# **CIS7 Project Documentation Guide**

In the documentation, provide at least 2 pages (single-space) that contains the following components of your course project:

1. Team name, members.

* Joash
* Nathalie
* Danielle Fernandez
  + <https://github.com/koa2019/vigenereCipher_cis7_discreteStruct_finalPrjt>

2. Project Information and details: (30 points)

* What problems are you solving in this project?
* What solutions are you implementing in the project?
* Provide explanation of calculations and algorithm implementation.
* What is the program objectives? Explain how your program is interacting with the user and its purpose.
* How is discrete structures implemented in the C++ program?
* What are the limitations of the program?
* Provide recommendation on improving the limitations of the program.

3. Flowchart OR Pseudocode. (30 points)

Vigenère Cipher Pseudocode

* Declare UPPER a global const char array with uppercase letters from ‘A’ to ‘Z’.
* Declare LOWER a global const char array with uppercase letters from ‘a’ to ‘z’.
* Declare three strings for plnText, textKey, and menuInt.
* Declare conType, a bool to represent which string the menu should display output. True for encryption or false for decryption.
* Display welcome message.
* Start a do…while loop. Loop while menuInt does not equal 3.
  + Call function to output menu options and prompt user for input.
  + Read user’s input and save to menuInt. 1 to encrypt, 2 to decrypt or 3 to exit.
  + Loop while to validate user’s input is less than 1 or greater 3,
    - Display invalid input message.
    - Call function to output menu options and prompt user for input.
    - Read user’s input. 1 to encrypt, 2 to decrypt or 3 to exit.
  + Switch case on user input
    - Case 1:
      * Set conType to true.
      * Call userInp function and pass plnText, textKey, conType.
        + plnText and textKey are reference variables.
      * Display plaintext.
      * Display key.
      * Call the encrypt function and pass plnText and textKey as arguments.
      * Display encrypted string.
    - Case 2:
      * Set conType to false.
      * Call userInp function and pass plnText, textKey, conType.
        + plnText and textKey are reference variables.
      * Display plaintext.
      * Display key.
      * Call the decrypt function and pass plnText and textKey as arguments.
      * Display decrypted string.
* End do…while loop
* Display exit message.

Pseudocode for menuCip Functions

* No function arguments.
* Display menu options. 1 for encrypt, 2 for decrypt, 3 to exit the program.

Pseudocode for userInp function

* Accepts string reference variable that represents either the plaintext string or an encrypted string.
* Accepts string reference variable that represents the cipher key.
* Accepts a boolean value that tells it which message to display to the user.
* Function reads and sets both strings with the user's input.

Pseudocode for encrypt function

* Accepts a constant string that represents the plaintext.
* Accepts a constant string that represents the cipher key.
* Declare a string for encrypted text.
* Loops through each letter in the plaintext string. Loop stops when i is less than the length of the plaintext string.
  + The if conditional checks if the current letter in the plaintext string is an uppercase letter, then:
    - Declare three integers: encChar, shift, encShift.
    - Set encChar with the subtraction of ‘A’ from the current plaintext’s character.
    - Set shift with the subtraction of the key[i] from ‘A’.
      * i equals i mod key’s length, because the key repeats to match the length of plaintext.
    - Set encShift by applying the encrypt formula: E = (encChar+shift) mod 26.
    - Concatenate the encrypted string with the UPPER arrays element that equals the value of encShift.
  + else if conditional checks if the current letter in the plaintext string is an lowercase letter, then:
    - Set encryptedChar with the subtraction of ‘a’ from the current plaintext’s character.
    - Set shift with the subtraction of the key[i] from ‘a’.
      * i equals i mod key’s length, because the key repeats to match the length of plaintext.
    - Set encShift by applying the encrypt formula: E = (encChar+shift) mod 26.
    - Concatenate the encrypted string with the LOWER arrays element that equals the value of encShift.
  + else concatenate a non-letter to the encrypted string.
* Return encrypted string.

Pseudocode for decrypt function

* Accepts a constant string that represents the plaintext.
* Accepts a constant string that represents the cipher key.
* Declare a string for decrypted text.
* Loops through each letter in the encrypted string. Loop stops when i is less than the length of the encrypted string.
  + The if conditional checks if the current letter in the encrypted string is an uppercase letter, then:
    - Declare four integers: decChar, shift, decShift, decAbs.
    - Set decChar with the subtraction of ‘A’ from the current character in the encrypted string.
    - Set shift with the subtraction of the key[i] from ‘A’.
      * i equals i mod key’s length, because the key repeats to match the length of plaintext.
    - Set decAbs by subtracting dechar and shift.
    - if conditional checks if decAbs is less than zero
      * Sets decShift with the decrypt formula: E = (decAbs +26) mod 26.
    - else
      * Set decAbs mod 26
    - Concatenate the decrypted string with the UPPER array’s element that equals the value of decShift.
  + else if conditional checks if the current letter in the encrypted string is an lowercase letter, then:
    - Declare four integers: decChar, shift, decShift, decAbs.
    - Set decChar with the subtraction of ‘A’ from the current character in the encrypted string.
    - Set shift with the subtraction of the key[i] from ‘a’.
      * i equals i mod key’s length, because the key repeats to match the length of plaintext.
    - Set decAbs by subtracting dechar and shift.
    - if conditional checks if decAbs is less than zero
      * Sets decShift with the decrypt formula: E = (decAbs +26) mod 26.
    - else
      * Set decAbs mod 26
    - Concatenate the decrypted string with the LOWER array’s element that equals the value of decShift.
  + else concatenate a non-letter to the decrypted string.
* Return encrypted string.

Flowchart: