Documentation

1. Team Name & Members: DYLO || Manuel Carrillo
2. Project Information:
   1. Problem to solve:
      1. The problem I’m going to solve is providing users the ability to use a Vigenere Cipher without any confusion. Most programs are difficult to understand and require external research for users and programmers.
      2. I aim to solve this problem by providing detailed instructions on front and back end while making the program as condensed as possible
      3. Calculation and algorithm explanation
      4. The objectives of this program is to ask user to type two entries into the terminal; one for the message and one for the key. After receiving valid entries, the program will then run the necessary algorithms to providing the correct message. This program is interacting with the user by asking them to input the values to use for the algorithms and then reading back the message accordingly.
      5. Discrete structures is implemented in this program as mathematical equations are used in conjunction with arrays to form a table of rows and columns
      6. The limitations of the program stem from the length of the user’s input as the larger the value the more time is needed for calculations to complete as loops are used to iterate through each of the values
      7. A recommended improvement would be to use recursive logic as it would allow the program to reuse values generated instead of iterating through each one unnecessarily
3. Flowchar Or Pseudocode
   * 1. Pseudocode

Team DYLO’s Vigenere Cipher Project

**Team Name**: DYLO

**Members**: Manuel Carrillo

**Project Information and Details**:

In this project, my team implemented the Vigenere Cipher, a cryptographic method used for encrypting and decrypting messages. The Vigenere Cipher enhances the security of communication by replacing letters in the message with different letters based on a keyword. My objective is to develop a program that allows users to encrypt and decrypt messages using the Vigenere Cipher.

To achieve this, I have implemented several solutions in my program. First, I provide a menu-based interface where users can choose between encryption, decryption, or quitting the program. When encrypting a message, the user is prompted to enter the message and a key. The program then generates the encrypted message by applying the Vigenere Cipher algorithm. Similarly, during decryption, the user enters an encrypted message and the key used for encryption. The program then performs the decryption process to recover the original message.

The core calculations and algorithm implementation in my program revolve around the Vigenere table. The Vigenere table is a 2-dimensional array that represents the letters of the alphabet. I generate this table by populating it with characters based on the alphabet. The encryption process involves finding the row and column indices in the Vigenere table based on the characters of the message and the key. Using these indices, I retrieve the corresponding character from the table to generate the encrypted message. The decryption process follows a similar approach, but in reverse. I find the row index based on the encrypted character and the column index based on the key character. Then, I retrieve the original character from the Vigenere table to decrypt the message.

The program interacts with the user by displaying messages and requesting inputs. I prompt the user to choose between encryption, decryption, or quitting the program. I provide clear instructions for entering the message and the key. After performing the encryption or decryption process, the program displays the resulting encrypted or decrypted message to the user.

In terms of implementing discrete structures, my program utilizes arrays and loops. I use a 2-dimensional array to represent the Vigenere table, storing the mapping of characters. Loops are employed to iterate over the characters of the message and the key, perform necessary calculations, and access the Vigenere table. These discrete structures allow us to efficiently implement the Vigenere Cipher algorithm in my C++ program.

While my program successfully implements the Vigenere Cipher, it does have some limitations. Currently, it only supports uppercase alphabets for both the message and the key. Special characters and spaces in the message are not handled, and the program lacks error handling for invalid inputs.

To improve my program, I recommend implementing the following enhancements. First, I can extend the program to support lowercase alphabets and other characters by expanding the Vigenere table. This would increase the flexibility and usability of the program. Second, I should implement error handling to validate user inputs and provide informative error messages. This would help users understand and correct any input mistakes. Lastly, enhancing the program to handle special characters and spaces in the message would make it more robust and adaptable to different types of messages.

In conclusion, my team has developed a C++ program that implements the Vigenere Cipher. Through a menu-based interface, users can encrypt and decrypt messages using the Vigenere Cipher algorithm. The program utilizes discrete structures such as arrays and loops to efficiently perform calculations and access the Vigenere table. While the program has some limitations, I have provided recommendations for improvement to enhance its functionality and usability. By implementing these enhancements, I can create a more versatile and error-resilient Vigenere Cipher program.