S-DES 加密解密工具开发手册

1. 概述

本项目实现了一个基于简化版数据加密标准 (S-DES) 的图形化工具，允许用户输入明文和密钥，进行加密和解密操作。该工具支持两种输入模式：ASCII 和 8 位二进制，并通过 Swing 界面呈现。用户可以通过该工具对文本进行加密或解密，便于理解和验证 S-DES 算法的工作原理。

2. 项目结构

- 包名: des
- 主类: desmethod，实现 S-DES 加密解密逻辑和图形化界面
- 依赖库: Java 标准库 (javax.swing, java.awt)

3. 核心功能

3.1 输入模式

工具支持两种输入模式：

- ASCII 模式: 用户可以输入任意文本，程序将其转换为二进制进行加密。
- 8 位二进制模式: 用户直接输入 8 位二进制数串，每 8 位作为一组进行加密或解密。

3.2 加密和解密

S-DES 加密过程基于输入明文和 10 位二进制密钥，经过一系列置换和 S 盒替换，生成密文。解密过程是加密的逆过程，使用相同的密钥进行恢复。

4. 开发环境

4.1 开发工具

Java JDK 8 或以上版本: 本项目使用 Java 语言开发，建议使用 JDK 8 或更新版本进行编译和运行。

集成开发环境 (IDE): 推荐使用 IntelliJ IDEA、Eclipse 或 NetBeans。

4.2 编译与运行

编译: 确保开发环境中已经配置好 Java 开发工具包（JDK），然后运行如下命令编译代码：

```
javac des/desmethod.java
```

运行: 编译成功后，执行以下命令启动程序：

```
java des.desmethod
```

5. 代码结构详解

5.1 S-DES 算法实现

5.1.1 密钥生成

- 通过 10 位二进制密钥生成两个子密钥 K1 和 K2：
- P10: 对输入密钥进行置换。
- LS1, LS2: 对密钥进行循环左移。
- P8: 对左移后的密钥再次置换，生成 K1 和 K2。

5.1.2 加密过程

• 初始置换 (IP): 对 8 位二进制明文执行初始置换。
• 第一轮函数应用 (F1):
对明文的右半部分进行扩展置换 (EP)。
与子密钥 K1 进行异或运算。
通过两个 S 盒（SS1, SS2）将结果映射到新的 4 位输出。
进行 P4 置换，生成结果与左半部分进行异或运算。
• 交换 (SW): 交换左右部分。
• 第二轮函数应用 (F2): 类似于 F1，但使用子密钥 K2。
• 逆初始置换 (IP^-1): 将结果执行逆初始置换，得到最终密文。

5.1.3 解密过程

解密过程与加密过程类似，但**子密钥的应用顺序相反**。首先使用 K2，然后使用 K1，恢复明文。

5.2 Swing 图形界面设计

5.2.1 主窗口

• 主窗口标题: "S-DES 加密解密工具"。
• 组件布局: 使用 GridBagLayout 布局管理器，调整组件位置及间距。

5.2.2 组件介绍

• JLabel: 显示提示文字，如输入模式、输入密钥等。
• JTextField: 用户输入明文/密文、密钥，及显示结果。
• JComboBox: 选择输入模式（ASCII 或 8 位二进制）。
• JButton: 加密、解密操作按钮。
• JLabel: 显示错误提示信息。

5.2.3 逻辑控制

输入模式选择 (JComboBox): 当选择 ASCII 输入模式时，禁用解密按钮；选择二进制模式时，启用解密按钮。

• 加密按钮: 获取用户输入的明文和密钥，执行加密操作，并在输出框显示结果。
• 解密按钮: 获取密文和密钥，执行解密操作，输出解密后的明文。

6. 关键功能函数

6.1 加密函数 encrypt

```
public static String encrypt(String plaintext, String K1, String K2) {
    plaintext = replace(plaintext, IP, 8);
    String L0 = plaintext.substring(0, 4);
    String R0 = plaintext.substring(4);
    String L1 = R0;

    String F1 = RandKtoS(R0, K1);
    String R1 = xorOperation(F1, L0);

    String F2 = RandKtoS(R1, K2);
    String L2 = xorOperation(L1, F2);
    String R2 = R1;
```

```
        String ciphertext = L2 + R2;
        return replace(ciphertext, IPIN, 8);
}
```

该函数接收明文和子密钥 K1, K2，执行 S-DES 加密流程，并返回密文。

6.2 解密函数 decrypt

```
public static String decrypt(String ciphertext, String K1, String K2) {
        ciphertext = replace(ciphertext, IP, 8);
        String L0 = ciphertext.substring(0, 4);
        String R0 = ciphertext.substring(4);
        String L1 = R0;

        String F1 = RandKtoS(R0, K2);
        String R1 = xorOperation(F1, L0);

        String F2 = RandKtoS(R1, K1);
        String L2 = xorOperation(L1, F2);
        String R2 = R1;

        String plaintext = L2 + R2;
        return replace(plaintext, IPIN, 8);
}
```

该函数与加密函数类似，使用相同的子密钥 K1, K2，但顺序相反。

6.3 辅助函数

- replace: 置换函数，按照指定数组对字符串进行重新排列。
- leftMove: 循环左移函数。
- xorOperation: 异或运算函数，用于两个二进制字符串进行逐位异或操作。
- RandKtoS: S 盒替换函数，用于将 4 位输入映射到新的 4 位输出。

7. 错误处理

如果用户输入的密钥不是 10 位二进制数，程序将显示错误提示信息，阻止进一步操作。当用户选择 ASCII 模式时，解密按钮不可用。

如果输入的二进制数长度不是 8 位的倍数，程序会提示错误，并要求用户重新输入。

8. 未来改进方向

- 增强密钥验证: 允许用户输入其他进制的密钥，并在内部进行转换。
- 优化用户体验: 在用户输入错误时，提供更详细的提示信息，如建议正确的输入格式等。
- 增加更多功能: 例如，增加更多的加密模式，如 AES 等。

9. 常见问题

9.1 如何输入有效的密钥？

密钥必须为 10 位二进制数（只包含 0 和 1）。如果密钥格式错误，程序将提示并拒绝继续操作。

9.2 ASCII 模式下为什么无法进行解密？

在 ASCII 模式下，工具只能进行加密。用户必须选择 8 位二进制模式才能进行解密操作。

9.3 程序无法启动，如何解决？

请确保已正确配置 Java 开发环境。检查是否安装了 JDK 并设置了 JAVA_HOME 环境变量。