## File & Text Encryption Application User Manual

Table of Contents for Encryption Application

Contents

[File & Text Encryption Application User Manual 1](#_Toc196574705)

[1. Introduction 3](#_Toc196574706)

[1.1. Vigenère Cipher 3](#_Toc196574707)

[1.2. Overview 3](#_Toc196574708)

[1.3. Algorithm Details 3](#_Toc196574709)

[2. Vigenère Text Cipher 3](#_Toc196574710)

[2.1. Steps 4](#_Toc196574711)

[2.2. Key Methods: 4](#_Toc196574712)

[3. Vernam Cipher 4](#_Toc196574713)

[3.1. Key Concepts: 4](#_Toc196574714)

[3.2. Key Methods: 4](#_Toc196574715)

[4. Graphical User Interface (MainApp) 4](#_Toc196574716)

[4.1. Text Encryption/Decryption Section: 5](#_Toc196574717)

[4.2. File Encryption/Decryption Section: 5](#_Toc196574718)

[4.3. Methods: 5](#_Toc196574719)

[4.4. Usage Instructions: 6](#_Toc196574720)

[4.5. Error Handling: 6](#_Toc196574721)

# Introduction

This document serves as a user manual and technical report for the Java-based File & Text Encryption Application. It covers:

* An overview of the application’s features.
* Detailed explanations of the Vigenère Cipher and Vernam Cipher algorithms.
* Example code snippets for encryption and decryption (text and file-based).
* A walkthrough of the Swing GUI (MainApp) structure and usage.

## Vigenère Cipher

## Overview

The Vigenère Cipher is a classical polyalphabetic substitution cipher. It encrypts alphabetic text by shifting each letter of the plaintext by an amount determined by a repeating key. This implementation has been made to handle non-letter characters which includes special handling for digits in the key.

* Classic Vigenère Cipher but adapted to handle non-letter characters (keeps them unchanged).
* Special handling if the key contains digits (converts digits to letters, e.g. ‘0’ -> ‘A’, ‘1’ -> ‘B’).

## Algorithm Details

**Key Preparation:** The key is repeated or truncated to match the length of the plaintext.

**Character Conversion:**

* Plaintext and key characters are converted to uppercase.
* Digits in the key (0–9) map to letters A–J.

**Encryption:**

* For each letter P in the plaintext and corresponding key letter K, compute:



* Non-letter characters remain unchanged.

**Decryption:**

* For each cipher letter C and key letter K, compute



# Vigenère Text Cipher

The Vigenère Cipher is also applied to text, not files in the application. Using a customized method to encrypt and decrypt text by shifting each letter based on a key. Non-alphabetic characters (like spaces and punctuation remain unchanged.

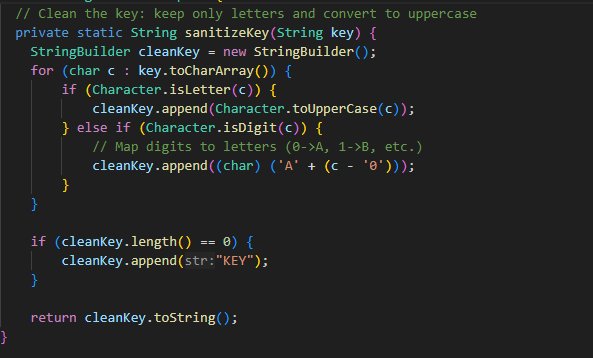


Figure . Sanitizing text before encryption

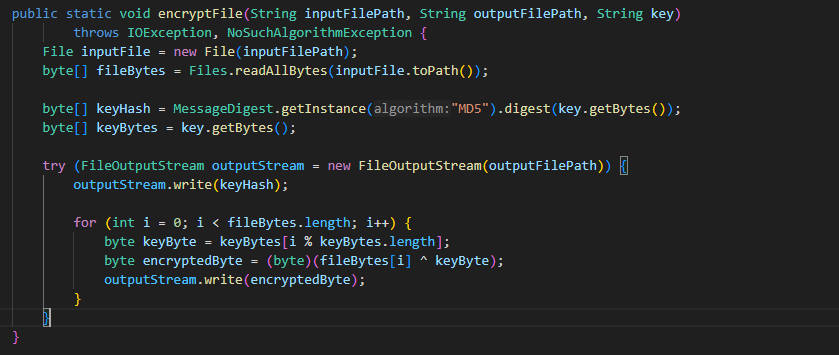


Figure . vigenere file encryption code

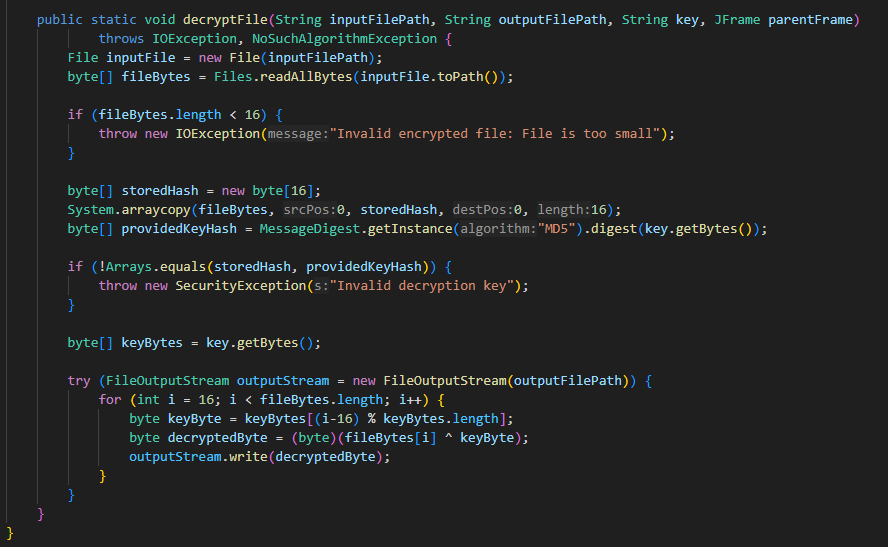


Figure .vigenere file decryption code

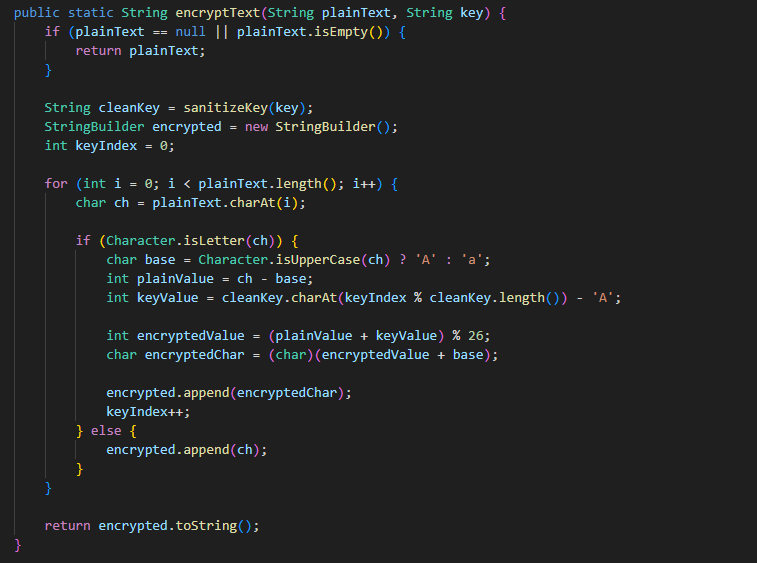


Figure . Vigenere encryption text code

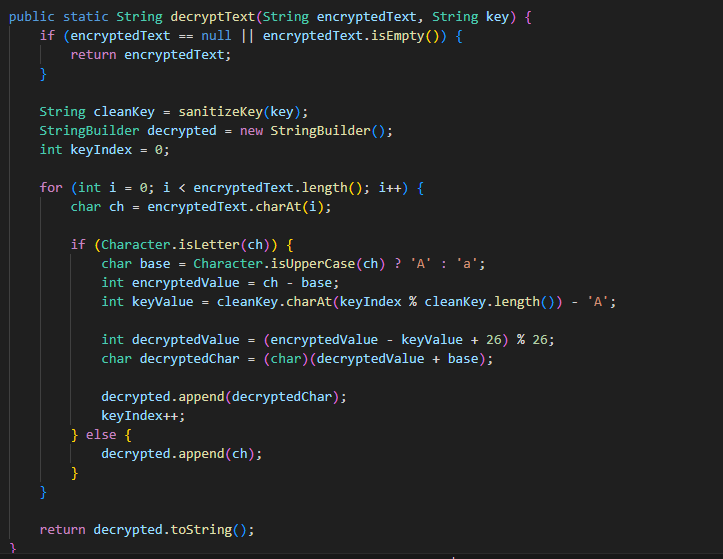


Figure .Vigenere decryption text code

## Steps

* **Key Sanitization:** Key is cleaned to remove nonalphabetic characters and digits, then mapped to letters/ (e.g., ‘0’ becomes ‘A’).
* **Encryption:** Each letter in plaintext is shifted by corresponding letter in the key.
* Key repeats as required to cover the full length of the plaintext.
* **Decryption:** Reverses the shift.

## Key Methods:

* sanitizeKey (String key)
* encryptText (String plaintext, String key)
* deryptText (String plaintext, String key)

# Vernam Cipher

The Vernam Cipher (a stream cipher variant often called the One-Time Pad when key is truly random) uses a byte-wise XOR between plaintext and key. In this implementation, the key repeats if shorter than the data.

A screen shot of a computer program

AI-generated content may be incorrect.

Figure . Vernam file encryption code

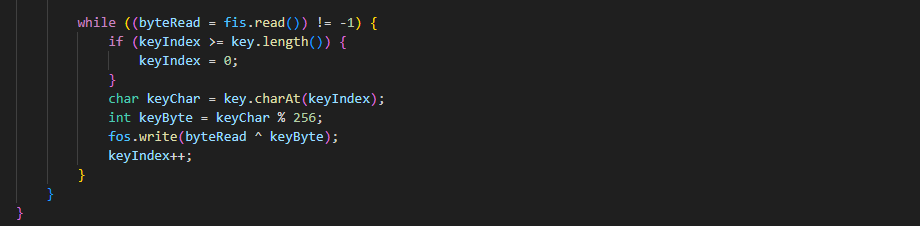
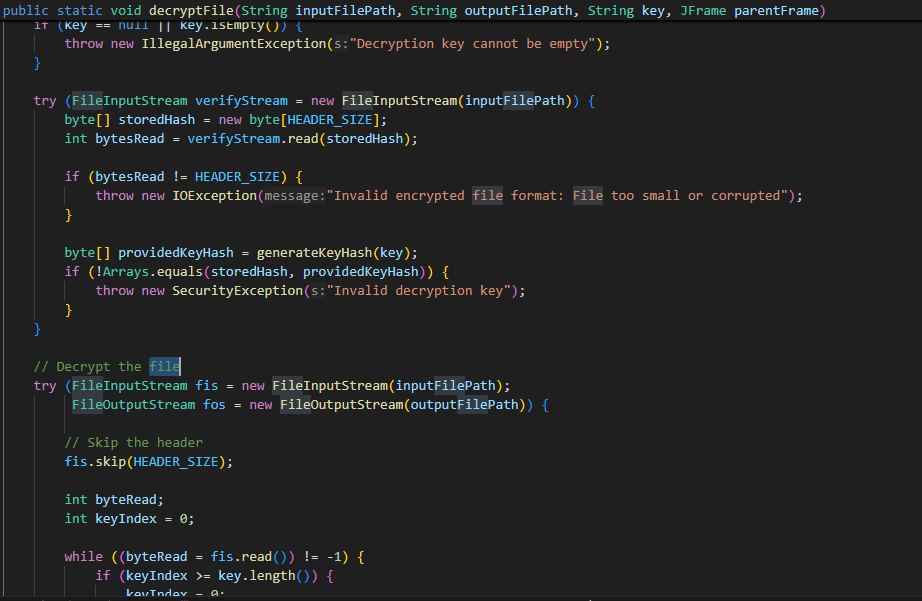


Figure . vernam file decryption code

A computer screen shot of text

AI-generated content may be incorrect.

Figure . Vernam text encryption code

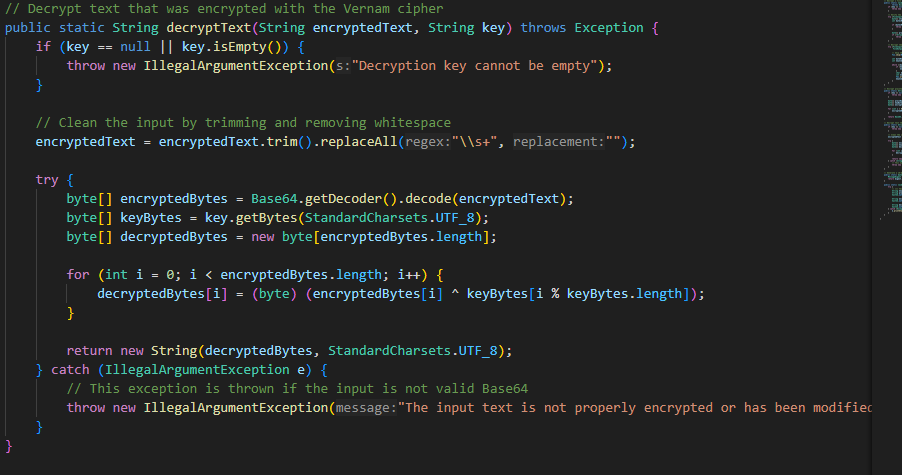


Figure . Vernam text decryption code

## Key Concepts:

* **Key:** The key used for both encryption and decryption must be kept secure. If key is repeatable or predictable, the security of the encryption is compromised.
* **XOR Operation:** Each byte of plaintext is XORed with the corresponding byte of the key to produce the ciphertext.

## Key Methods:

* encryptFile (String inputFilePath, String outputFilePath, String key)
* decryptFile (String inputFilePath, String outputFilePath, String key, JFrame parentFrame)
* encryptText (String plainText, String key)
* decryptText (String encryptedText, String key)
* generateKeyHash (String key)

# Graphical User Interface (MainApp)

MainApp is a graphical user interface (GUI) application built using Java Swing. It provides functionality for both text encryption/decryption and file encryption/decryption using multiple encryption algorithms. The application supports:

* Vigenère Cipher
* Vernam Cipher
* Columnar
* Own Algorithm
* Bonus Algorithm

It allows users to encrypt or decrypt text and files using a user-provided key and selected algorithm. The application includes real-time status updates and error handling for seamless user interaction

## Text Encryption/Decryption Section:

* **Text Algorithm Selector (JComboBox):** A drop-down menu allowing users to choose between the encryption algorithms.
* **Text Key Field (JTextField):** A text field entering the encryption or decryption key.
* **Input Text Area (JTextArea**): A large text area where users can input plain text for encryption.
* **Output Text Area (JTextArea):** A read-only text area where the encrypted/decrypted text will be displayed.
* **Encrypt/Decrypt Buttons (JButton):** Buttons that trigger encryption or decryption operations for the input text based on the selected algorithm.

## File Encryption/Decryption Section:

* **File Algorithm Selector (JComboBox**): A dropdown menu allowing users to choose between two file encryption algorithms:
* **File Key Field (JTextField):** A text field for entering the encryption/decryption key for file operations.
* **Input File Field (JTextField):** Displays the selected input file path for encryption or decryption.
* **Output File Field (JTextField):** Displays the selected output file path for saving the encrypted/decrypted file.
* **Browse Buttons (JButton):** Buttons that open file dialog boxes to select input and output files.
* **Encrypt/Decrypt Buttons (JButton):** Buttons that trigger encryption or decryption operations for files based on the selected algorithm.
* **Status Area (JTextArea):**

Displays real-time status messages about the encryption/decryption process, including success and error messages.

## Methods:

|  |  |
| --- | --- |
| Method | Description |
| createTextPanel() | Builds text encryption/decryption panel. |
| createFilePanel() | Builds file encryption/decryption panel. |
| createStatusPanel() | Builds status area panel. |
| selectInputFile() | Open file chooser for input file. |
| selectOutputFile() | Open file chooser for output file. |
| encryptText() | Encrypt text using selected algorithm. |
| decryptText() | Decrypt text using selected algorithm. |
| encryptFile() | Encrypt selected input file. |
| decryptFile() | Decrypt selected input file. |
| showError() | Display error messages. |
| validateFileInputs() | Validate file operation inputs. |

## Usage Instructions:

**Text Encryption/Decryption:**

* Select the encryption algorithm from the dropdown.
* Enter the text to encrypt in the input area.
* Provide the encryption/decryption key.
* Click the "Encrypt Text" or "Decrypt Text" button to perform the operation.

**File Encryption/Decryption**:

* Select the encryption algorithm from the dropdown.
* Select the input file using the "Browse" button.
* Provide the output file path.
* Enter the encryption/decryption key.
* Click the "Encrypt File" or "Decrypt File" button to perform the operation.

**Status Updates:**

* View the status of operations (success or failure) in the status area.

## Error Handling:

* **Text Operations:** Errors shown in Status Area and Dialog Box.
* **File Operations:** Error if missing file paths or invalid key.

## Team Member Responsibilities

| **Name** | **Responsibility** |
| --- | --- |

|  |  |
| --- | --- |
| M. Mathebula | Vernam Cipher, GUI File Handling, Vigenère Cipher |
| S. Sofnia (apologies if I spelled surname wrong) | Add duties here |