Tarea 1: [Nombre del Algoritmo]

Nombres y Apellidos: Luis Fredy Huachaca Vargas

Enlace del Github: https://github.com/Luis-Huachaca-HV/LUIS_FREDY_HUACHACA_VARGAS/

Comentarios:

- Para hacer el encriptador del mensaje, considere un abecedario de 26 caracteres, que se usará como diccionario de nuestro código, después de ello haré un for de 26 veces, que evaluará el lugar de la primera palabra y reemplazará en otra cadena esta palabra más sus posiciones de acuerdo a la clave, si la suma del lugar más la clave no pasa de 26 se hará el reemplazo común, si el lugar más la clave pasa de 26 hará la función módulo a la clave más el lugar y modificará la otra string.
- El desencriptador será un poco lo mismo, sin embargo, cambiaremos la suma de la posición con clave con una resta, si nos pasamos a un número negativo en esa resta, haremos la función módulo que saque un residuo positivo y modifique la cadena.
- En el caso del desencriptador sin clave, por fuerza bruta, entendí que es importante iterar sobre la cantidad de caracteres disponibles, entonces, usaremos esta iteración de 26 veces probando cada valor iterativo como clave, ya que, al usar módulo en el desencriptador con clave, aprovechamos que son solo 26 los dígitos probables.

Estructura del Algoritmo:

- Clases
    - emisor_cesar.h

```
#pragma once
#include <iostream>
#include <string>
using namespace std;
```

```
class Emisor {
private:
int clave;
public:
string mensaje;
string mensaje_oc;

Emisor(string _mensaje_oc, string _mensaje, int _clave) {
mensaje_oc = _mensaje_oc;
mensaje = _mensaje;
clave = _clave;
};

string get_mensaje_oc() {
return mensaje_oc;
};
string get_mensaje() {
return mensaje;
};

void mostrar_mensaje() {
cout << mensaje << endl;

};
const int get_clave() {
return clave;
};

};
```

- receptor_cesar.h

```
#pragma once
#include <iostream>
#include <string>


using namespace std;



class Receptor {
private:
int clave;
public:
string mensaje_oc;
string mensaje_co;

Receptor(string _mensaje_oc, string _mensaje_co, int _clave) {
mensaje_oc = _mensaje_oc;
mensaje_co = _mensaje_co;
clave = _clave;
};
int get_clave() {
return clave;
};

string get_mensaje_oc() {
return mensaje_oc;
};
string get_mensaje_co() {
return mensaje_co;
};
```

```
void mostrar_mensaje() {
cout << mensaje_co << endl;
}
};
```

- Código
    - Cesar.cpp

```
// Cesar.cpp : This file contains the 'main' function. Program execution begins and ends
there.
//


#include <iostream>
#include <string>
#include "emisor_cesar.h"
#include "receptor_cesar.h"

using namespace std;

int fe_modulo(int a, int n) {
int q = a / n;
int r = a - (n * q);

if (r < 0) {
if (q <= 0)
q = q - 1;
else
q = q + 1;

r = a - (n * q);
}

return r;
}
string cesar_encriptador( string frase, int clave) {
string abc = "abcdefghijklmnopqrstuvwxyz";
int lug = 0;
string new_str = "";
for (int i = 0; i < frase.size(); i++) {
lug = abc.find(frase[i]);
if (lug + clave < abc.size())
new_str = new_str + abc[lug+clave];
else {
new_str = new_str + abc[fe_modulo((lug + clave), abc.size())];
}
}
return new_str;
}


string cesar_desencriptador(string frase_oc, int clave) {
string abc = "abcdefghijklmnopqrstuvwxyz";
int ancho = abc.size();
int lug = 0;
int n_pos = 0;
string fra_co = "";
for (int i = 0; i < frase_oc.size(); i++) {
n_pos = abc.find(frase_oc[i]);
if ((n_pos - clave >= 0))
fra_co = fra_co + abc[n_pos-clave];
else {
lug = fe_modulo(n_pos-clave, ancho);
```

```
fra_co = fra_co + abc[lug];
}
}

return fra_co;

}
long long powint(int base, int exp) {
long long result = base;
for (int i = 0; i < exp; i++) {
result *= base;
}
return result;
}

void cesardescipher(string frase);

int main()
{
//Cesar
//string msje;
string mensaje_inicial = "";
cout << "ingrese su mensaje" << endl;
getline(cin,mensaje_inicial);
int clave;
cout << "ingrese su clave: ";
cin >> clave;
Emisor emisor("",mensaje_inicial,clave);

string emi_mensaje = emisor.get_mensaje();
emisor.mostrar_mensaje();



string msj_oc = cesar_encriptador(emi_mensaje,clave);
cout << msj_oc << endl;
Receptor receptor(msj_oc, "",clave);
cout << receptor.mensaje_oc << endl;

receptor.mensaje_co = cesar_desencriptador(msj_oc,clave);
receptor.mostrar_mensaje();


cout << "\n descifrado por fuerza bruta\n";

//parte de descifrador por fuerza bruta
cesardescipher("hols");

return 0;
}


//descifrador por fuerza bruta

void cesardescipher(string frase){
string msj = "";
int val = 0;
for (int i = 0; i < 26; i++) {
msj = cesar_desencriptador(frase, i);

cout << msj << " "<< i <<  endl;
}
}
```

Copie el código: no se olvide