Indian Institute of Information Technology, Nagpur

Project Report On: Vigenère Cipher Implementation in C++

Made By:

BT23CSE114 Aditya Aragkar

Guided By:

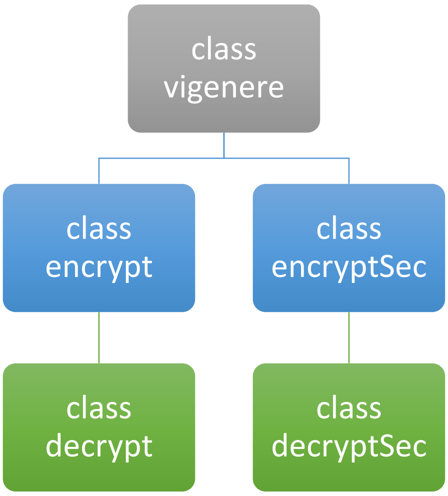
Dr. Milind Penurkar

1. Introduction

This report details the implementation of a Vigenère cipher for secure encryption and decryption of text files. The Vigenère cipher is a method of encrypting alphabetic text by using a simple form of polyalphabetic substitution. This program allows users to encrypt and decrypt files using both standard and secure methods, providing a user-friendly interface for managing sensitive information.

2. Classes and Objects

The implementation consists of several classes that encapsulate the functionality of the Vigenère cipher:



a. vigenere

- Purpose: Serves as the base class for creating the Vigenère table and managing the alphabet.
- It is an abstract class as it contains one pure virtual function named table() and as a result its object can't be created.
- Attributes:
    - char vigenere[27][27]: A 2D array representing the Vigenère table.
    - aditya<char> alphabets: A custom dynamic array to store the alphabet. It is made using templates. Its Definition are in aragkar.h header file.
    - int alpha: Tracks the ASCII value of characters.

b. encrypt

- Purpose: Inherits from vigenere and implements the basic encryption functionality.
- Key Methods: Includes methods for constructing the Vigenère table and encrypting file contents.

c. encryptSec

- Purpose: Inherits from vigenere and provides a secure encryption method.
- Key Methods: Similar to encrypt, but incorporates an additional layer of security using a secondary keyword.

d. decrypt

- Purpose: Inherits from encrypt and implements the decryption functionality.
- Key Methods: Reads encrypted files and decrypts their contents.

e. decryptSec

- Purpose: Inherits from encryptSec and provides secure decryption.
- Key Methods: Decrypts files that were encrypted using the secure method.

3. Functions

The program includes several key functions that facilitate its operation:

a. **vigenere**

   - list(): Initializes the alphabet list for the Vigenère cipher. This is used for simple encryption.
   - listSec(string key): Populates the alphabet list with a keyword while avoiding duplicates. This is used for secure encryption.
   - vigenere_border(): Sets up the borders of the Vigenère table.
   - table(): A pure virtual function in vigenere class. Constructs the Vigenère table based on the alphabet.

b. **encrypt**

   - table(): the function is defined here for construction of Vigenère cipher table.
   - encryp(): Reads a file, encrypts its contents, and writes back the encrypted content.

c. **decrypt**

   - decryp(): Reads an encrypted file, decrypts its contents, and writes back the decrypted content.

d. **encryptSec**

   - table(): this function is defined here for construction of Vigenère cipher table not similar to the table() function in class encrypt. This function uses different logic for cipher table construction.
   - encryp(): Reads a file, encrypts its contents, and writes back the encrypted content.

e. **decrypt**

   - decryp(): Reads an encrypted file, decrypts its contents, and writes back the decrypted content.

4. Object-Oriented Programming Principles

The implementation adheres to key principles of Object-Oriented Programming (OOP):

- **Encapsulation:** Each class encapsulates its data and methods, providing a clear interface for interaction. For example, the vigenere class manages the Vigenère table and alphabet, while the encrypt and decrypt classes handle encryption and decryption processes.
- **Inheritance:** The use of inheritance allows for code reuse and the creation of specialized classes. For instance, encryptSec extends the functionality of encrypt by adding secure encryption features.
- **Polymorphism:** The table() method in the vigenere class is declared as a pure virtual function, allowing derived classes to provide their specific implementations. This enables flexibility in how the Vigenère table is constructed.
- **Templates:** The use of templates in the implementation, specifically in the header file *aragkar.h*, allows for the creation of generic classes that can operate on any data type. The instantiation of *aditya<char> alphabets* demonstrate how templates can be used to create a class that specifically handles character data, while the same template can be reused for other data types, enhancing code flexibility and reducing redundancy.

5. Conclusion

The implementation of the Vigenère cipher provides a robust solution for secure file encryption and decryption. The program is designed to be user-friendly, allowing easy selection of operations. The use of secure keywords enhances the encryption process, ensuring that sensitive information remains protected.