National University of Singapore
School of Continuing and Lifelong Education

TIC1001: Introduction to Computing and Programming I
Semester I, 2019/2020

Problem Set 4
Strings and Vectors

Release date: 10th October 2019, 8:00 pm
Due: 10th November 2019, 6:00 pm

There are several tasks for this problem set. Make sure you answer every one of them in the **template file** provided before finalizing your submission on Coursemology.

Cipher Codes

In cryptography, a cipher (or cypher) is an algorithm for performing encryption or decryption—a series of well-defined steps that can be followed as a procedure. An alternative, less common term is encipherment. To encipher or encode is to convert information into cipher or code. In common parlance, "cipher" is synonymous with "code", as they are both a set of steps that encrypt a message; however, the concepts are distinct in cryptography, especially classical cryptography.

Codes generally substitute different length strings of characters in the output, while ciphers generally substitute the same number of characters as are input. There are exceptions and some cipher systems may use slightly more, or fewer, characters when output versus the number that were input.

Source: Wikipedia

Task 1: String Rotation (5 marks)

Before we begin coding the different ciphers, we first define some helper functions.

A. The function `char shift_char(char c, int n)` takes a character c and if c is a letter, it returns a letter n places away from c in alphabetical order. For example, if $c =$ `'c'`, $n = 2$ will return `'e'` and $n = -2$ will return `'a'`. The letters will wrap around A and Z, i.e., A -1 = Z and Z $+1$ = A.

If c is a lowercase letter, then the returned character will also be lowercase, and similarly for uppercase.

If c is not a letter, then c is simply returned.

Note that it is possible for $n > 26$ or $n < -26$. The counting will simply keep wrapping around, e.g. $n = 27$ will have the same effect as $n = 1$.

B. The function `string &rotate(string &s, int n)` takes a string s and shifts the characters of s by n places, causing the characters to wrap around the string. For example, given the string `"Hello World!"`, a shift of $n = 2$ will result in `"llo World!He"`. Basically, the

characters will shift 2 places to the left, resulting in the first two characters to be appended to the end of the string. In other words, $s_i = s_{i+n}$, with the appropriate wrap around handled.

The function will modify the input string s, as well as return s to allow for function composition. Note that $n < 0$ results in a shift in the opposite direction. It is also possible that n is larger than the length of the string, in which case the rotation will simply wrap around as many times as required.

Task 2: Caeser Cipher (5 marks)

A Caeser Cipher is also known as a shift cipher. In this scheme, the letters in the text is simply shifted by some fixed number. For example, if the shift is 3, then the following substitution scheme will be used, where the letters in the top row will be replaced by the corresponding letter in the bottom row:

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
d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c

A string such as `"Lorem ipsum dolor sit amet!"` will be encoded as `"Oruhp lsvxp groru vlw dphw!"`.

Write the function `string &caeser(string &s, int n)` which takes in a string s and encrypts it using a shift of n. Only the letters in the string need to be encrypted with its corresponding letter. Punctuations, digits, space and all other characters will remain unchanged. The function should modify s and also return it for function composition.

Note that to decrypt a text encoded with a shift of n, we simply call the same function with a shift of $-n$.

Task 3: Substitute (5 marks)

A Caeser Cipher is easily cracked because there are only 26 possible shifts (anything more ore less will simply wrap around). A Substitution cipher works by using a random replacement key. For example, one such key could be:

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
p	h	q	g	i	u	m	e	a	y	l	n	o	f	d	x	j	k	r	c	v	s	t	z	w	b

One can compute that there are $26!$ possible permutations of the key, which is quite a lot to crack using brute force.

A. Write the function `string &substitute(string &s, string key)` which takes a string s and a key and perform a substitution encryption of s. The key is simply a string representing the bottom row of the table above.

B. Write a function `string &unsubstitute(string &s, string key)` which takes a string s that has previously been encrypted by key, and decrypt it to return the original string.

Note that both functions should modify the string s and also return it to allow for function composition.

For example, the string `"Lorem ipsum dolor sit amet!"` using the key `"phqgiumeaylnofdxjkrcvstzwb"` will be encrypted as `"Ndkio axrvo gdndk rac poic!"`.

Task 4: Vigenere Cipher (5 marks)

While it is almost impossible obtain the key for a substitution cipher by trial-and-error, it is rather easy to crack the cipher by analysing the frequency of the letters and matching it with the known frequency of the language. This is because the substitution cipher is a monoalphabetic cipher, where each letter is always replaced with the same letter.

A Vigenere Cipher is a polyalphabetic cipher, where the same letter can be replaced by different letters at different times during the encryption process.

Instead of a key of letters, the key for a Vigenere cipher is a sequence of integers. For example, `7, 5, -6, 22, -13, 6`. Each letter in the plaintext string will then be subjected to a shift according to the corresponding number:

L	o	r	e	m		i	p	s	u	m		d	o	l	o	r		s	i	t
7	5	-6	22	-13	6	7	5	-6	22	-13	6	7	5	-6	22	-13	6	7	5	-6
S	t	l	a	z		p	u	m	q	z		k	t	f	k	e		z	n	n

The key is repeated as shown in the middle row. Each plaintext letter in the first row is then shifted by the corresponding number in the key to get the ciphertext in the bottom row. Note that punctuation, space and other non-letter characters remain unchanged.

A. Write the function `string &vigenere(string &s, vector<int> key)` which takes in a string s and encrypts it using a Vigenere cipher with the given key, which is a vector of integers.

B. Write the function `string &unvigenere(string &s, vector<int> key)` which takes in an encrypted string s that was encrypted using the given key, and decrypts it to obtain the original plaintext string.

Note again that both functions should modify the string s and also return it to allow for function composition.

Task 5: Cryptanalysis (5 marks)

As mention earlier, substitution ciphers can be easily broken using frequency analysis. Write the function `void freq(string s)` which takes a string as input, and prints the number of occurrences of each letter in the English alphabet, arranged in alphabetical order.

For example, `freq("Lorem ipsum dolor sit amet, consectetur adipiscing elit!")` will print the following as one continuous line:

```
a:2 b:0 c:3 d:2 e:5 f:0 g:1 h:0 i:6 j:0 k:0 l:3 m:3 n:2 o:4 p:2 q:0 r:3 s:4 t:5 u:2 v:0
w:0 x:0 y:0 z:0
```

As with before, upper and lower case letters are equivalent and all non-letters are ignored.