Project 1

For this project, you will implement a simple Monoalphabetic Substitution Cipher (MASC). Such ciphers are of the "historical" variety, which is to say that they are not used for any modern, real-world encryption purposes but instead provide some insight into different approaches to protecting the confidentiality of a message. Here, we will use them to hone our ability to gather user input, work with strings, and write functions in C++.

Background:

A substitution cipher works by replacing characters from the original, non-encrypted message (the plaintext) with other characters in accordance with a pre-established key. As the name suggests, MASCs are substitution ciphers that only use a single alphabet. The replacement can be viewed as a one-to-one correspondence between the original alphabet and a ciphertext alphabet:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
C	W	E	I	G	U	J	M	Z	B	P	O	A	H	Q	R	S	D	L	X	T	V	K	Y	F	N

We would take each letter in our plaintext in turn, locating it in the upper alphabet and replacing it with the corresponding letter from the lower alphabet. In this way, the lower alphabet corresponds to the notion of a key: CWEIGUJMZBPOAHQRSDLXTVKYFN.

Using the above example, we would encrypt "Happy Halloween" as MCRRF MCOOQKGGH (note that by tradition we often capitalize all of the ciphertext). To further obscure the relationship between the two texts, we might also impose a uniform spacing on the ciphertext: MCRRF MCOOQ KGGH, placing a space after every five letters, for example. (If the result is not a multiple of our chosen token-length, we could also pad it using a particular letter... we won't bother with that here.)

You'll likely have already noticed that each plaintext letter is always mapped to the same ciphertext letter, which means the frequency of any particular letter in the original version of the message is preserved in the enciphered version . This why, in spite of having a key-space of $26! \approx 2^{88}$, this cipher is easily broken.

Requirements:

Your program should meet the following requirements:

1. Your program should encrypt a user-supplied message using a MASC. At a minimum, there should be separate functions that implement the following functionality:
    a. A function should gather a key from user input at the keyboard
        i. The candidate key from the user should contain only alphabetic characters
        ii. Such a key should be validated to ensure it contains exactly 26 English alphabetic characters and that no letters are repeated

b. If a key is gathered from the user, a function should convert each letter to uppercase
    i. Hint: there is a nice utility function available to you, named `toupper()`, which takes in a character and returns its uppercase counterpart
c. As an alternative to gathering the key from the user, a key can be randomly generated
    i. I will supply you with this function
d. A plaintext (potentially several words long) entered at the console can now be encrypted using the key
    i. This encryption function should be able to create a uniformly-spaced ciphertext with tokens of user-specified size
    ii. The user can specify a uniform spacing value or rely on a default value
        1. The function should have a default argument of 5 for the spacing
        2. If the user enters a negative value, the program should leave the original spacing unchanged

Once the above functions have performed the encryption, your main() function should display the ciphertext for the user.

You are not limited to just the above functions – you can provide more functions if you wish. Also, some of the interaction with the user may certainly take place in main(). For example, you can greet the user and prompt for the key towards the top of main(), prompt for (and gather) their input regarding spacing towards the middle of main(), and display the result to them at the end of main()... all of that is fine. Just be sure you have at least four functions other than main() that perform the work specified above (that counts the function I'll provide to you, so you really just need to write the other three).

For this project, you do not need to perform the decryption. Just the encryption phase is required here.

With regard to input validation: you must check the user's input of a key for validity, as described above. You must also allow the input plaintext to be more than one word and handle that properly. But otherwise, you don't need to worry about malformed input, exception handling, etc. If there are any non-alphabetic characters in the user-supplied plaintext (i.e., the string they're asking you to encrypt), you can just skip over these characters during the encryption process.

Tips:

You may recall that `cin.ignore()` discards the next character. Also useful is `cin.peek()`, which copies the next character but then leaves it still sitting in the input stream.

Sample output is provided below, for your use in testing and evaluating your program's results.

SUBMITTING YOUR WORK:

Once your programs for this assignment are complete, producing correct results without logic errors, you should submit your source code file to the course's Canvas page.

Please note that you do not need to zip up your entire solution/project and submit it – Canvas seems to work best with individual files. So if you wrote your program in a file named SubCipher.cpp, for example, you may just upload that file.

Sample runs:

[Run 1]

Testing Monoalphabetic Substitution Cipher (MASC) program:

Please make a selection:
(1) Type in a key
(2) Have a key generated for you

Choice: 1

Enter a key to be used for encryption and decryption. Include each letter of alphabet, none repeated:
ZHSRWOTEVXQPIABGYUCLFNJMDK
Key is now: ZHSRWOTEVXQPIABGYUCLFNJMDK

Enter the plaintext: THE Time Traveller (for so it will be convenient to speak of him) was expounding a recondite matter to us.

We will now disguise the original spacing.
How many letters should appear in each grouping?
(press Enter for default spacing of 5, negative entry leaves original spacing)
Spacing:

Ciphertext is: LEWLV IWLUZ NWPPW UOBUC BVLJV PPHWS BANWA VWALL BCGWZ QBOEV IJZCW MGBFA RVATZ UWSBA RVLWI ZLLWU LBFC

[Run 2]

Testing Monoalphabetic Substitution Cipher (MASC) program:

Please make a selection:
(1) Type in a key
(2) Have a key generated for you

Choice: 1

Enter a key to be used for encryption and decryption. Include each letter of alphabet, none repeated:
ZZHSRWOTEVXQPIABGYUCLFNJMDK
Incorrect key length. Must be 26 characters.

Enter a key to be used for encryption and decryption. Include each letter of alphabet, none repeated, no spaces:
ZHSRWOTEVXQPIABGYUCLFNJMD
Incorrect key length. Must be 26 characters.

Enter a key to be used for encryption and decryption. Include each letter of alphabet, none repeated, no spaces:
ZHSRWOTEVXQPIABGYUCLFNJMD?
Error: invalid character in key.

Enter a key to be used for encryption and decryption. Include each letter of alphabet, none repeated, no spaces:
ZZSRWOTEVXQPIABGYUCLFNJMDK
Error: letters in key must be used only once.

Enter a key to be used for encryption and decryption. Include each letter of alphabet, none repeated, no spaces:
zhsrwotevxqpiabgyuclfnjmdk
Key is now: ZHSRWOTEVXQPIABGYUCLFNJMDK

Enter the plaintext: this cipher is not secure

We will now disguise the original spacing.
How many letters should appear in each grouping?
(press Enter for default spacing of 5, negative entry leaves original spacing)
Spacing: 3

Ciphertext is: LEV CSV GEW UVC ABL CWS FUW

[Run 3]

Testing Monoalphabetic Substitution Cipher (MASC) program:

Please make a selection:
(1) Type in a key
(2) Have a key generated for you

Choice: 2

Generating random key...
Key is now: PGQLWXAKYNBRJVHOICESZMFTDU

Enter the plaintext: Four score and seven years ago

We will now disguise the original spacing.
How many letters should appear in each grouping?
(press Enter for default spacing of 5, negative entry leaves original spacing)
Spacing: 4

Ciphertext is: XHZC EQHC WPVL EWMW VDWP CEPA H

[Run 4]

Testing Monoalphabetic Substitution Cipher (MASC) program:

Please make a selection:
(1) Type in a key
(2) Have a key generated for you

Choice: 2

Generating random key...
Key is now: JXTSZDEOPWCIBKQRYHLMVNAGUF

Enter the plaintext: It was a bright cold day in April, and the clocks were striking thirteen

We will now disguise the original spacing.
How many letters should appear in each grouping?
(press Enter for default spacing of 5, negative entry leaves original spacing)
Spacing:

Ciphertext is: PMAJL JXHPE OMTQI SSJUP KJRHP IJKSM OZTIQ TCLAZ HZLMH PCPKE MOPHM ZZK

[Run 5]

Testing Monoalphabetic Substitution Cipher (MASC) program:

Please make a selection:
(1) Type in a key
(2) Have a key generated for you

Choice: 2

Generating random key...
Key is now: XSIRUOVKTBQNHAZGWMCLFPJYDE

Enter the plaintext: THE WHOLE OF GAUL IS DIVIDED INTO THREE PARTS

We will now disguise the original spacing.
How many letters should appear in each grouping?
(press Enter for default spacing of 5, negative entry leaves original spacing)
Spacing: 7

Ciphertext is: LKUJKZN UZOVXFN TCRTPTR URTALZL KMUUGXM LC

[Run 6]

Testing Monoalphabetic Substitution Cipher (MASC) program:

Please make a selection:
(1) Type in a key
(2) Have a key generated for you

Choice: 2

Generating random key...
Key is now: JLHUVDCRONGKWBQEMAFYSXTIZP

Enter the plaintext: 1234567890

We will now disguise the original spacing.
How many letters should appear in each grouping?
(press Enter for default spacing of 5, negative entry leaves original spacing)
Spacing: 3

Ciphertext is:

[Run 7]

Testing Monoalphabetic Substitution Cipher (MASC) program:

Please make a selection:
(1) Type in a key
(2) Have a key generated for you

Choice: 2

Generating random key...
Key is now: AZHQBOVSYJXTKRINPCFMUGELDW

Enter the plaintext: When in the course of human events

We will now disguise the original spacing.
How many letters should appear in each grouping?
(press Enter for default spacing of 5, negative entry leaves original spacing)
Spacing: -1

Ciphertext is: ESBR YR MSB HIUCFB IO SUKAR BGBRMF