**CMSC203**

**作业 #3**

在[加密](https://en.wikipedia.org/wiki/Cryptography)学**中**，加密是编码消息或信息的过程，只有授权方才能访问，而未授权方不能访问。在加密方案中，预期的信息或消息（称为纯文本）使用加密算法（一种密码）进行加密，生成只有解密时才能读取的密文。

谜团机器是一个便携式[密码机](https://en.wikipedia.org/wiki/Cipher)和转[子搅乱](https://en.wikipedia.org/wiki/Rotor_machine)器的家庭[，成为](https://en.wikipedia.org/wiki/Nazi_Germany)纳粹德国的主要[加密系统](https://en.wikipedia.org/wiki/Cryptography)。 1932年12[月，波兰总参谋部密码](https://en.wikipedia.org/wiki/Polish_General_Staff)局在一名德国间谍提供的法国提供的情报材料帮助下，打破了该情报。

纳粹德国使用的谜机

二战期间

**分配说明**

编写一个Java程序，使用两种类似的方法加密和解密一个短语，每个方法都不安全，以现代标准。

第一种方法称为凯撒密码，是一个简单的"替换密码"，其中消息中的字符被替换字符替换。

第二种方法，由于乔万·巴蒂斯塔·贝拉索（b 1505，d 1581），使用一个关键词，其中单词中的每个字符指定消息中相应字符的偏移量，并根据需要环绕关键词。

**此分配测试的概念**

* 使用循环
* 字符串和字符处理
* ASCII 代码

**类**

**数据管理器类** **=** **加密管理器.java**

* 实现此文件中指定的每种方法。 提供的版本将在控制台中打印错误消息，因为它们只是骨架。
* 每个方法是静态的，因此无需创建数据管理器的实例。
* 使用简单的描述记录每个方法，并使用在线注释 （//...） 用简单描述和您的姓名对类进行记录。（对于每个文档字符串来说，只需一个简短的句子片段即可。
* 方法如下所述。
* 公共静态布尔字符串（字符串纯文本）;

此方法根据字符串和字符串的字符确定字符串是否在 ASCII 代码LOWER\_BOUNDUPPER\_BOUND范围内。 参数纯文本是要加密的字符串。 如果所有字符都在允许的范围内，则该方法返回 true;如果任何字符位于外部，则返回 false。

* 公共静态字符串加密Caesar（字符串纯文本，int键）;

此方法根据凯撒密码加密字符串。 整数键指定偏移量，纯文本中的每个字符都替换为指定距离它的距离的字符。 参数纯文本是要加密的大写字符串。参数键是一个整数，用于指定每个字符的偏移量。 该方法返回加密的字符串。

* 公共静态字符串解密（字符串加密文本，int 密钥）;

此方法根据凯撒密码解密字符串。 整数键指定偏移量，加密文本中的每个字符都替换为其前的字符"偏移"字符。 这是加密的Caesar方法的反面。 参数加密文本是要解密的加密字符串，密钥是用于加密原始文本的整数。 该方法返回原始纯文本字符串。

* 公共静态字符串加密贝拉索 （字符串纯文本， 字符串贝拉索斯特）;

此方法根据贝拉索密码对字符串进行加密。 纯文本中的每个字符根据 bellasoStr 中相应字符的 ASCII 值进行偏移，该值重复以对应于纯文本的长度。该方法返回加密的字符串。

* 公共静态字符串解密贝拉索（字符串加密文本，字符串贝拉索斯特）;

此方法根据贝拉索密码解密字符串。 加密文本中的每个字符都替换为与 bellasoStr 中的字符对应的字符，该字符重复以对应于纯文本的长度。 这是加密贝拉索方法的反面。参数加密文本是要解密的加密字符串，bellasoStr 是用于加密原始文本的字符串。 该方法返回原始纯文本字符串。

* 如果您希望使逻辑更易于遵循，请添加其他方法。

**GUI 驱动程序类** **=**  **（提供）**

* 提供图形用户界面（GUI） . 确保 GUI 将编译并运行您的方法。如果加密管理器.java 中的方法标头不完全采用指定的格式，则 GUI不会编译。 首次运行应用程序时，所有方法都将引发异常，这些异常将由 GUI 捕获并在控制台中打印出来。
* 不要修改 GUI。
* GUI 负责大写输入字符串。

**JUnit 测试/测试驱动程序**

* 实现方法后，运行测试驱动程序。 确保驱动程序文件产生以下输出，如RED所示（当然，运行测试驱动程序时输出不会为红色）：

"这个测试应该成功" 在边界里吗？真实

"这个测试应该失败， 因为 [ 超出范围" 它在边界中吗？假

"此测试应因小写字母而失败"是否位于边界中？假

凯撒密码"这是另一个测试"应返回"WKLV#LV#DQRWKHU#WHVW"：WKLV#LV#DQRWKHU#WHVW

贝拉索密码的"这是另一个测试"应该返回"WU\VR9F#N！RF88U-'HED'：WU+VR9F#N！RF88U -'HED

凯撒解密"WKLV#LV#DQRWKHU#WHVW"应返回"这是另一个测试"：这是另一个测试

贝拉索解密"WU+VR9F#N！RF88U -"HED"应返回"这是另一个测试"：这是另一个测试

* 运行 JUnit 测试文件（提供）。 确保 JUnit 测试全部成功。
* 包括加密测试.java和加密管理器测试.java 与您的提交的其余部分。

**分配详细信息**

第一种方法称为凯撒密码，是一个简单的"替换密码"，其中消息中的字符被替换字符替换。 替换根据整数键完成，该键指定替换字符的偏移量。 例如，键为 3 的字符串 ABC 将被 DEF 替换。

如果键大于我们想要考虑的字符范围，我们通过从键中减去范围来"环绕"，直到键在所需范围内。 例如，如果我们有一个范围从空格 （'）到 '\_'（即 ASCII 32 到 ASCII 95），并且键是 120，我们注意到 120 超出范围。 因此，我们从 120 中减去 95-32+1+64，给出56，在 ASCII 中，字符为 "8"。8 如果键更高，我们可以一遍又一遍地从键中减去范围，直到键在所需范围内。

由于我们指定的范围不包括小写字母，GUI（提供）会将字符串更改为大写。 您可以在一个位置找到[ASCII http://www.asciitable.com/，](http://www.asciitable.com/)或者在互联网上找到许多其他地方。

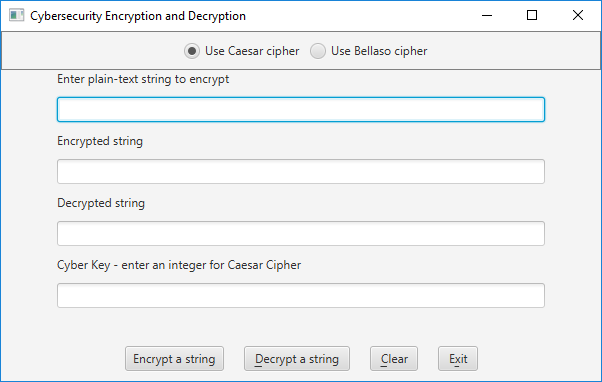
第二种方法，由于乔万·巴蒂斯塔·贝拉索（b 1505，d 1581），使用一个关键词，其中单词中的每个字符指定消息中相应字符的偏移量，并根据需要环绕关键词。

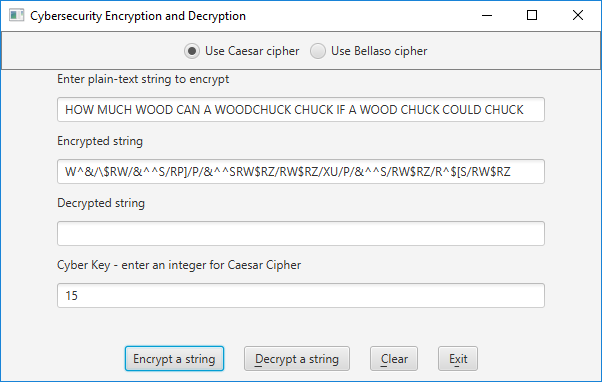
因此，对于字符串 ABCDEFG 和关键词 CMSC，关键词首先扩展到字符串的长度，即 CMSCCMS。 然后 A 替换为"A"偏移量为"C"，即 ASCII 65×67×132。 字符的范围也指定，我们and 再次说''到'\_'（即，ASCII 32到ACII 95）。. 范围为 95-32×1×64。 在我们的示例中example，偏移量通过按范围减少 132 来"包装"，直到它是允许的范围。 132 在加密阶段调整为 132-64+68，, or character ‘或字符"D"。’ 然后，相同的逻辑应用于纯文本"B"的第二个字母，由关键词"M"的第二个字母移动。 这将导致字符"O"作为加密阶段中的第二个字母，等等。 在每种方法中，如果生成的整数大于 95（我们范围的顶部），则整数将"环绕"，以便它保持在指定范围内。 其结果是[DOVGHSZ].

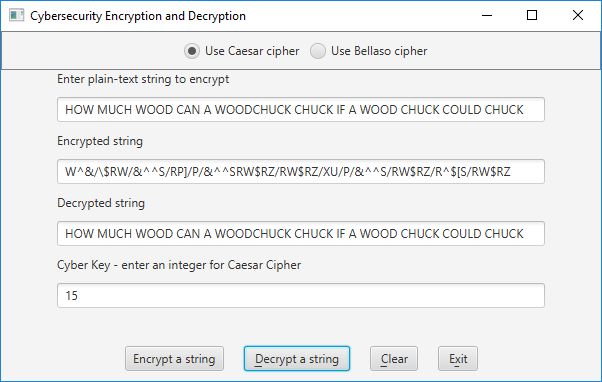
您的程序将实现在文件"CryptoManager.java"中指定的几种方法。 提供图形用户界面以及测试文件，您应该使用该界面来确保方法正常工作。 请务必完全遵循命名，因为测试将不起作用。

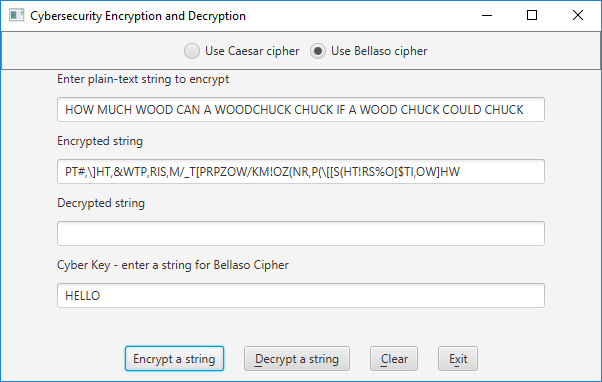
Java 有几个功能，我们还没有在类中介绍。 只需遵循本文档和文件 CryptoManager.java 中指定的语法。首先，所需的方法是"静态的"，这意味着即使尚未创建实例，它们也可用于类。 例如，要调用静态方法，"public 静态无效myMethod（）;" 语法为( CryptoManager.myMethod （）;" 在此 pr oject 中可能有用的另一个功能是字符串上的方法 charAt （i），它返回字符串位置 i 处的字符（基于零）。 所以 [这字符串].charat （3）;将返回字符的。

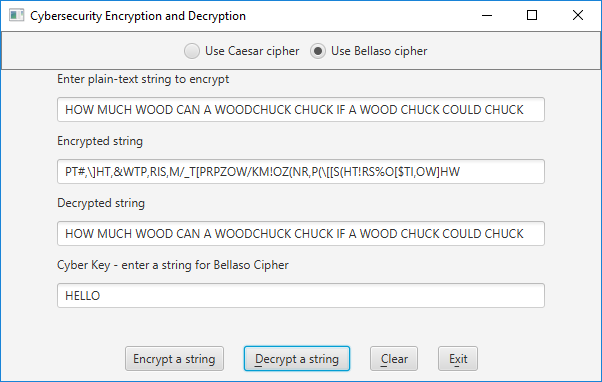
**例子**











**交付**

* 为 CryptoManager.java 中指定的每个方法输入伪代码。 您的伪代码应该是英语和 java 之间的一部分。 无需阐明变量声明等的所有细节，但同样，伪代码需要有足够的细节，使有能力的Java程序员能够实现它。
* 在
* 至少两次凯撒密码测试
* 贝拉索密码至少两次测试
* 至少一个字符串将失败，因为它具有可接受的字符之外的字符。
* 学习经验：突出您从本项目工作的经验教训和经验教训。
* 你学到了什么？
* 你在挣扎什么？
* 在下次项目上，您将采取哪些不同的做法？
* 包括您成功的项目哪些部分，以及您未成功的项目部分（如果有）。
* GitHub：在存储库中（请参阅实验 1），上传最初在 Blackboard 中为项目提供的文件。 完成设计和编程后，请上传 Word 文件和 java 文件。 您需要将这些文件作为目录的内容上载，以便将来的上载可以分开进行。 获取并提交 GitHub 存储库的屏幕截图。

**笔记：**

* 正确的命名约定：应命名除 0 和 1 以外的所有常量。常号应该是所有大写，变量名称应该以小写开始，但后续单词应该是标题大小写。变量和方法名称应描述变量或方法的作用。应避免单字母名称。
* 文档：所有编程项目的文档要求是程序顶部的一个块注释，其中包含用于开发项目的课程名称、项目编号、名称、日期和平台 /编译器。如果使用未创建的任何代码或特定算法，应在相应的注释块中引用其源。如有必要，应提供其他意见，以澄清程序。

缩进：它必须在整个程序中保持一致，并且必须反映控制结构

**提交详细信息**

提交以下文件：

* 名称为 FirstInitialLastName\_Assignment3.docx 的 Word 文档应包括：
* 在 CryptoManager.java 中指定的每个方法的伪代码。
* 测试计划
* 根据测试计划筛选 Eclipse输出的快照
* GitHub 提交的屏幕快照
* 经验 教训
* 检查列表
* zip 文件将仅包含 .java 文件，并将命名为：FirstInitialLastName\_Assignment3\_Moss.zip。 **此 .zip 将没有任何文件夹 - 只有 .java 文件。**

下面是作业 3 的示例：

R\_Brown\_Assignment2. docx

R\_Brown\_Assignment2\_Moss.zip [仅包含以下内容的压缩文件]

加密管理器. java

加密测试. java

加密管理器测试. java

**测试计划**

使用至少3个测试用例测试程序。**确保测试涵盖所有可能的情况。**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 输入文本 | 输入键 | 加密（方法1） | 加密（方法2） | 解密（方法1） | 解密（方法2） |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**分级鲁比克**

参见附件： CMSC203 分配 3 Rubric\_Summer20.xlsx

作业 3检查列表

|  |  |  |  |
| --- | --- | --- | --- |
| **#** |  | **Y/N** | **评论** |
|  | **分配文件：** |  |  |
|  | * 第一初始姓氏 \_分配 # \_Moss.zip | **<是或否>** |  |
|  | * FirstInitialLastName\_Assignment#.docx/.pdf | **<是或否>** |  |
|  | * 源 java 文件 | **<是或否>** |  |
|  | **程序编译** | **<是或否>** |  |
|  | **程序运行时具有与测试计划相关的所需输出** | **<是或否>** |  |
|  | **文档文件：** |  |  |
|  | * 综合测试计划 | **<是或否>** |  |
|  | * 测试计划中列出的每个测试用例的屏幕截图 | **<是或否>** |  |
|  | * 提交分配# 的 GitHub帐户的屏幕截图（如果需要） | **<是或否或不/A>** |  |
|  | * UML 图（如果需要） | **<是或否或不/A>** |  |
|  | * 算法/伪代码（如果需要） | **<是或否或不/A>** |  |
|  | * 流程图（如果需要） | **<是或否或不/A>** |  |
|  | * 经验 教训 | **<是或否>** |  |
|  | * 清单已完成并包含在文档中 | **<是或否>** |  |