到的字符串即为有效用户名。
密码：
密码长度必须为 15 个字符。
每个字符 c_i（i从 0 到 14）需满足变换后等于字符 '0'（ASCII 值 48）。
变换公式为：c_i = (char)(111 - i)，对应 ASCII 字符从 'o'递减到 'a'。
因此密码为字符串 "onmlkjihgfedcba"。
Flag：
登录成功后显示的 Flag 格式为 flag{用户名密码}，即拼接用户名和密码。
```

ai给个脚本直接出

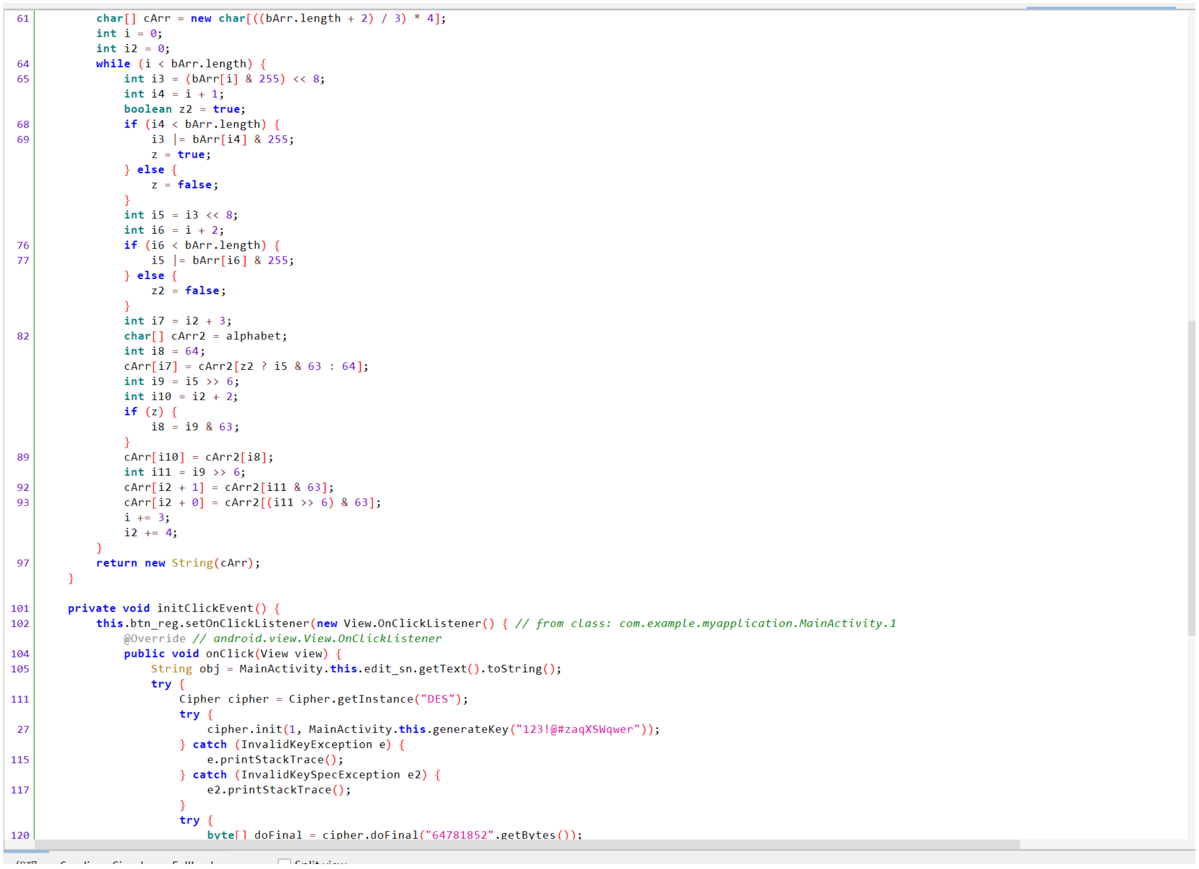
```python
import hashlib

# 计算用户名
def get_username():
    input_str = "zhishixuebao"
    md5_hash = hashlib.md5(input_str.encode()).hexdigest()  # 计算 MD5
    username = md5_hash[::2]  # 取奇数位字符（偶数索引）
    return username

# 计算密码
def get_password():
    password_chars = [chr(111 - i) for i in range(15)]  # 生成字符列表
    password = ''.join(password_chars)  # 拼接成字符串
    return password

# 主程序
if __name__ == "__main__":
    username = get_username()
    password = get_password()
    flag = f"flag{{{username}{password}}}"

    print(f"用户名: {username}")
    print(f"密码: {password}")
    print(f"Flag: {flag}")
```

Android

jadx打开

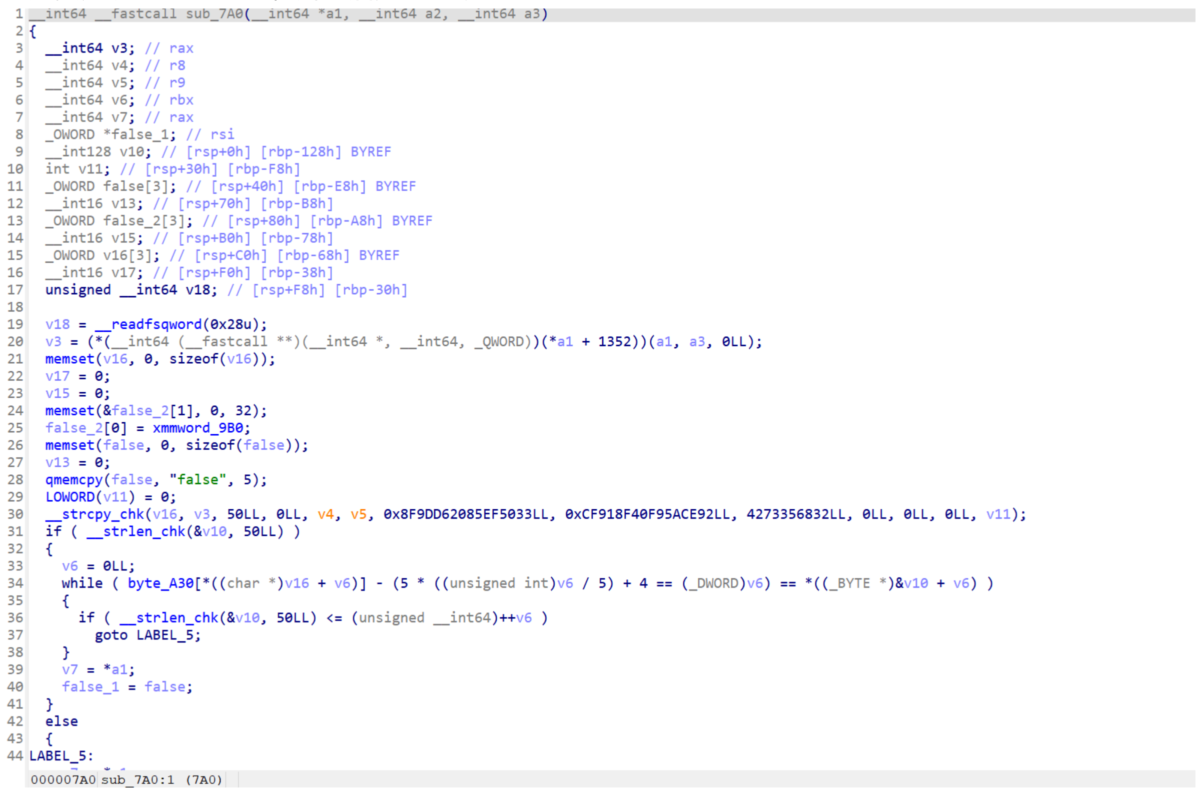


```java
package com.example.myapplication;

import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import java.io.FileNotFoundException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.security.spec.InvalidKeySpecException;
import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.DESKeySpec;

/* loaded from: classes.dex */
public class MainActivity extends AppCompatActivity {
    private static char[] alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=".toCharArray();
    Button btn_reg;
    EditText edit_sn;
    String temp = "JNI_Onload";

    public static native String Judge(String str);

    static {
        System.loadLibrary("native-lib");
    }

    /* JADX INFO: Access modifiers changed from: protected */
    @Override // androidx.appcompat.app.AppCompatActivity, androidx.fragment.app.FragmentActivity, androidx.activity.ComponentActivity, androidx.core.app.ComponentAct
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);
        this.edit_sn = (EditText) findViewById(R.id.editText1);
        this.btn_reg = (Button) findViewById(R.id.button1);
        initClickEvent();
    }

    /* JADX INFO: Access modifiers changed from: private */
    public SecretKey generateKey(String str) throws NoSuchAlgorithmException, InvalidKeySpecException, InvalidKeyException {
        SecretKeyFactory secretKeyFactory = SecretKeyFactory.getInstance("DES");
        DESKeySpec dESKeySpec = new DESKeySpec(str.getBytes());
        secretKeyFactory.generateSecret(dESKeySpec);
        return secretKeyFactory.generateSecret(dESKeySpec);
    }

    static String encode(byte[] bArr) {
        boolean z;
        char[] cArr = new char[((bArr.length + 2) / 3) * 4];
```

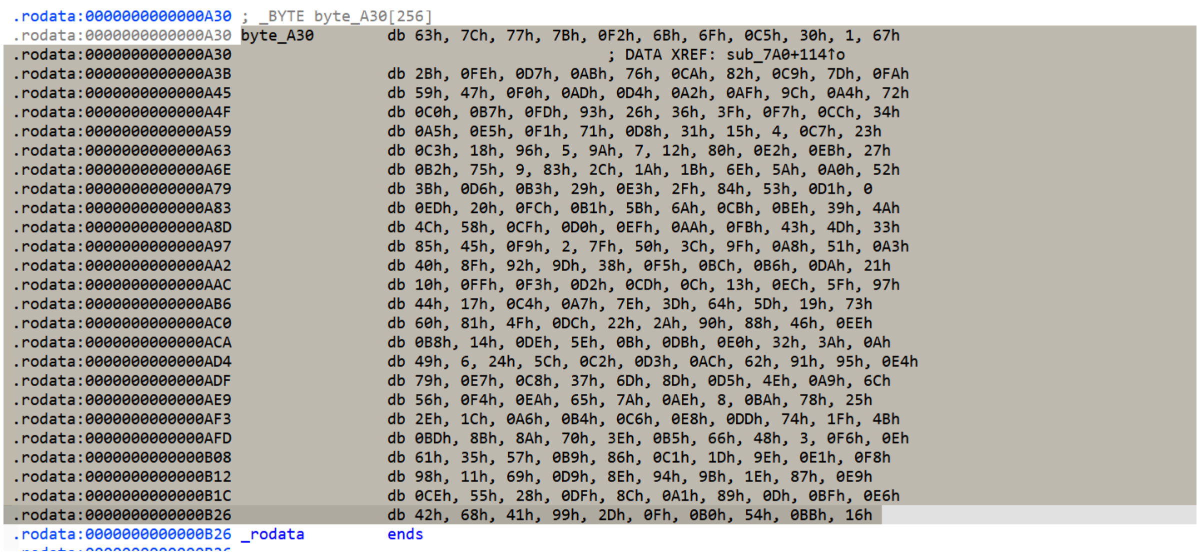


```java
        char[] cArr = new char[((bArr.length + 2) / 3) * 4];
        int i = 0;
        int i2 = 0;
        while (i < bArr.length) {
            int i3 = (bArr[i] & 255) << 8;
            int i4 = i + 1;
            boolean z2 = true;
            if (i4 < bArr.length) {
                i3 |= bArr[i4] & 255;
                z = true;
            } else {
                z = false;
            }
            int i5 = i3 << 8;
            int i6 = i + 2;
            if (i6 < bArr.length) {
                i5 |= bArr[i6] & 255;
            } else {
                z2 = false;
            }
            int i7 = i2 + 3;
            char[] cArr2 = alphabet;
            int i8 = 64;
            cArr[i7] = cArr2[z2 ? i5 & 63 : 64];
            int i9 = i5 >> 6;
            int i10 = i2 + 2;
            if (z) {
                i8 = i9 & 63;
            }
            cArr[i10] = cArr2[i8];
            int i11 = i9 >> 6;
            cArr[i2 + 1] = cArr2[i11 & 63];
            cArr[i2 + 0] = cArr2[(i11 >> 6) & 63];
            i += 3;
            i2 += 4;
        }
        return new String(cArr);
    }

    private void initClickEvent() {
        this.btn_reg.setOnClickListener(new View.OnClickListener() { // from class: com.example.myapplication.MainActivity.1
            @Override // android.view.View.OnClickListener
            public void onClick(View view) {
                String obj = MainActivity.this.edit_sn.getText().toString();
                try {
                    Cipher cipher = Cipher.getInstance("DES");
                    try {
                        cipher.init(1, MainActivity.this.generateKey("123!@#zaqXSWqwer"));
                    } catch (InvalidKeyException e) {
                        e.printStackTrace();
                    } catch (InvalidKeySpecException e2) {
                        e2.printStackTrace();
                    }
                    try {
                        byte[] doFinal = cipher.doFinal("64781852".getBytes());
```

可知要分析libnative.so，解包后用ida打开



```
__int64 __fastcall sub_7A0(__int64 *a1, __int64 a2, __int64 a3)
{
  __int64 v3; // rax
  __int64 v4; // r8
  __int64 v5; // r9
  __int64 v6; // rbx
  __int64 v7; // rax
  _OWORD *false_1; // rsi
  __int128 v10; // [rsp+0h] [rbp-128h] BYREF
  int v11; // [rsp+30h] [rbp-F8h]
  _OWORD false[3]; // [rsp+40h] [rbp-E8h] BYREF
  __int16 v13; // [rsp+70h] [rbp-B8h]
  _OWORD false_2[3]; // [rsp+80h] [rbp-A8h] BYREF
  __int16 v15; // [rsp+B0h] [rbp-78h]
  _OWORD v16[3]; // [rsp+C0h] [rbp-68h] BYREF
  __int16 v17; // [rsp+F0h] [rbp-38h]
  unsigned __int64 v18; // [rsp+F8h] [rbp-30h]

  v18 = __readfsqword(0x28u);
  v3 = (*(__int64 (__fastcall **)(__int64 *, __int64, _QWORD))(*a1 + 1352))(a1, a3, 0LL);
  memset(v16, 0, sizeof(v16));
  v17 = 0;
  v15 = 0;
  memset(&false_2[1], 0, 32);
  false_2[0] = xmmword_9B0;
  memset(false, 0, sizeof(false));
  v13 = 0;
  qmemcpy(false, "false", 5);
  LOWORD(v11) = 0;
  __strcpy_chk(v16, v3, 50LL, 0LL, v4, v5, 0x8F9DD62085EF5033LL, 0xCF918F40F95ACE92LL, 4273356832LL, 0LL, 0LL, 0LL, v11);
  if ( __strlen_chk(&v10, 50LL) )
  {
    v6 = 0LL;
    while ( byte_A30[*((char *)v16 + v6)] - (5 * ((unsigned int)v6 / 5) + 4 == (_DWORD)v6) == *((_BYTE *)&v10 + v6) )
    {
      if ( __strlen_chk(&v10, 50LL) <= (unsigned __int64)++v6 )
        goto LABEL_5;
    }
    v7 = *a1;
    false_1 = false;
  }
  else
  {
LABEL_5:
```



```
.rodata:0000000000000A30 ; _BYTE byte_A30[256]
.rodata:0000000000000A30 byte_A30        db 63h, 7Ch, 77h, 7Bh, 0F2h, 6Bh, 6Fh, 0C5h, 30h, 1, 67h
.rodata:0000000000000A30                                         ; DATA XREF: sub_7A0+114↑o
.rodata:0000000000000A3B                 db 2Bh, 0FEh, 0D7h, 0ABh, 76h, 0CAh, 82h, 0C9h, 7Dh, 0FAh
.rodata:0000000000000A45                 db 59h, 47h, 0F0h, 0ADh, 0D4h, 0A2h, 0AFh, 9Ch, 0A4h, 72h
.rodata:0000000000000A4F                 db 0C0h, 0B7h, 0FDh, 93h, 26h, 36h, 3Fh, 0F7h, 0CCh, 34h
.rodata:0000000000000A59                 db 0A5h, 0E5h, 0F1h, 71h, 0D8h, 31h, 15h, 4, 0C7h, 23h
.rodata:0000000000000A63                 db 0C3h, 18h, 96h, 5, 9Ah, 7, 12h, 80h, 0E2h, 0EBh, 27h
.rodata:0000000000000A6E                 db 0B2h, 75h, 9, 83h, 2Ch, 1Ah, 1Bh, 6Eh, 5Ah, 0A0h, 52h
.rodata:0000000000000A79                 db 3Bh, 0D6h, 0B3h, 29h, 0E3h, 2Fh, 84h, 53h, 0D1h, 0
.rodata:0000000000000A83                 db 0EDh, 20h, 0FCh, 0B1h, 5Bh, 6Ah, 0CBh, 0BEh, 39h, 4Ah
.rodata:0000000000000A8D                 db 4Ch, 58h, 0CFh, 0D0h, 0EFh, 0AAh, 0FBh, 43h, 4Dh, 33h
.rodata:0000000000000A97                 db 85h, 45h, 0F9h, 2, 7Fh, 50h, 3Ch, 9Fh, 0A8h, 51h, 0A3h
.rodata:0000000000000AA2                 db 40h, 8Fh, 92h, 9Dh, 38h, 0F5h, 0BCh, 0B6h, 0DAh, 21h
.rodata:0000000000000AAC                 db 10h, 0FFh, 0F3h, 0D2h, 0CDh, 0Ch, 13h, 0ECh, 5Fh, 97h
.rodata:0000000000000AB6                 db 44h, 17h, 0C4h, 0A7h, 7Eh, 3Dh, 64h, 5Dh, 19h, 73h
.rodata:0000000000000AC0                 db 60h, 81h, 4Fh, 0DCh, 22h, 2Ah, 90h, 88h, 46h, 0EEh
.rodata:0000000000000ACA                 db 0B8h, 14h, 0DEh, 5Eh, 0Bh, 0DBh, 0E0h, 32h, 3Ah, 0Ah
.rodata:0000000000000AD4                 db 49h, 6, 24h, 5Ch, 0C2h, 0D3h, 0ACh, 62h, 91h, 95h, 0E4h
.rodata:0000000000000ADF                 db 79h, 0E7h, 0C8h, 37h, 6Dh, 8Dh, 0D5h, 4Eh, 0A9h, 6Ch
.rodata:0000000000000AE9                 db 56h, 0F4h, 0EAh, 65h, 7Ah, 0AEh, 8, 0BAh, 78h, 25h
.rodata:0000000000000AF3                 db 2Eh, 1Ch, 0A6h, 0B4h, 0C6h, 0E8h, 0DDh, 74h, 1Fh, 4Bh
.rodata:0000000000000AFD                 db 0BDh, 8Bh, 8Ah, 70h, 3Eh, 0B5h, 66h, 48h, 3, 0F6h, 0Eh
.rodata:0000000000000B08                 db 61h, 35h, 57h, 0B9h, 86h, 0C1h, 1Dh, 9Eh, 0E1h, 0F8h
.rodata:0000000000000B12                 db 98h, 11h, 69h, 0D9h, 8Eh, 94h, 9Bh, 1Eh, 87h, 0E9h
.rodata:0000000000000B1C                 db 0CEh, 55h, 28h, 0DFh, 8Ch, 0A1h, 89h, 0Dh, 0BFh, 0E6h
.rodata:0000000000000B26                 db 42h, 68h, 41h, 99h, 2Dh, 0Fh, 0B0h, 54h, 0BBh, 16h
.rodata:0000000000000B26 _rodata         ends
```

那么这时就明白了jaadx里的DES其实和flag并不相干，ida中的十六进制常数是小端序的目标字节序列，byteA30是标准的AES的S盒，要解密则是需要逆向

```
byte_A30[ input[v6] ] - (5 * (v6 / 5) + 4 == v6) == target[v6]
```

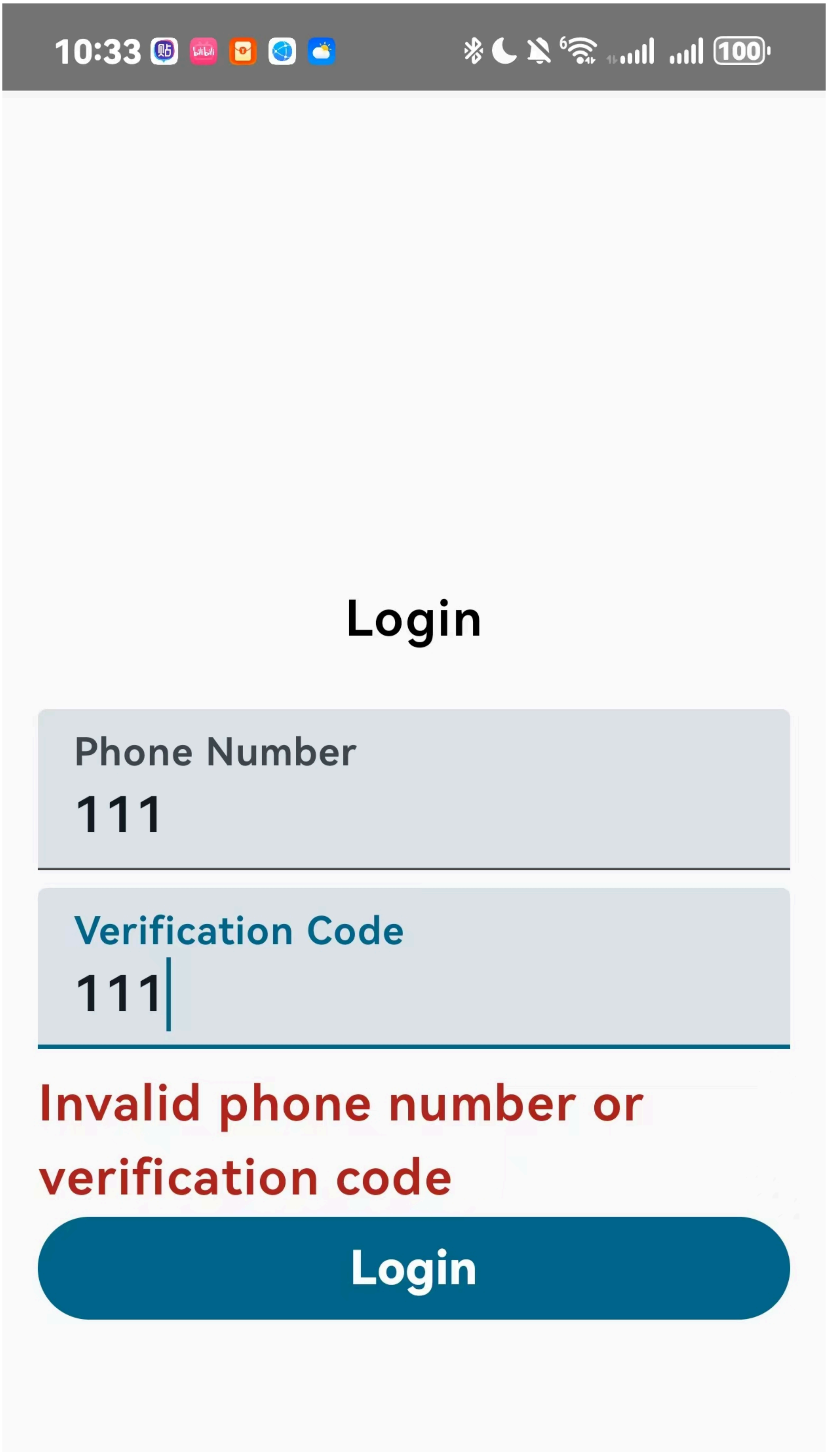
解密脚本如下

```
import struct

def solve():
    # 1. 提取目标字节 (Target Bytes)
    # 这里的常数来自 IDA 反编译图中的 __strcpy_chk 参数部分
    # 注意：这些是小端序 (Little Endian) 存储的
    # 0x8F9DD62085EF5033
    # 0xCF918F40F95ACE92
    # 4273356832 -> 0xFEB9F6A0
    val1 = 0x8F9DD62085EF5033
    val2 = 0xCF918F40F95ACE92
    val3 = 4273356832

    # 将长整型转换为字节数组 (Little Endian)
    target_bytes = list(struct.pack('<Q', val1)) + \
                   list(struct.pack('<Q', val2)) + \
                   list(struct.pack('<I', val3))

    # 目前提取的字节序列:
    # 33 50 EF 85 20 D6 9D 8F 92 CE 5A F9 40 8F 91 CF A0 F6 B9 FE

    # 2. 定义 AES S-Box (来源于 byte_A30)
    sbox = [
        0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F, 0xC5, 0x30, 0x01, 0x67, 0x2B,
0xFE, 0xD7, 0xAB, 0x76,
        0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xAF,
0x9C, 0xA4, 0x72, 0xC0,
        0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1,
0x71, 0xD8, 0x31, 0x15,
        0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07, 0x12, 0x80, 0xE2,
0xEB, 0x27, 0xB2, 0x75,
        0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3,
0x29, 0xE3, 0x2F, 0x84,
        0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39,
0x4A, 0x4C, 0x58, 0xCF,
        0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85, 0x45, 0xF9, 0x02, 0x7F,
0x50, 0x3C, 0x9F, 0xA8,
        0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21,
0x10, 0xFF, 0xF3, 0xD2,
        0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17, 0xC4, 0xA7, 0x7E, 0x3D,
0x64, 0x5D, 0x19, 0x73,
        0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46, 0xEE, 0xB8, 0x14,
0xDE, 0x5E, 0x0B, 0xDB,
        0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62,
0x91, 0x95, 0xE4, 0x79,
        0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA,
0x65, 0x7A, 0xAE, 0x08,
        0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8, 0xDD, 0x74, 0x1F,
0x4B, 0xBD, 0x8B, 0x8A,
        0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E, 0x61, 0x35, 0x57, 0xB9,
0x86, 0xC1, 0x1D, 0x9E,
        0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B, 0x1E, 0x87, 0xE9,
0xCE, 0x55, 0x28, 0xDF,
        0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68, 0x41, 0x99, 0x2D, 0x0F,
0xB0, 0x54, 0xBB, 0x16
    ]

    # 创建 S-Box 的反向查找表
    inv_sbox = {v: k for k, v in enumerate(sbox)}

    flag = ""

    # 3. 解密循环
    for i in range(len(target_bytes)):
        # 计算 Modifier: 对应代码中的 (5 * (i // 5) + 4 == i)
        # 即当索引 i % 5 == 4 时，Modifier 为 1，否则为 0
        modifier = 1 if (i % 5 == 4) else 0

        # 原始逻辑: Sbox[input] - modifier == target
        # 逆向逻辑: Sbox[input] == target + modifier
        # 所以我们需要找 Sbox 中值为 (target + modifier) 的索引
        target_val = target_bytes[i]
        search_val = (target_val + modifier) & 0xFF

        if search_val in inv_sbox:
            recovered_char_code = inv_sbox[search_val]
            flag += chr(recovered_char_code)
        else:
            flag += "?"
            print(f"Error at index {i}: val {search_val} not in Sbox")

    print(f"解密出的 Flag: {flag}")

if __name__ == '__main__':
    solve()
```

ReverseGame

这道题不知道为什么我的jadx并没有反编译出密文密钥，而官方wp中是可以搜索到的，不解，所以理应来说我只能获取到密钥，那么就说一下过程吧（其实只用jadx就能做，但是我没有密文密钥）

首先jadx打开有mainactivity和mainactivitykt两个主要部分，最主要的就在kt里，逻辑大概为
第一关，假登录，只要手机号和验证码不为空，就会调用 `onLoginSuccess` 跳转到下一页，但是运行的话会出现错误提示，


10:33

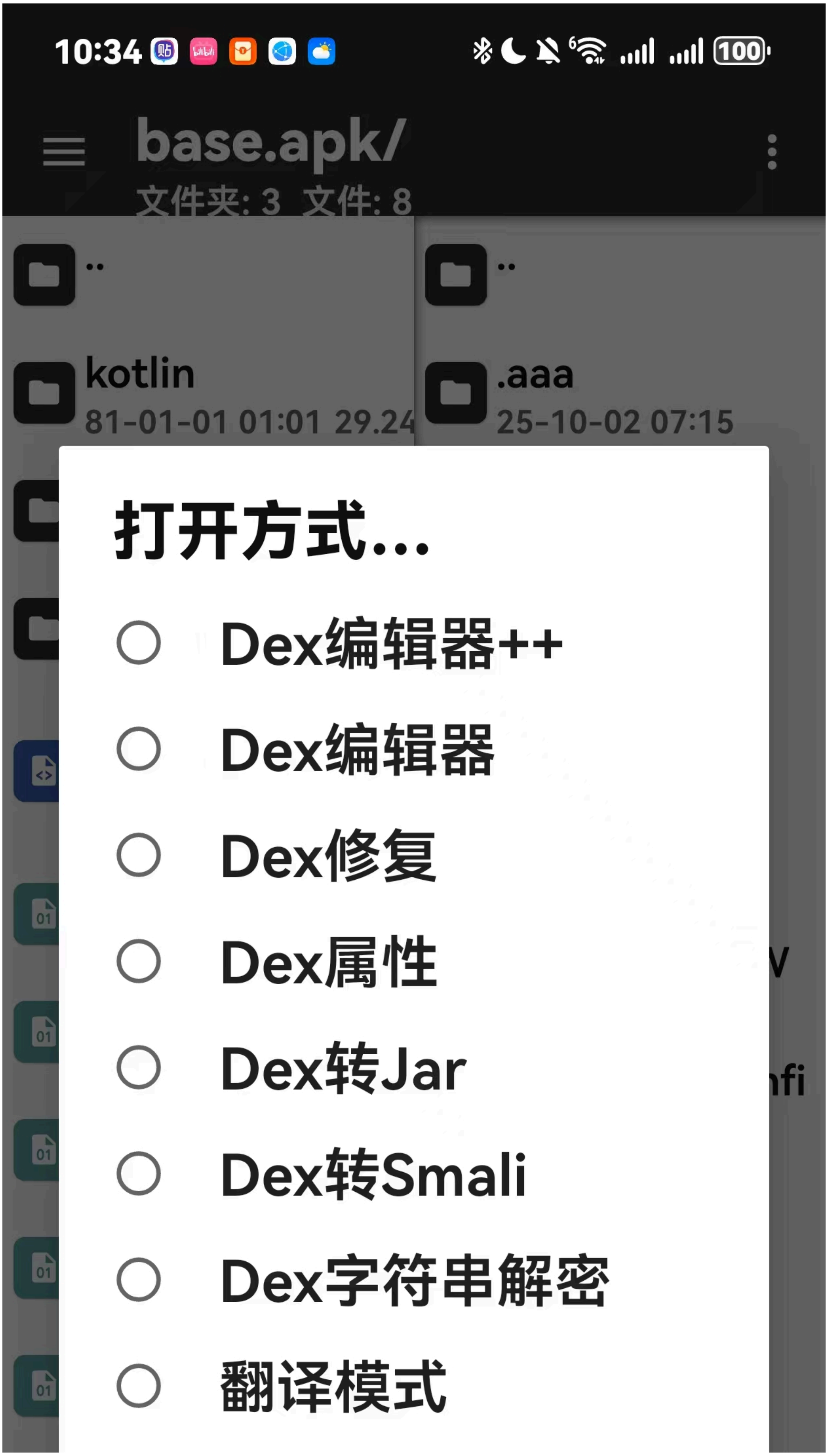
Login

Phone Number
111

Verification Code
111

Invalid phone number or verification code

用mt打开，打开class.dex文件，（class.dex包含了应用程序的所有编译后的代码，是Dalvik虚拟机（Android平台的虚拟机）可以理解和执行的格式）



10:34

base.apk/

文件夹: 3 文件: 8

..

..

kotlin

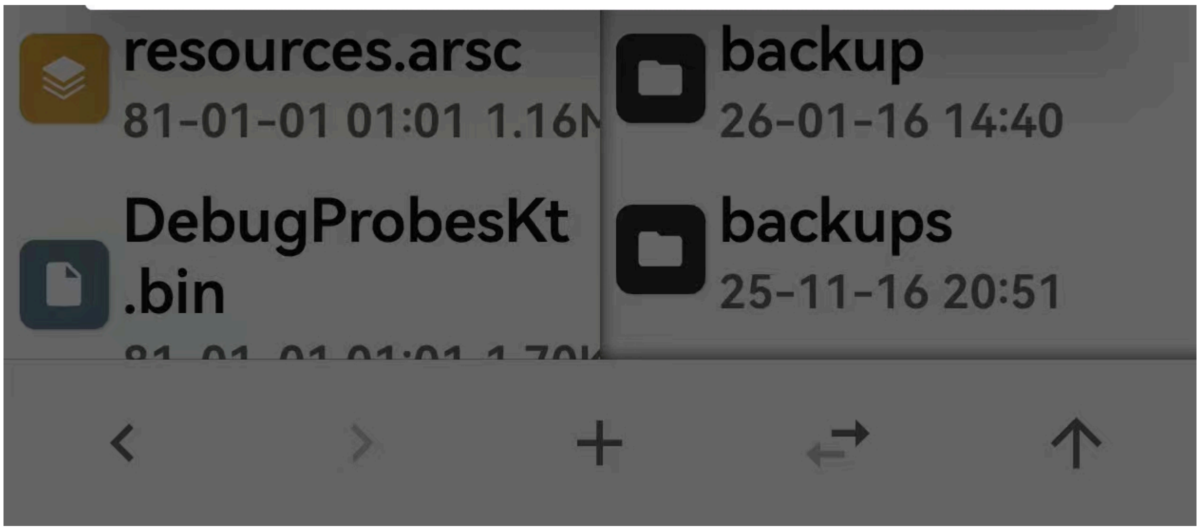
81-01-01 01:01 29.24

.aaa

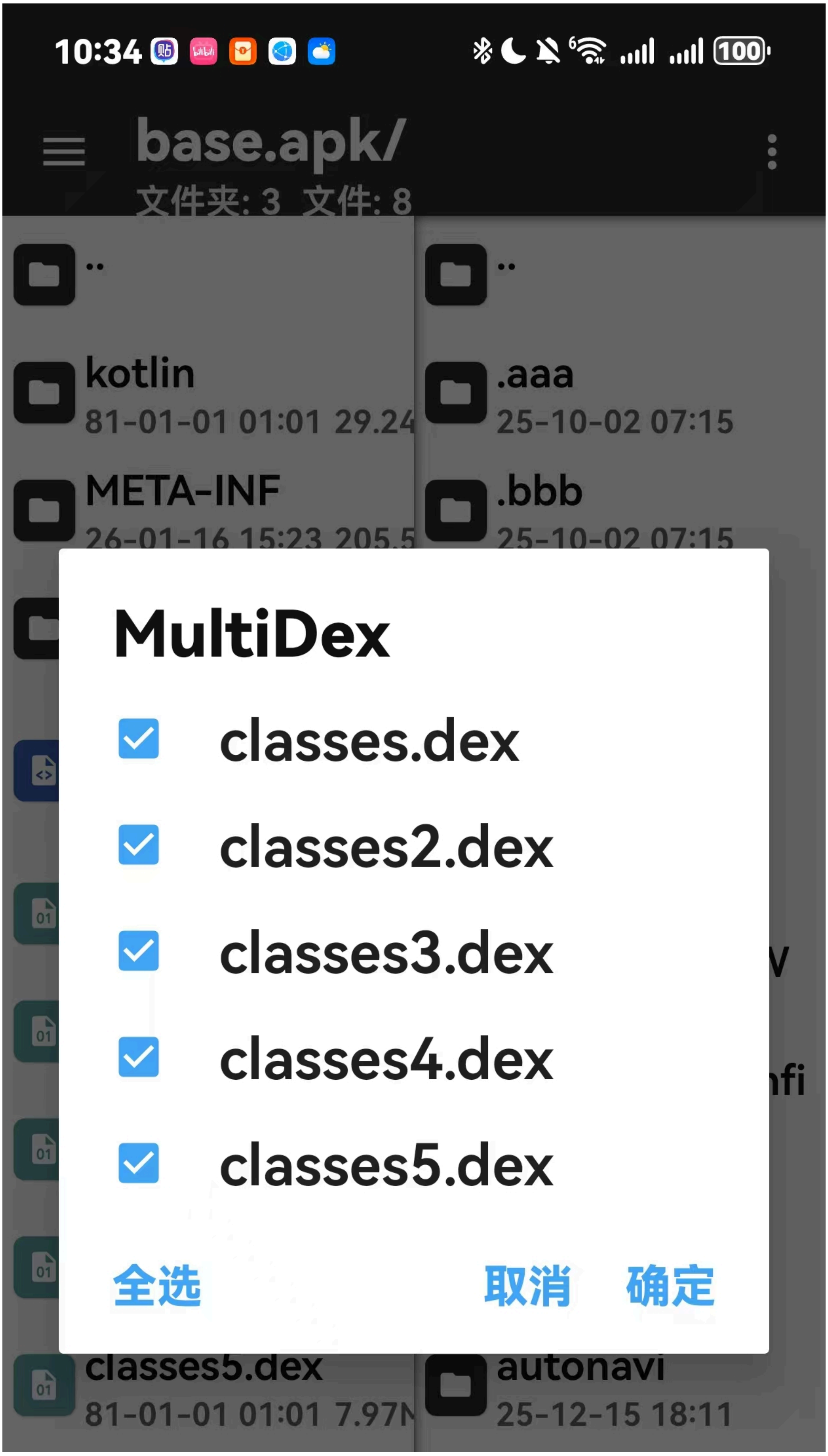
25-10-02 07:15

打开方式...

- ○ Dex编辑器++
- ○ Dex编辑器
- ○ Dex修复
- ○ Dex属性
- ○ Dex转Jar
- ○ Dex转Smali
- ○ Dex字符串解密
- ○ 翻译模式


resources.arsc
81-01-01 01:01 1.16M
DebugProbesKt.bin
backup
26-01-16 14:40
backups
25-11-16 20:51

选择第一个，



10:34

base.apk/

文件夹: 3 文件: 8

..

kotlin
81-01-01 01:01 29.24

META-INF
26-01-16 15:23 205.5

classes5.dex
81-01-01 01:01 7.97M

..

.aaa
25-10-02 07:15

.bbb
25-10-02 07:15

autonavi
25-12-15 18:11

MultiDex

☑ classes.dex

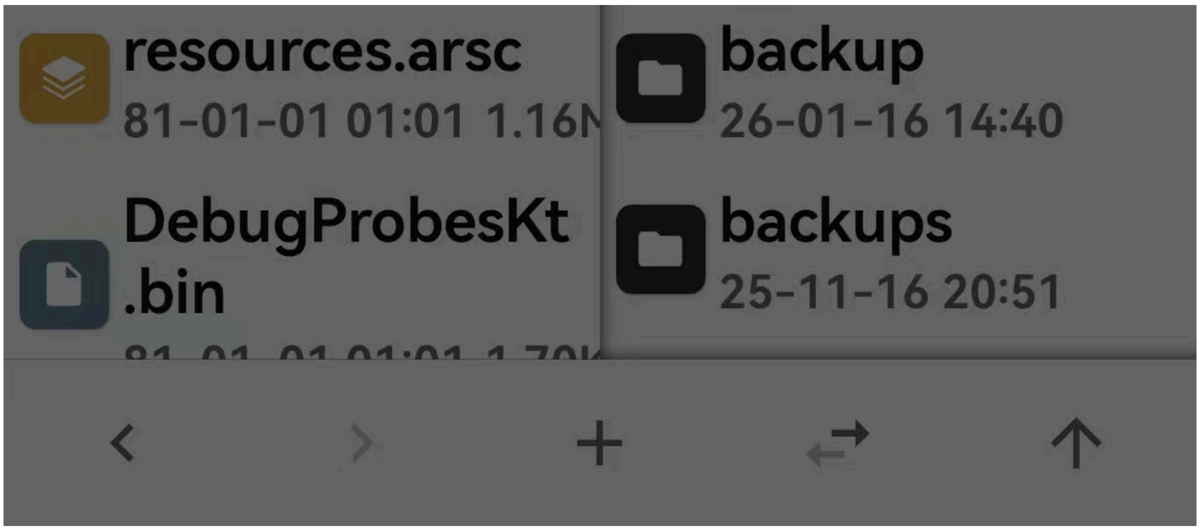
☑ classes2.dex

☑ classes3.dex

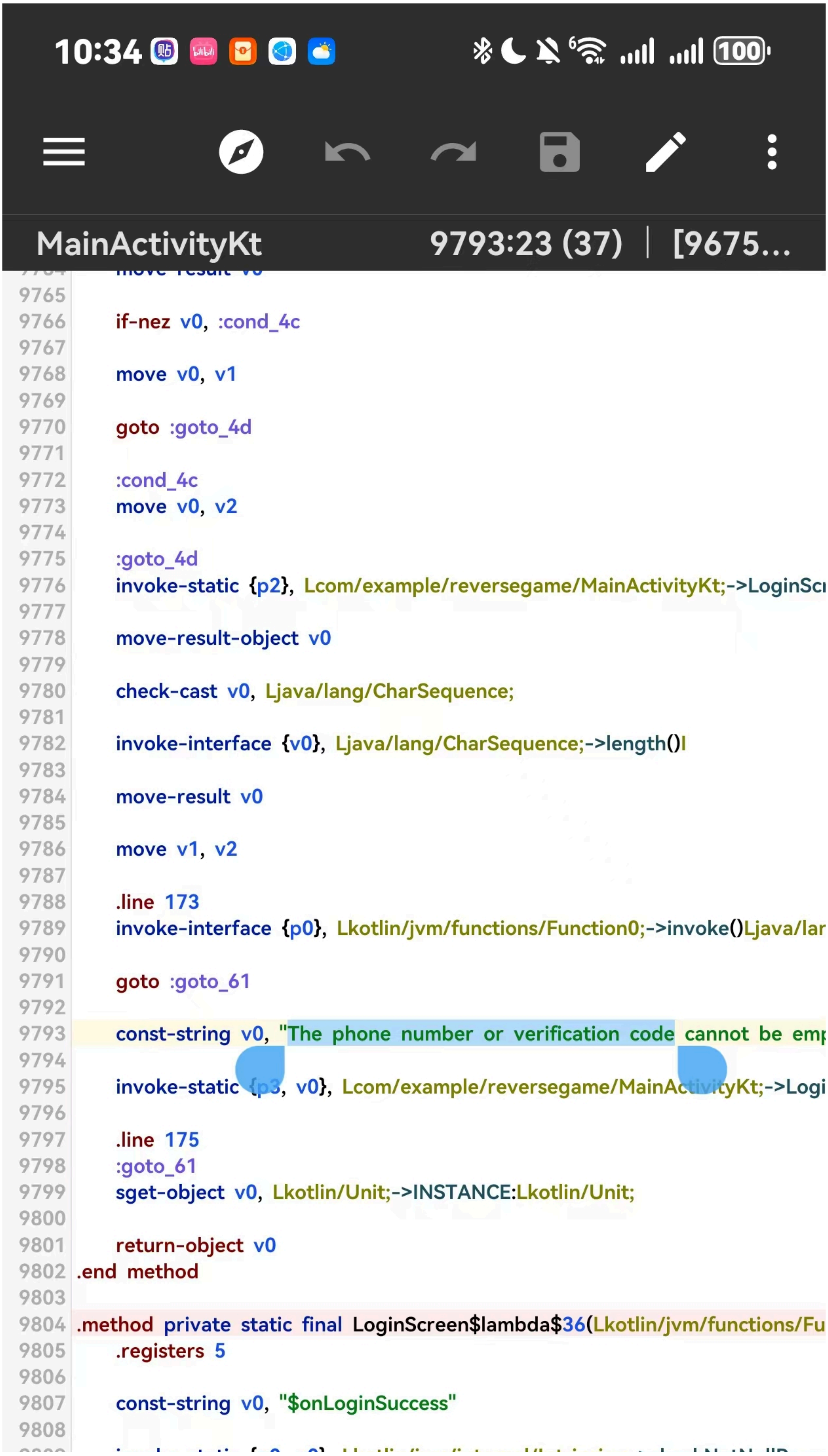
☑ classes4.dex

☑ classes5.dex

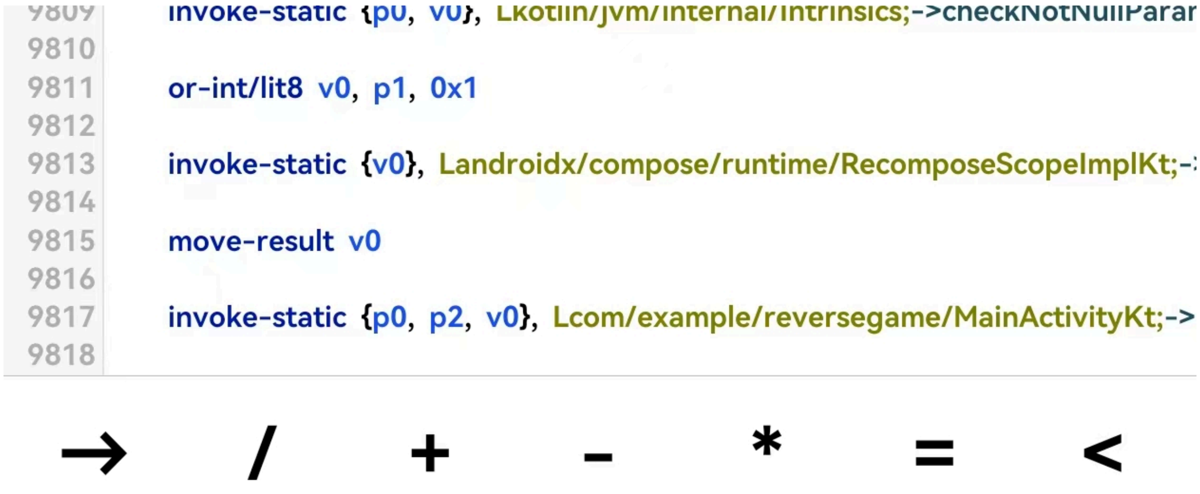
全选　　取消　确定


resources.arsc
81-01-01 01:01 1.16M
DebugProbesKt
.bin
backup
26-01-16 14:40
backups
25-11-16 20:51

插一句：（欸，你为什么不用模拟器要用真机啊？因为真机方便而且简单好操作）

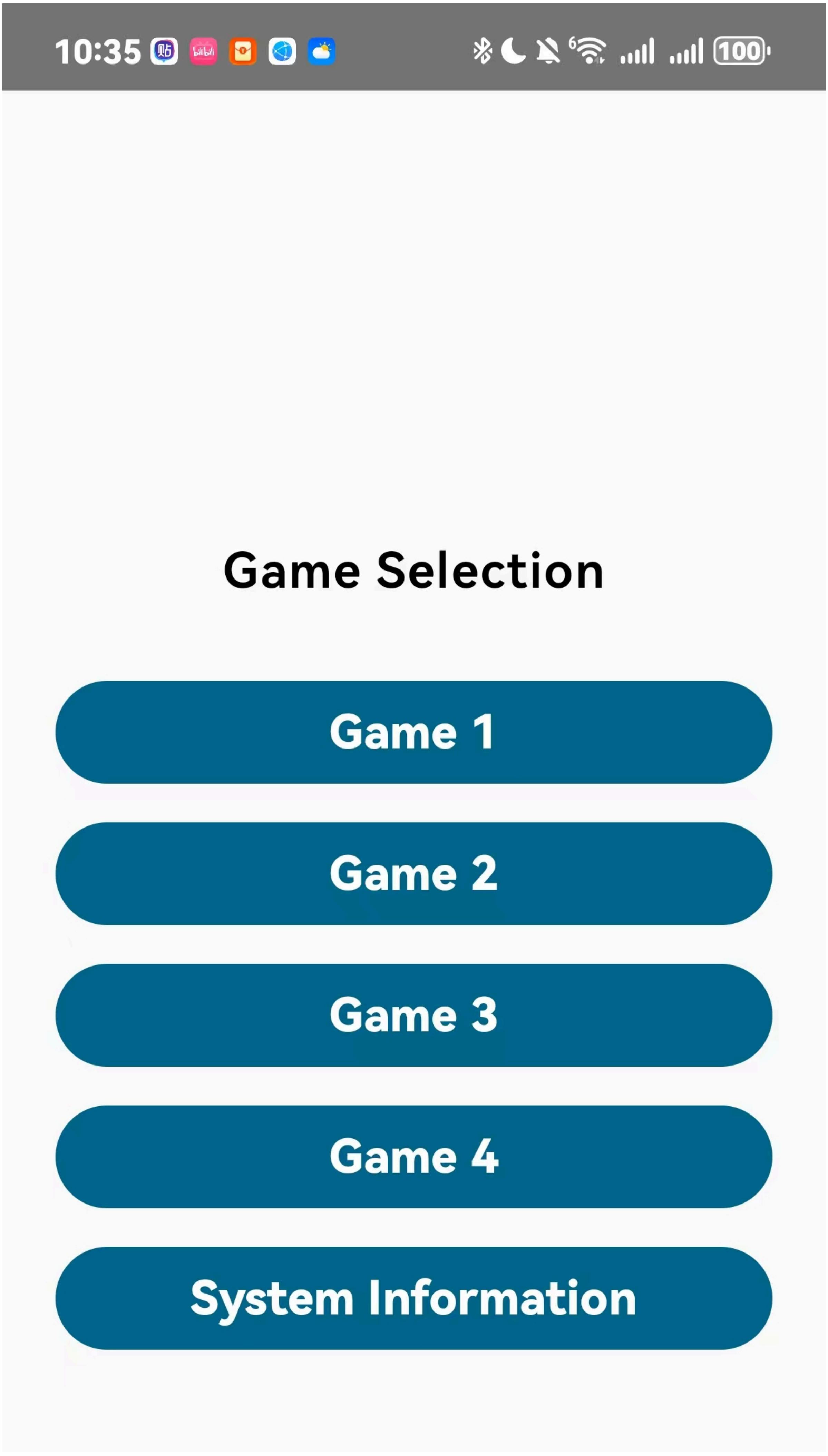


```
    move-result v0

    if-nez v0, :cond_4c

    move v0, v1

    goto :goto_4d

    :cond_4c
    move v0, v2

    :goto_4d
    invoke-static {p2}, Lcom/example/reversegame/MainActivityKt;->LoginScr

    move-result-object v0

    check-cast v0, Ljava/lang/CharSequence;

    invoke-interface {v0}, Ljava/lang/CharSequence;->length()I

    move-result v0

    move v1, v2

    .line 173
    invoke-interface {p0}, Lkotlin/jvm/functions/Function0;->invoke()Ljava/lar

    goto :goto_61

    const-string v0, "The phone number or verification code cannot be emp

    invoke-static {p3, v0}, Lcom/example/reversegame/MainActivityKt;->Logi

    .line 175
    :goto_61
    sget-object v0, Lkotlin/Unit;->INSTANCE:Lkotlin/Unit;

    return-object v0
.end method

.method private static final LoginScreen$lambda$36(Lkotlin/jvm/functions/Fu
    .registers 5

    const-string v0, "$onLoginSuccess"
```

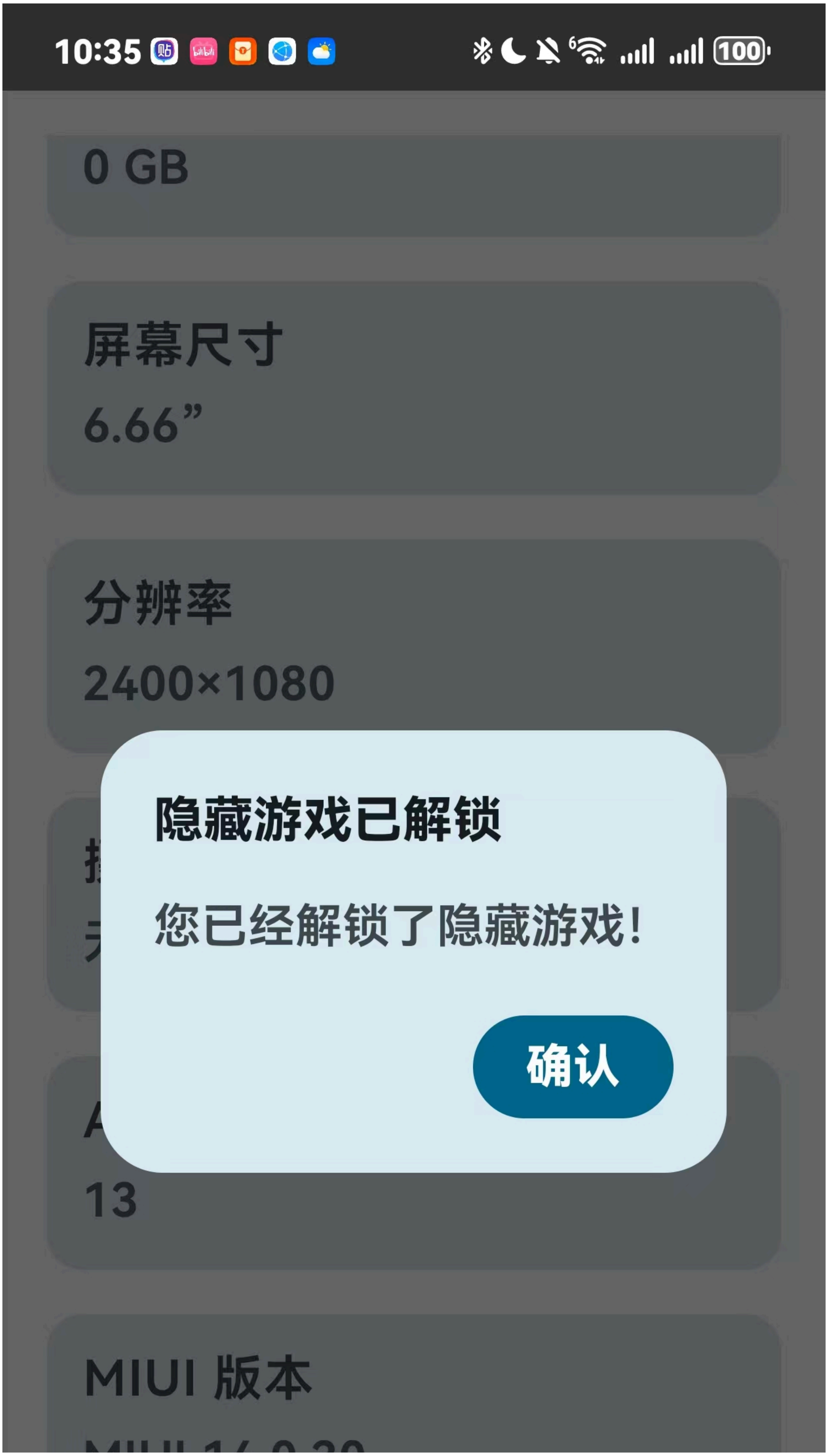


```
    invoke-static {p0, v0}, Lkotlin/jvm/internal/Intrinsics;->checkNotNullParar

    or-int/lit8 v0, p1, 0x1

    invoke-static {v0}, Landroidx/compose/runtime/RecomposeScopeImplKt;-

    move-result v0

    invoke-static {p0, p2, v0}, Lcom/example/reversegame/MainActivityKt;->
```

定位到我们刚刚的报错的话，然后把周围有关于认证的的if都删掉，（这个截图已经删掉了）


10:35
100
Game Selection
Game 1
Game 2
Game 3
Game 4
System Information

直接进，点最下面的system



10:35

0 GB

屏幕尺寸

6.66”

分辨率

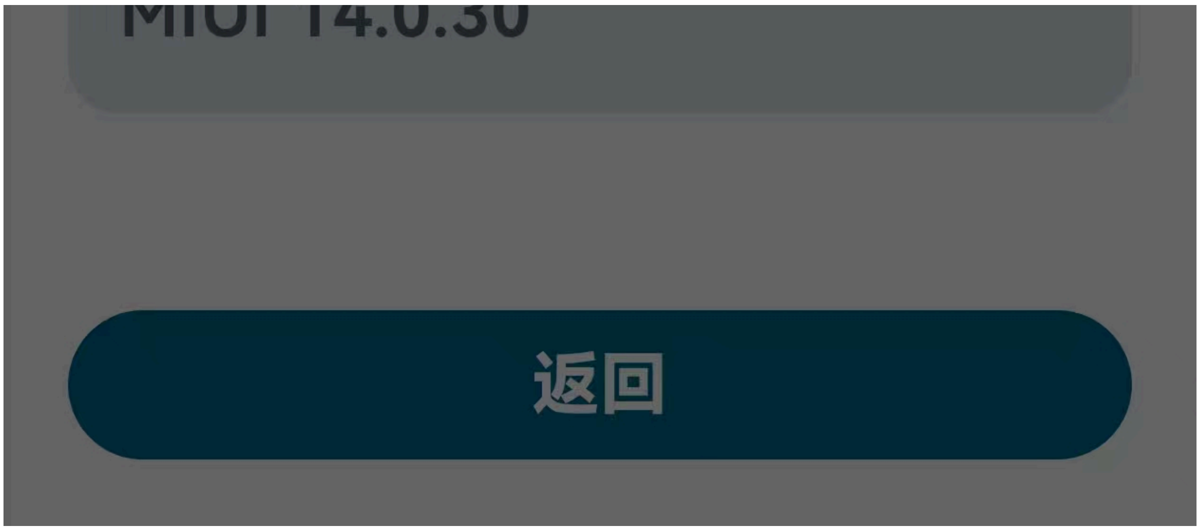
2400×1080

隐藏游戏已解锁

您已经解锁了隐藏游戏!

13

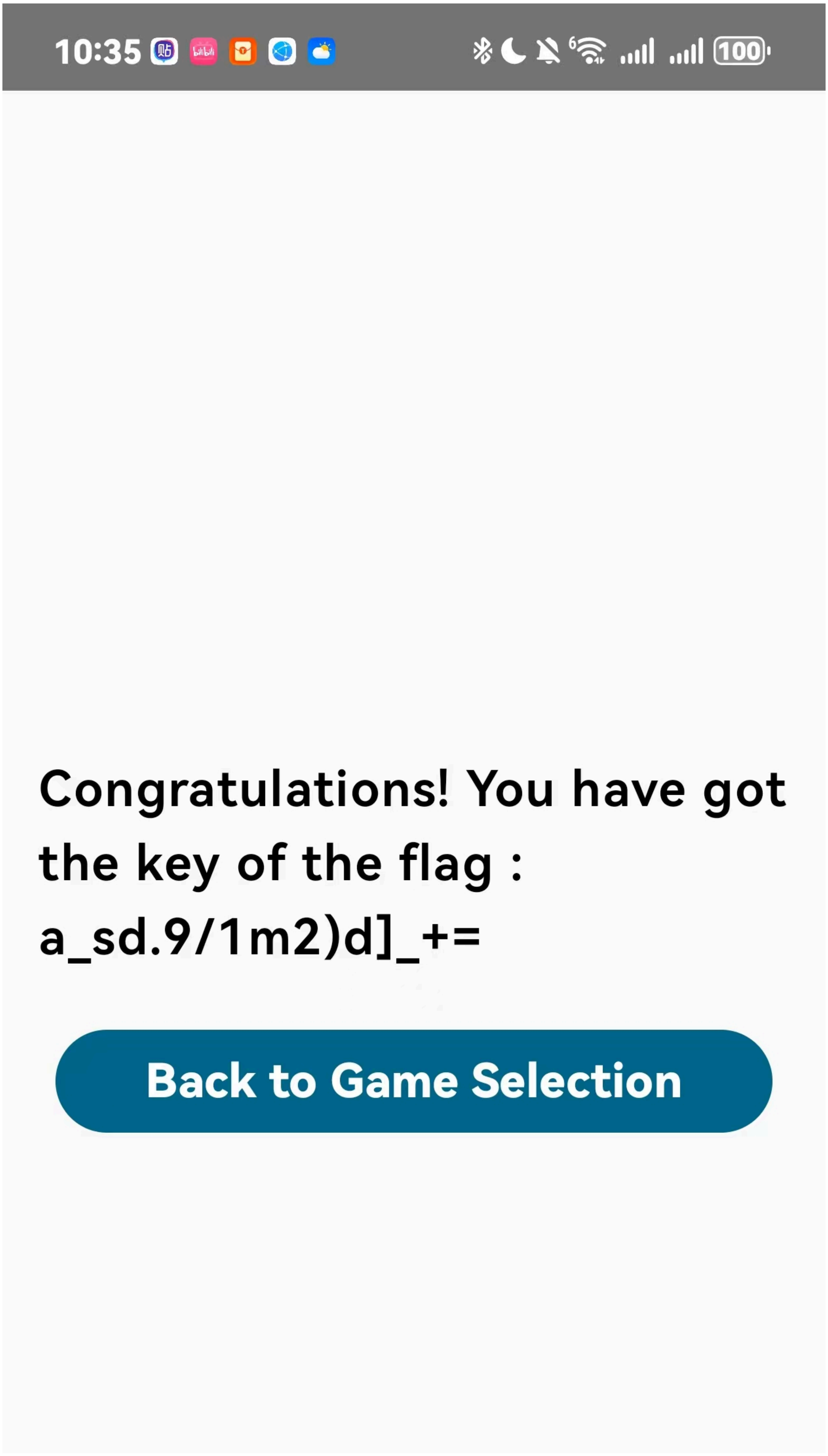
MIUI 版本



然后点击安卓型号十次解锁隐藏游戏（当然这个可以直接在jiadx分析出来）

```
if (clickCount >= 10 && !isHiddenGameUnlocked)
```

然后隐藏游戏也是可以通过在mt中搜索错误信息修改条件通过



Congratulations! You have got the key of the flag : a_sd.9/1m2)d]_+=

这就是密钥，官方wp中拿这个密钥在jadx中搜索直接看到密文，但我搜不到（，
总之这道题其实用jadx完全可以做出，不需要在手机上运行。
下一篇博客发一下解包，修改，打包，签名的流程吧