Tarea 2

Programación orientada al objeto

$1^{er.}$ Semestre 2018

Introducción

Esta tarea está diseñada para que Ud. experimente con las clases `string`, `vector`, `list` y/o `map` del lenguaje de programación C++. Si lo prefiere, puede también usar otros contenedores secuenciales y/o asociativos de la STL. Asimismo, puede aplicar algunos de los algoritmos genéricos de la STL.

1. Codificando textos usando una escítala

En criptografía, una escítala es un herramienta para generar mensajes encriptados. Consiste en un cilindro en el cual se envuelve un trozo de pergamino ó de cuero sobre el cual se escribe el mensaje. Este pergamino es luego enviado al destinatario, el cual enrolla el pergamino en un cilindro del mismo diámetro para leer el mensaje. Esto puede verse en la figura 1.

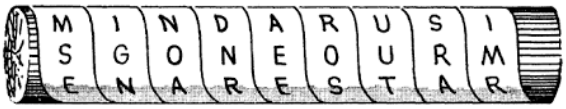


Figura 1: Cómo usar una escítala

Esta técnica fue usada en Grecia y Esparta para comunicarse durante las campañas militares, pues tiene la ventaja de ser de encriptación y decriptación rápida. Podemos encontrar una descripción del procedimiento en la obra de Plutarco, *Vida de Lisandro*.

Como ejemplo, si se usa una escítala para enviar el mensaje

"`Enunlugardelamanchadecuyonombrenoquieroacordarme`"

tal que ésta se enrolla 8 veces en el cilindro, el texto grabado en el trozo de cuero ó pergamino es

"`Ercoocndhnqoueaournldmidlaebeau Mcrrrgaueomanynae`"

Esto se muestra en la figura 2.

En este problema, entonces, se le solicita escribir una función llamada `string escitala(const vector<string>& texto, const unsigned int vueltas)`, que reciba como argumentos las palabras a encriptar y el número de vueltas al cilindro que da la escítala, y que retorne un objeto

E	n	u	n	l	u	g	a
r	d	e	l	a	M	a	n
c	h	a	d	e	c	u	y
o	n	o	m	b	r	e	n
o	q	u	i	e	r	o	a
c	o	r	d	a	r	m	e

Figura 2: Encriptación usando una escítala

`string` conteniendo el texto encriptado. Para mayor simplicidad, Ud. puede eliminar los espacios en el texto.

Puede usar el siguiente esqueleto de código para probar su función.

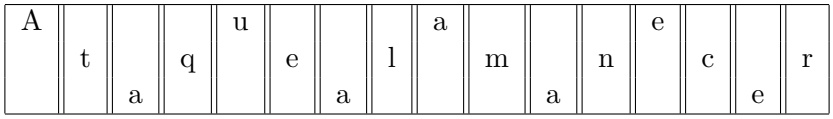
```
#include <iostream>
#include <vector>
#include <string>

using namespace std;

string escitala(const vector<string>& texto, const unsigned int vueltas);
int main() {
    unsigned int lados;
    cout << "Ingrese el número de lados: ";
    cin >> lados;
    cout << "Ingrese la frase a codificar (termine con ^D): ";

    vector<string> texto;
    string temp;
    while (cin >> temp) {
        texto.push_back(temp);
    }

    // INSERTE AQUI SU CODIGO
    cout << "La frase codificada es: " << escitala(texto, lados) << endl;
    return 0;
}
```

En este problema, Ud. puede usar índices ó iteradores, pero su código será mejor calificado si utiliza iteradores.

2. Codificando textos usando el código *empalizada*

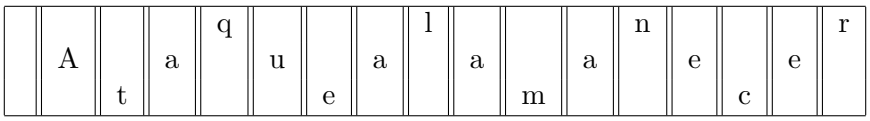
El código *empalizada*, también llamado código *Zig-Zag*, es un método de encriptación que se utilizó en tiempos de la guerra civil estadounidense para ocultar mensajes. Su principio básico se describe a continuación.

Dado un mensaje a transmitir, se supone que éste se escribe en las bardas de una empalizada en forma ascendente y descendente. La siguiente figura muestra la codificación del mensaje `Ataque al amanecer`, usando 3 filas y 16 bardas:



Nuevamente, se han eliminado los espacios del texto a codificar. El mensaje codificado se lee en las bardas de izquierda a derecha y de arriba hacia abajo. En el ejemplo, el mensaje codificado es `AuAetqelmncraAae`.

La dificultad del criptograma puede aumentarse al agregar un desplazamiento inicial a la codificación. Por ejemplo, la siguiente figura muestra la codificación del mismo mensaje `Ataque al amanecer` usando un desplazamiento de 1. En este caso, el mensaje codificado es `qlnrAauAAaeetemc`.



Se pide, entonces, escribir un programa que implemente la codificación y decodificación de mensajes via este método, usando objetos `string`, `vector` y/o `list`. En ambos casos, el texto a procesar debe estar en el archivo `entrada.txt`, y su programa debe escribir su salida en el archivo `salida.txt`. Su programa recibirá como entrada dada por el usuario la función a realizar (`codificar/decodificar`), el número de filas f a utilizar en la codificación y el desplazamiento inicial d. Puede ignorar los espacios en el texto de entrada. Use sólo iteradores para resolver este problema.

Finalmente, incluya en su informe la decodificación de este mensaje:

```
RABONIVGALACAOECRLEUOEVUDAONRCADHDMRRAEOTEIILOLNILGA
OIGLDRDAAAEONAHSQELSLNSIGNOERLGSALATAIEGECEOEECMHOMV
AADEZTERNRNYGEUALVQNSOSCDYBSSSTOELPNAUONCIAPEHENULNC
NNIORAHPEUDEDASRATAFOORNLGSURACMOUSROAOELRAAINDSIOMS
SSUNULAAUOOURAMCOUNILSEAODIULCCRALOAEALIANEOATBDJSNS
LMAIDMNUTEDSDNMYQDUQHONAGAOOMCPSLNAAEODOSREA
```

3. Codificando textos usando un *one-time pad*

Los métodos criptográficos anteriores son sencillos de analizar y de quebrar, ya que siguen patrones de distribución espacial sencillos. Un método criptográfico muchísimo más difícil de quebrar se basa en reemplazar las letras del texto original por cifras tomadas de *pads*, textos predefinidos conocidos sólo por el transmisor y el receptor del mensaje. Si este texto es usado sólo una vez, este método se conoce como *one-time pad*.

Por ejemplo, sea el siguiente *pad*, correspondiente a los primeros dos párrafos de *Don Quijote de la Mancha*, al cual se ha agregado información relativa a las filas y columnas del texto.

```
         1         2         3         4         5         6         7
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345
1  En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho
2  tiempo que vivia un hidalgo de los de lanza en astillero, adarga antigua,
3  rocin flaco y galgo corredor. Una olla de algo mas vaca que carnero,
4  salpicon las mas noches, duelos y quebrantos los sabados, lantejas los
5  viernes, algun palomino de añadidura los domingos, consumian las tres
6  partes de su hacienda. El resto della concluian sayo de velarte, calzas de
7  velludo para las fiestas, con sus pantuflos de lo mesmo, y los dias de
8  entresemana se honraba con su vellori de lo mas fino. Tenia en su casa una
9  ama que pasaba de los cuarenta, y una sobrina que no llegaba a los veinte,
10 y un mozo de campo y plaza, que asi ensillaba el rocin como tomaba la
11 podadera. Frisaba la edad de nuestro hidalgo con los cincuenta años; era de
12 complexion recia, seco de carnes, enjuto de rostro, gran madrugador y amigo
13 de la caza. Quieren decir que tenia el sobrenombre de Quijada, o Quesada,
14 que en esto hay alguna diferencia en los autores que deste caso escriben;
15 aunque, por conjeturas verosimiles, se deja entender que se llamaba
16 Quejana. Pero esto importa poco a nuestro cuento; basta que en la narracion
17 del no se salga un punto de la verdad.
18
19 Con estas razones perdia el pobre caballero el juicio, y desvelabase por
20 entenderlas y desentrañarles el sentido, que no se lo sacara ni las
21 entendiera el mesmo Aristoteles, si resucitara para solo ello. No estaba
22 muy bien con las heridas que don Belianis daba y recebia, porque se
23 imaginaba que, por grandes maestros que le hubiesen curado, no dejaria de
24 tener el rostro y todo el cuerpo lleno de cicatrices y señales. Pero, con
25 todo, alababa en su autor aquel acabar su libro con la promesa de aquella
26 inacabable aventura, y muchas veces le vino deseo de tomar la pluma y dalle
27 fin al pie de la letra, como alli se promete; y sin duda alguna lo hiciera,
28 y aun saliera con ello, si otros mayores y continuos pensamientos no se lo
29 estorbaran. Tuvo muchas veces competencia con el cura de su lugar -que era
30 hombre docto, graduado en Siguenza-, sobre cual habia sido mejor caballero:
31 Palmerin de Ingalaterra o Amadis de Gaula; mas maese Nicolas, barbero del
32 mesmo pueblo, decia que ninguno llegaba al Caballero del Febo, y que si
33 alguno se le podia comparar, era don Galaor, hermano de Amadis de Gaula,
34 porque tenia muy acomodada condicion para todo; que no era caballero
35 melindroso, ni tan lloron como su hermano, y que en lo de la valentia no le
36 iba en zaga.
```

Este método de encriptación consiste en reemplazar cada letra del texto original por la fila y la columna de una ocurrencia de la letra en el texto. Ya que cada letra puede aparecer muchas veces en el texto, el número de pares distintos que pueden representar una letra cualquiera es muy alto. Por ejemplo, la letra `a` puede representarse como los pares $(1, 10)$, $(3, 38)$, $(8, 74)$ y $(36, 3)$, entre muchos otros. Luego, cada vez que se debe codificar esta letra se reemplaza por alguno de estos pares escogido al azar. Mientras más extenso sea el *pad*, mayor será la complejidad del código generado.

Ojos avizores ya habrán notado que en el texto de *Don Quijote de la Mancha* no aparecen las letras `k` ni `w`. Reemplácelas por los pares $(0,0)$ y $(0,1)$ respectivamente.

Quebrar este código es extraordinariamente difícil: sin embargo, si se encriptan muchos mensajes o textos de gran extensión usando el mismo *pad*, es posible descifrar el mensaje usando métodos estadísticos. Por ello, idealmente cada *pad* debiese ser usado sólo una vez y posteriormente descartado y destruido.

Como se mencionó anteriormente, el proceso de encriptación de un texto consiste en reemplazar cada letra del texto original por un par $(fila, columna)$ correspondiente a esa letra, escogido al azar. El proceso de decriptación, en cambio, consiste en identificar la letra existente en la posición $(fila, columna)$ del *pad*. Escriba, entonces, un programa que implemente este método de encriptación que use el *pad* mostrado. Su programa debe leer un archivo de entrada conteniendo el texto a encriptar y generar un archivo de salida conteniendo el texto encriptado. Escriba, además, un programa que desencripte el texto encriptado usando el mismo *pad*. Puede usar el siguiente esqueleto de código para ambos programas.

```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
using namespace std;

int main(){
    ifstream entrada("entrada.txt");
    ifstream donQuijote("DonQuijote01.txt");
    ofstream salida("salida.txt");

    // Inserte codigo que lea el texto de entrada

    vector<string> texto;
    string temp;
    while(getline(donQuijote, temp)) {
        texto.push_back(temp);
    }
    donQuijote.close();

    // Inserte su codigo aqui, y agregue codigo que escriba el texto de salida
    return 0;
}
```

Sugerencia: su solución a este problema puede verse beneficiado en términos de claridad y concisión si Ud. utiliza contenedores asociativos e iteradores.

Esta tarea puede ser realizada en grupos de máximo 3 personas. Sus códigos fuentes deberán ser subidos a InfoAlumno como un archivo de nombre `integrantes.rar` antes de las 6 pm del día 24 de mayo. Además, deberá entregar un informe escrito en secretaría de Electrónica antes de esa fecha, que debe incluir un listado de sus programas y diagramas de flujo generales, además de una descripción de las dificultades encontradas y cómo éstas fueron solucionadas.