**1. 引言**

**项目背景**

S-DES（Simplified Data Encryption Standard）是一种简化的数据加密标准，用于教学和学习目的。它模拟了DES算法的基本结构和加密过程，但使用较小的数据块和密钥，使得加密和解密过程更加容易理解和实现。

**项目目标**

本项目旨在实现一个基于S-DES算法的加密工具，提供图形用户界面（GUI）支持，使用户能够方便地进行加密和解密操作。同时，该项目也将作为学习加密算法实现的平台，通过一系列测试关卡，确保算法的正确性和效率。

**文档概述**

本开发文档旨在为开发者和用户提供全面的指导，包括安装指南、用户界面操作说明、开发手册以及接口文档。通过本文档，开发者可以了解系统的架构和API的使用，而用户可以快速掌握如何使用本工具进行加密和解密。

**2. 用户指南**

**安装指南**

**环境要求**：确保你的计算机安装了Java环境（推荐Java 8或更高版本）。

**下载程序：**从项目官网下载最新版本的程序安装包。

**安装程序：**双击下载的安装包，按照安装向导一步一步进行操作。

**启动程序：**安装完成后，可以在开始菜单或程序安装目录找到程序图标，双击打开程序。

**界面介绍**

**输入框：**用于输入8bit的明文或密文，以及10bit的密钥。

**操作按钮：**包括“加密”和“解密”按钮，用于执行相应的操作。

**输出框**：显示操作结果，即8bit的密文或明文。

**状态栏：**显示程序状态信息和错误提示。

功能使用说明

**加密过程：**在输入框中输入8bit的明文和10bit的密钥，点击“加密”按钮，输出框将显示加密后的8bit密文。

**解密过程：**在输入框中输入8bit的密文和相同的10bit密钥，点击“解密”按钮，输出框将显示解密后的8bit明文。

**暴力破解过程：**在输入框中输入8bit的密文和已知的明文。点击“暴力破解”按钮，程序将尝试所有可能的10bit密钥进行解密。破解完成后，输出框将显示找到的密钥和破解时间。

**错误提示与处理**

当输入的明文、密钥或密文长度不正确时，状态栏将显示错误提示，并阻止执行加密或解密操作。

**示例操作**

**加密示例：**输入明文“01010101”和密钥“1111111111”，点击“加密”按钮，输出框显示密文。

**解密示例：**输入密文（上一步骤的输出）和相同的密钥“1111111111”，点击“解密”按钮，输出框显示原始明文“01010101”。

**暴力破解示例：**输入密文（如“11001100”）和已知明文（如“01010101”），点击“暴力破解”按钮，程序将自动找到匹配的密钥并显示破解结果。

**3. 开发手册**

**系统架构：**

**用户界面模块**：负责与用户的交互，包括接收输入和显示输出。

**加密模块：**实现S-DES算法的加密过程。

**解密模块**：实现S-DES算法的解密过程。

**错误处理模块**：负责处理输入错误和异常情况。

**开发环境设置**

**推荐使用以下开发环境：**IDE：IntelliJ IDEA或Eclipse。

**语言：**Java。

**版本控制：**Git。

**编程接口（API）文档**

**加密接口**

**功能**：将8bit明文和10bit密钥作为输入，输出8bit密文。

**方法签名**：public String encrypt(String plainText, String key)

**参数：**

**plainText：**8bit明文。

**key：**10bit密钥。

**返回值：**加密后的8bit密文。

**解密接口**

**功能：**将8bit明文和10bit密钥作为输入，输出8bit密文。

**方法签名：**public String decrypt(String cipherText, String key)

**参数：**

**cipherText：**8bit密文。

**key：**10bit密钥。

**返回值：**解密后的8bit明文。

**错误处理接口**

**功能：**处理输入错误，确保数据格式正确。

**方法签名：**public boolean validateInput(String input, int length)

**参数：**

**input：**待验证的输入字符串。

**length：**期望的输入长度。

**返回值：**输入是否有效的布尔值。

**数据结构与算法说明**

**P-Box、S-Box等转换单元的实现：**详细描述P-Box和S-Box的实现方式，以及它们在加密和解密过程中的作用。

**加密算法流程**：描述加密过程的每一步，包括密钥扩展、初始置换、功能性变换、逆初始置换等。

**解密算法流程**：描述解密过程的每一步，与加密过程类似但顺序相反。

**单元测试与集成测试**

**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。

加密正确测试：输入8bit的明文和10bit的密钥，输出为8bit的密文。



解密正确测试：输入8bit的密文和10bit的密钥，输出为8bit的明文。



当明文、密钥或密文输入不为8、10、8bit时，会提示







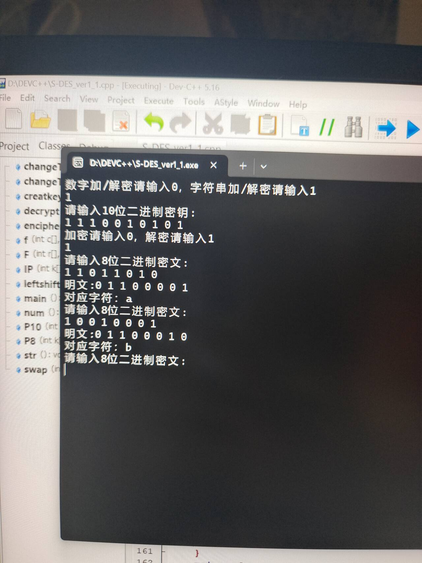
**第2关：交叉测试**

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。

我们的程序：

其他组同学的程序：



**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)。

加密过程：输入明文字符串和10bit的密钥，输出密文（转换成字符串或乱码）



解密过程：输入密文字符串和10bit的密钥，输出明文（转换成字符串或乱码）



**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。

正确测试：输入8bit的明文和密文，会显示对应的密钥。

如果找到密钥会显示暴力破解完成时间，找到的密钥数量和具体的值。

**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}](" \l ")，是否会出现选择不同的密钥[K\_{i}\ne K\_{j}](" \l ")加密得到相同密文[C\_n](" \l ")的情况？

是的，对于随机选择的一个明密文对可能有不止一个密钥key，因此也会出现选择不同密钥进行加密得到相同密文的情况。

**4. 接口文档**

**加密接口**

**输入参数：**

**plainText（String）：**8bit明文。

**key（String）：**10bit密钥。

**输出参数：**

**返回值（String）：**8bit密文。

**接口调用示例：**

String plainText = "01010101";

String key = "1111111111";

String cipherText = encrypt(plainText, key);

**解密接口**

**输入参数：**

**cipherText（String）：**8bit密文。

**key（String）：**10bit密钥。

**输出参数：**

**返回值（String）**：8bit明文。

**接口调用示例：**

String cipherText = "11001100";

String key = "1111111111";

String plainText = decrypt(cipherText, key);

**错误处理接口**

**功能：**处理输入错误，确保数据格式正确。

**方法签名：**public boolean validateInput(String input, int length)

**参数：**

**input（String）：**待验证的输入字符串。

**length（int）：**期望的输入长度。

**返回值：**输入是否有效的布尔值。

**接口调用示例：**

String input = "11111111";

boolean isValid = validateInput(input, 8);

**错误处理接口**

**错误代码与描述：**

1：输入长度不正确。

2：输入格式错误。

**异常处理逻辑：**在接口方法中通过抛出异常来处理错误情况。

**5. 高级功能与测试**

**扩展功能实现**

**ASCII字符串加密与解密：**

**实现细节：**

首先将ASCII字符串转换为二进制格式，然后分组进行加密或解密，最后将结果转换回ASCII字符串。

**示例：**

输入字符串“hello”，转换为二进制后分组加密，输出可能是不可读的字符串。

**暴力破解实现**

**破解算法：**

**描述：**

尝试所有可能的10bit密钥，对给定的密文进行解密，直到找到与已知明文匹配的结果。

**多线程优化：**

使用多线程技术并行尝试不同的密钥，以加快破解速度。

**测试结果与分析：**

展示暴力破解的效率，包括破解时间、破解成功的密钥数量和具体值。

封闭测试分析

**密钥唯一性分析：**

分析是否存在多个密钥能够对同一明文加密得到相同的密文。

**结论：**

通过测试发现，对于特定的明文和密文对，可能存在多个有效的密钥。

**不同密钥产生相同密文的可能性：**

分析选择不同的密钥进行加密得到相同密文的可能性。

**结论**：

理论上，由于S-DES密钥空间相对较小，存在不同密钥产生相同密文的可能性，但实际中发生的概率较低。

**6. 附录**

**术语表**

S-DES：简化的数据加密标准。

P-Box：置换盒，用于数据的重新排列。

S-Box：替换盒，用于数据的替换。

密钥扩展：将密钥扩展为加密过程所需的形式。

暴力破解：尝试所有可能的密钥来破解加密的方法。

**参考文献**

列出用于编写本开发文档和实现S-DES算法的参考书籍、论文和在线资源。