CIS-7 Project Documentation

Saim Ahmed, Syed Moiz, Emma Wuysang

Case 3 (Vigenere Cipher Decryption)

Sunday, December 8

Advisor: Kasey Nguyen, PhD

Project Information & Details :

1. **What problems are you solving in this project?**
   This project which highlights the Vigenere Cipher Decryption works upon encrypting and decrypting a specific user given text. The main obstacles we worked on solving were applying the substitution to the alphabet, accounting for the specific ASCII conversions, and addressing non-alphabetical letters in the given text for encryption.

2. **What solutions are you implementing in the project?**
   In our program, we decided on solutions that focused on utilizing specific computational methods, specifically modular arithmetic in terms of addressing the range alphabetical shifts for the plaintext to encrypted and decrypted; working with programming cryptography, case conversions to take into consideration of uppercase and lowercase lettering and finally, using substitution/permutations in the solving process for the actual conversions between the text.

3. **Provide explanation of calculations and algorithm implementation.**

   Ei = (Pi + Ki) mod 26 ,

   which is what models the conversion in the program is seen in our encryption function using “((plainNum + keyNum) % 26).”

   This is significant primarily due to it representing the main formula for applying Vigenere’s Cipher. It works by staying with the alphabetical range of 26, and computing the specific rotation to the encrypted alphabet.

   Ei = (Pi + Ki + 26) mod 26 ,

   similarly to the formula above,

   The function is primarily the same, however instead of encrypting, this focuses on the decryption portion of the program. It is seen in that specific function using “((ciphernum - keyNum +26)% 26)”. The mod 26 is pivotal to the program as stated above, in order to remain the position within the given alphabet. The addition in 26 to this formula is due to the handling of potential negative differences and retrieving the specific data for the original character before encryption.

   Encryption :

   The function works with the initialized plaintext and key from main, and is implemented to utilize a for loop for all alphabetical letters, and creating the conversion for plaintext/keywords to the corresponded numbers with ASCII arithmetic. This is seen with the ternary operators which handle the arithmetic from the key to plaintext.

For example, if the plaintext's first letter was H and the key's first letter was K, it would shift since (7+10) mod 26 is equivalent to 17 which is R. However, it is also significant to note that the keyIndex% key.size() is what accounts for when the key is shorter than the given plaintext.

Decryption :

While similar in process to the encryption function, the modular arithmetic is the primary difference since it is attempting to retrieve the original ascii value to the letter. As explained above in the formulas. All of these are pushed in order to register to the string, this is looped for each character within the plaintext until done.

4. **What is the program objectives? Explain how your program is interacting with the user and its purpose.**
   The primary function of this program was to enable the user to enter any given plaintext and a provided key, that would register an encrypted message, with the ability to also decrypt a message. This specific project is focused on the user's input, and objective to display an accurate cipher utilizing ASCII. An implementation of coding with cryptography was a general focus for the entirety of the program along with properly implementing the modular arithmetic/formula.

5. **How is discrete structures implemented in the C++ program?**
   As mentioned in question 2, a variety of discrete structures topics are utilized. From the modular arithmetic, finite sets, ternary operations, applying strings for sequences, properly indexing, and mapping plaintext to the ciphered text; this program focuses on implementation for those specific subjects with user-given input for plaintext and decryption/encryption.

6. **What are the limitations of the program?**

   At first, a limitation that existed was accounting for non-alphabetical letters within the plaintext. There was also whether the plaintext included numbers or the length of the text. We decided to treat the uppercase and lowercase separately instead of making it all uppercase or lowercase, so there could be potential issues in handling there. Also, the program does not account for if the user has varied letters/spaces in the keyword.

7. **Provide recommendations on improving the limitations of the program.**

   We decided to implement an if statement for converting only the letters that are considered an alphabetical letter. There was not a specific cap we put on the length of text, though numbers were not accounted for, the isDigit() could also be implemented. A possible recommendation could just be switching each character to uppercase/lowercase and keeping it that way. Implement further lines of code that take into consideration if

there are anomalies for the variable key when later converting, or return the message of not accepting the text

Pseudocode :

Write a function for encryption that uses strings.
Use the given plaintext and key variables from main.
Initialize an empty string for encrypted text.
Initialize the number key to integer 0.
Initialize the index for key to integer 0.
Initialize the plain text number to integer 0.
initialize an encryption char variable.

For each character inside the length of the plaintext,
If the character isn't alphabetical,
leave the non-alphabetical letter in the encryptedText and continue.
If the plaintext letter is uppercase, subtract the ASCII value from the letter 'A', but if lowercase, subtract 'a' to represent the number in the alphabet.
Assign that value to the plain text number.
Use the value of the index within the key for determining the specific character, for cases that the value of the index is not in the range for the key, utilize a modulus to wrap the key from the beginning.
Then, if the key is uppercase, convert to a number and subtract using 'A', but if not uppercase, use 'a'.
Assign value to the number key.
If uppercase, add the plain text number and the key number and finally the modulus of 26 to stay within range. To convert back to a letter, add 'A'.
If lowercase, then add 'a' instead.
Assign the value to the encryption char variable.
Put the encryption char variable into the encrypted text string.
Increment the key index.
Return the encrypted text string after going through the letters of plaintext.

Write a function for decryption that uses strings.
Use the given ciphertext and key variables from main.
Initialize an empty string for decrypted text.
Initialize the cipher number to integer 0.
Initialize the key number to integer 0.
Initialize the index for key to integer 0.
initialize a decryption char variable.

For each character inside the length of the ciphertext,
If the character isn't alphabetical,
leave the non-alphabetical letter in the decryptedText and continue.
If the ciphertext letter is uppercase, subtract the ASCII value from the letter 'A', but if lowercase, subtract 'a' to represent the number in the alphabet.
Assign that value to the cipher number.
Use the value of the index within the key for determining the specific character, for cases that the value of the index is not in the range for the key, utilize a modulus to wrap the key from the beginning.
Then, if the key is uppercase, convert to a number and subtract using 'A', but if

not uppercase, use ‘a’.

Assign value to the number key.

If uppercase, subtract the cipher text number and the key number, add 26 to hold the positive values and finally the modulus of 26 to stay within range. To convert back to a letter, add ‘A’. If lowercase, then add ‘a’ instead.

Assign the value to the decrypted char variable.

Put the encryption char variable into the decrypted text string.

Increment the key index.

Return the decrypted text string after going through the letters of cipher text.

In the main function,

Initialize the string variables plaintext and a key.

Using do-while loop{}
Request PLAINTEXT input
Write user input to the plaintext variable.

Request KEY input
Write user input to the key variable.

Call the function for encryption that takes in the two string variables, plaintext and key.

Read out the input of the given ciphertext.

Call the function for decryption that takes in the two string variables, ciphertext and key.

Read out the input of the given decrypted text.

Repeat action until (user input is blank).

Return a zero to conclude main.