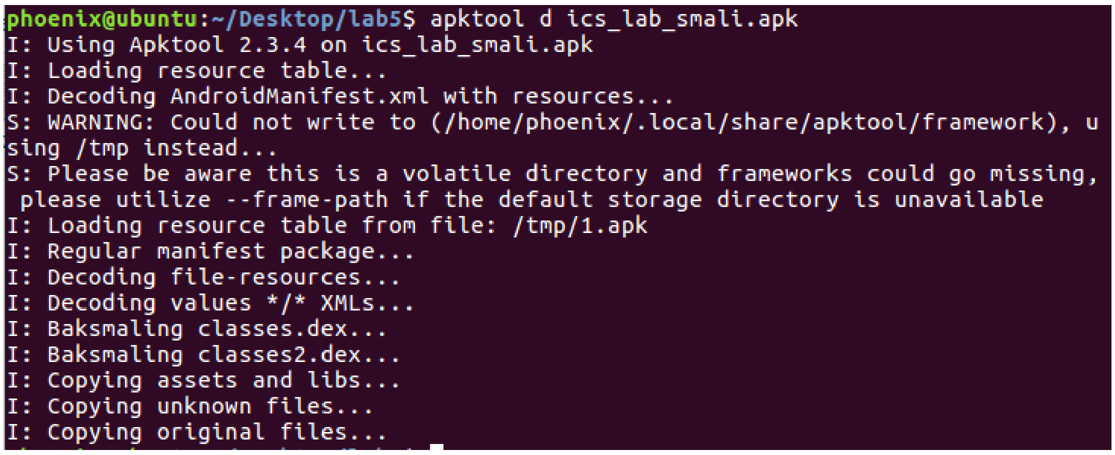
Smali Lab

成功安装 apktool，运行命令“apktool d ics_lab_smali.apk”，显示结果如下：



```
phoenix@ubuntu:~/Desktop/lab5$ apktool d ics_lab_smali.apk
I: Using Apktool 2.3.4 on ics_lab_smali.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
S: WARNING: Could not write to (/home/phoenix/.local/share/apktool/framework), u
sing /tmp instead...
S: Please be aware this is a volatile directory and frameworks could go missing,
 please utilize --frame-path if the default storage directory is unavailable
I: Loading resource table from file: /tmp/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Baksmaling classes2.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
```

在文件 MainActivity.smali 中，与 check 方法有关的内容如下：

```
.method public check(Ljava/lang/String;)Ljava/lang/String;
    .locals 13
    .param p1, "in"    # Ljava/lang/String;

    .line 42
    const/4 v0, 0x5

    .line 43
    .local v0, "k1":I
    const/16 v1, 0x9

    .line 44
    .local v1, "k2":I
    invoke-virtual {p1}, Ljava/lang/String;->length()I

    move-result v2

    .line 46
    .local v2, "len":I
    const-string v3, ""

    .line 47
    .local v3, "out":Ljava/lang/String;
    const/4 v4, 0x2

    new-array v4, v4, [C

    fill-array-data v4, :array_0

    .line 49
    .local v4, "b":[C
    add-int/lit8 v5, v2, -0x9

    const/4 v6, 0x1

    :try_start_0
    div-int v5, v6, v5
    :try_end_0
    .catch Ljava/lang/Exception; {:try_start_0 .. :try_end_0} :catch_1
    .line 50
    .local v5, "d":I
    const/4 v7, 0x0

    move-object v8, v3

    const/4 v3, 0x0

    .local v3, "i":I
    .local v8, "out":Ljava/lang/String;
    :goto_0
    if-ge v3, v2, :cond_0

    .line 51
    :try_start_1
    invoke-virtual {p1}, Ljava/lang/String;->toLowerCase()Ljava/lang/String;

    move-result-object v9

    invoke-virtual {v9}, Ljava/lang/String;->toCharArray()[C

    move-result-object v9

    aget-char v9, v9, v3

    aget-char v10, v4, v7

    sub-int/2addr v9, v10

    .line 52
    .local v9, "enc":I
    new-instance v10, Ljava/lang/StringBuilder;

    invoke-direct {v10}, Ljava/lang/StringBuilder;-><init>()V

    invoke-virtual                              {v10,                              v8},
Ljava/lang/StringBuilder;->append(Ljava/lang/String;)Ljava/lang/StringBuilder;

    mul-int v11, v0, v9

    add-int/2addr v11, v1

    rem-int/lit8 v11, v11, 0x1a

    aget-char v12, v4, v6

    add-int/2addr v11, v12

    int-to-char v11, v11

    invoke-static {v11}, Ljava/lang/String;->valueOf(C)Ljava/lang/String;

    move-result-object v11

    invoke-virtual                              {v10,                              v11},
Ljava/lang/StringBuilder;->append(Ljava/lang/String;)Ljava/lang/StringBuilder;

    invoke-virtual {v10}, Ljava/lang/StringBuilder;->toString()Ljava/lang/String;

    move-result-object v10
    :try_end_1
    .catch Ljava/lang/Exception; {:try_start_1 .. :try_end_1} :catch_0

    move-object v8, v10

    .line 50
    add-int/lit8 v3, v3, 0x1
```

```
        goto :goto_0

        .line 54
        .end local v3       # "i":I
        .end local v5       # "d":I
        .end local v9       # "enc":I
        :catch_0
        move-exception v3

        goto :goto_1

        .line 57
        :cond_0
        goto :goto_2

        .line 54
        .end local v8       # "out":Ljava/lang/String;
        .local v3, "out":Ljava/lang/String;
        :catch_1
        move-exception v5

        move-object v8, v3

        move-object v3, v5

        .line 55
        .local v3, "e":Ljava/lang/Exception;
        .restart local v8       # "out":Ljava/lang/String;
        :goto_1
        const/4 v5, 0x3

        new-array v5, v5, [C

        fill-array-data v5, :array_1

        .line 56
        .local v5, "c":[C
        invoke-static {v5}, Ljava/lang/String;->valueOf([C)Ljava/lang/String;

        move-result-object v8

        .line 58
        .end local v3       # "e":Ljava/lang/Exception;
        .end local v5       # "c":[C
        :goto_2
        return-object v8

        nop

        :array_0
        .array-data 2
        0x61s
        0x41s
        .end array-data

        :array_1
        .array-data 2
        0x65s
        0x72s
        0x72s
        .end array-data
.end method
```

1. The Check Function:

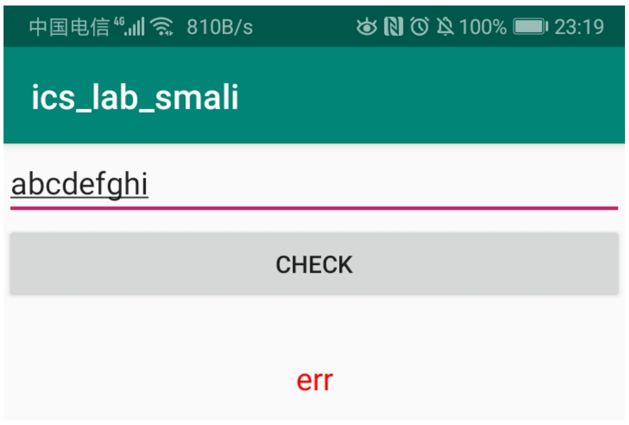
根据 smali 的规则转换为 java 形式:

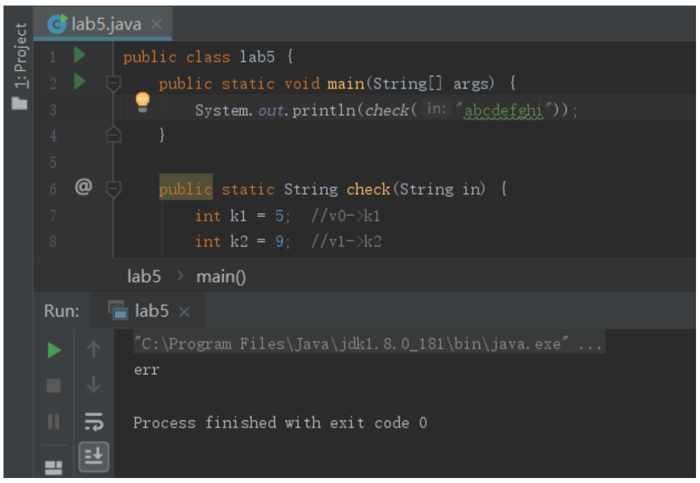
```
public String check(String in) {
        int k1 = 5;    //v0->k1
        int k2 = 9;    //v1->k2
        int len = in.length();    //v2->len
        String out = "";    //v3->out
        char[] b = {97, 65};    //v4->b
        int d;    //v5->d, v6->1
        try {
                d = 1 / (len - 9);
                //v7->0, v8->out
                for (int i = 0; i < len; i++) {    //v3->i
                        try {
                                int enc = in.toLowerCase().toCharArray()[i] - b[0];    //v9->enc
                                out = new StringBuilder().append(out).append(String.valueOf((char)
                                                (((k1 * enc) + k2) % 26 + b[1]))).toString();
                        } catch (Exception e) {
                        }
                }
                return out;
        } catch (Exception e) {
        }
        char[] c = {101, 114, 114};
        return String.valueOf(c);
}
```

2. Find one input to get the result “err”:

由 java 形式的函数 check 可以发现，当 d = 1 / (len – 9) 抛出异常时，返回{101, 144, 144}，也即“err”。显然，只有 len，也就是输入的字符串的长度等于 9 时，得到的结果为“err”。

因此构造字符串“abcdefghi”，检验结果为：





```
public class lab5 {
    public static void main(String[] args) {
        System.out.println(check( in: "abcdefghi"));
    }

    public static String check(String in) {
        int k1 = 5;   //v0->k1
        int k2 = 9;   //v1->k2
```

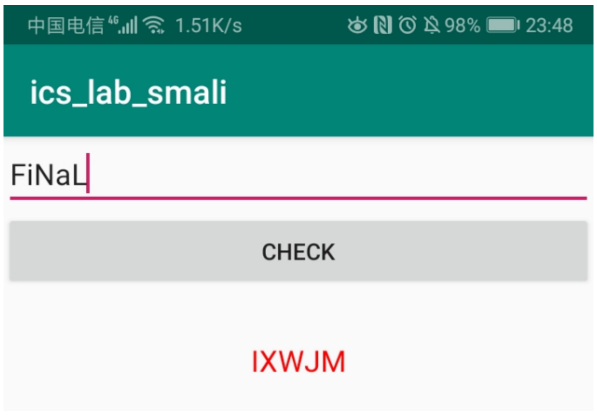
```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
err

Process finished with exit code 0
```

3. Find one input to get the result “IXWJM”:

由 java 形式的函数 check 可以发现，对不会抛出异常的输入的字符串，会先将其中的大写字母都转换为小写字母，再将每一个字符的 ASCII 码 x，先减去 97，再乘 5 加 9，模 26 后加 65，最后重新组合成字符串。也就是新的字符的 ASCII 码为 y = ((x – 97) * 5 + 9) % 26 + 65。因此有 x = ((26 * n + y - 65) – 9) / 5 + 97，其中 (26 * n + y - 65) – 9) 应当被 5 整除。

因为“IXWJM”的 ASCII 码分别为“73, 88, 87, 74, 77”，假定原字符均为字母，容易得出对应的原始 ASCII 码分别为“102, 105, 110, 97, 108”，也就是“final”，其中每一个小写字母都可以被其对应的大写字母替代。

因此构造字符串“FiNaL”，检验结果：



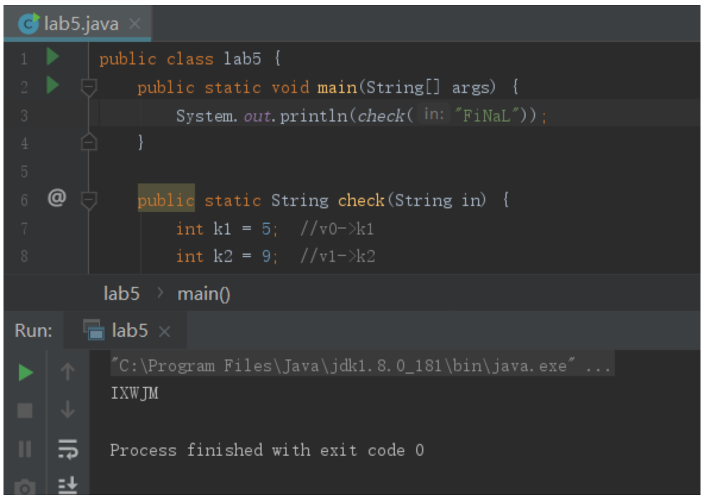
中国电信 1.51K/s 98% 23:48

ics_lab_smali

FiNaL

CHECK

IXWJM



lab5.java

```java
public class lab5 {
    public static void main(String[] args) {
        System.out.println(check( in: "FiNaL"));
    }

    public static String check(String in) {
        int k1 = 5;  //v0->k1
        int k2 = 9;  //v1->k2
```

lab5 › main()

Run: lab5

```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
IXWJM

Process finished with exit code 0
```