S-DES 加密解密程序测试结果

3.1 第1关：基本测试

根据S-DES算法编写和调试程序，提供GUI解密支持用户交互。输入可以是8bit的数据和10bit的密钥，输出是8bit的密文。

我们在 UI 界面中勾选 Bit，并给出满足要求的输入进行测试。

1.加密：输入 10bit 密钥以及 8bit 明文，选择 Bit 编码类型，点击加密按钮，即可在密文的文本框里获得 8bit 密文。

2.解密：输入 10bit 密钥以及 8bit 密文，选择 Bit 编码类型，点击解密按钮，即可在密文的文本框里获得 8bit 明文。

可以看到明文加密解密后得到本身，说明我们的程序能够正确处理 Bit 输入。

3.2 第2关：交叉测试

考虑到是**算法标准**，所有人在编写程序的时候需要使用相同算法流程和转换单元(P-Box、S-Box等)，以保证算法和程序在异构的系统或平台上都可以正常运行。

设有A和B两组位同学(选择相同的密钥K)；则A、B组同学编写的程序对明文P进行加密得到相同的密文C；或者B组同学接收到A组程序加密的密文C，使用B组程序进行解密可得到与A相同的P。

本算法已与张芷芮刘俐莹组、罗丹陈露组、宋选存朱佩芩组进行共同测验通过。

经过三次的交叉测试，可以看出我们的算法实现是正确的。

3.3 第3关：扩展功能

考虑到向实用性扩展，加密算法的数据输入可以是ASII编码字符串(分组为1 Byte)，对应地输出也可以是ACII字符串(很可能是乱码)。

1.加密：输入一个 String 类型的 ASCII 编码字符串明文和 10bits 的秘钥 key，选择 ASCII 编码类型，点击加密按钮，获得密文。

2.解密：输入密文和 10bits 的秘钥，选择 ASCII 编码类型，点击解密按钮，获得明文。

经过测试，可以看出我们对 ASCII 码的处理无误。

3.4 第4关：暴力破解

假设你找到了使用相同密钥的明、密文对(一个或多个)，请尝试使用暴力破解的方法找到正确的密钥Key。在编写程序时，你也可以考虑使用多线程的方式提升破解的效率。请设定时间戳，用视频或动图展示你在多长时间内完成了暴力破解。

我们尝试了暴力破解三对明密文对，破解的平均时间 2ms。

运行如下：

```
// 明密文对
//key=1111100000
plaintextList.add(new int[]{0, 1, 0, 1,0, 1, 0, 1});
ciphertextList.add(new int[]{0, 0, 0, 1, 0, 1, 1, 1});

//key=1010101010
plaintextList.add(new int[]{0,0, 0, 0, 0,0, 0, 0});
ciphertextList.add(new int[]{1, 1, 1, 1, 1, 0, 1, 0});

//key=1011010010
plaintextList.add(new int[]{ 0, 0, 1, 0, 1,1,1,1});
ciphertextList.add(new int[]{1, 1, 0,0, 0, 0, 0, 1});
```

```
Possible key is:  0000110100
Crack Time: 3222 microsecond
Possible key is:  1001110000
Crack Time: 3545 microsecond
Possible key is:  1000101101
Crack Time: 1416 microsecond
Average Decryption Time is 2 millisecond
```

3.5 第5关：封闭测试

根据第4关的结果，进一步分析，对于你随机选择的一个明密文对，是不是有不止一个密钥Key? 进一步扩展，对应明文空间任意给定的明文分组 P_n ，是否会出现选择不同的密钥 $K_i \neq K_j$ 加密得到相同密文 C_n 的情况?

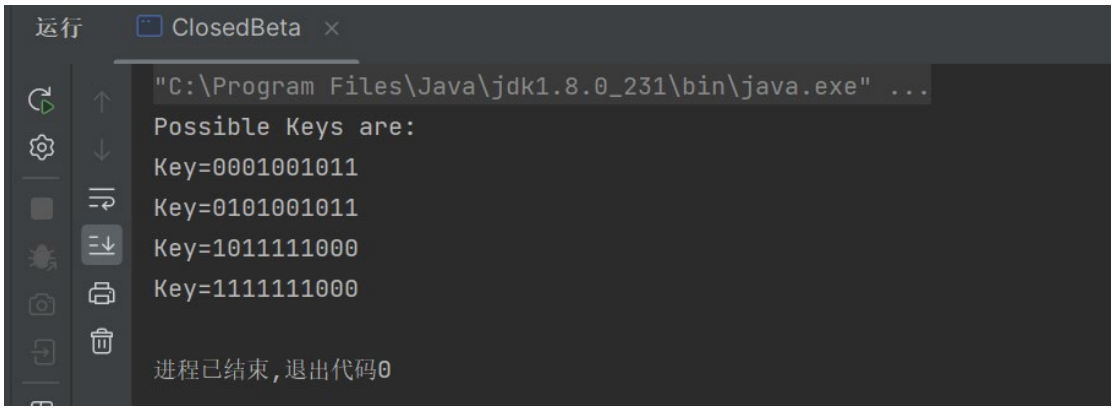
是。我们尝试了对一个明文密文对解析，发现有 4 种不同的密钥可以用于加密得到相同的密文。

```java
public static void main(String[] args) {
    // 存储找到的密钥
    List<String> foundKeys = new ArrayList<>();

    //key=1111111000
    int[] plaintext = {1,1,1,1,1,1,1,1};
    int[] ciphertext = {0,1,0,1,1, 1, 1, 1};

    for (int i = 0; i < 1024; i++) {
        // 将i转换为10位二进制形式作为密钥
        String binaryKey = Integer.toBinaryString(i);
        while (binaryKey.length() < 10) {
            // 确保密钥长度为10位
            binaryKey = "0" + binaryKey;
        }
        // 解密
        int[] decrypted = SDES_Decrypt.decrypt(ciphertext, binaryKey);
        // 比较是否一致
        if (Arrays.equals(plaintext, decrypted)) {
            foundKeys.add(binaryKey); // 将找到的密钥添加到列表中
        }
    }
```

找到的密钥如下：



根据刚才的结果可知，对于同一个明密文对破解后会出现不止一个密钥的情况。

因此对应明文空间任意给定的明文分组P_n，会出现选择不同的密钥$K(i)! = K(j)$，但加密得到相同密文C_n的情况，如下例：

S-DES 的输入中密钥空间为2^{10}，明文空间为2^8，而输出中的密文空间为2^8；由于$2^{10} \times 2^8 > 2^8$，输入组合比输出组合多，所以一定存在多个明文密钥输入组合映射到同一个输出密文的情况。