Tarea 2: [Rail Fance]

Nombres y Apellidos: Massiel Oviedo Sivincha

Enlace del Github:
https://github.com/massielovi/Massiel_OviedoSivincha/blob/main/Rail%20Fance

---

Comentarios:

- Tuve en cuenta las instrucciones, como: un key (filas), no trabajar con matrices, la lectura horizontal del cifrado y trabajar con string y vectores.
- Una de las dificultades que tuve en el código fue como haría que la manera lógica en la que había pensado se hiciera realidad en el código; tomando en cuenta las diferentes sumas dependiendo de la posición y tener tres secuencias en un mismo for.
- Preferí subirlo enteramente en un cpp, sin hpp debido a que me gustaría experimentar y aprender más de GitHub para los próximos trabajos.
- Al realizar este trabajo me di cuenta que al poner varios casos de un problema en específico, me ayuda a tener un sentido más amplio en cuanto a mi trabajo, además te das cuenta que una vez hecho el cifrado, el descifrado se te facilitara.

a) Estructura del Algoritmo:

o Clases

`class Send_message`

- Una clase que nos facilitara poner simplemente:
  Emisor.cifrado();
  Emisor.decifrado();
  Además, nos ayudara a manejar variables privadas como filas, message, message_arreglado y vector <char> criptado.

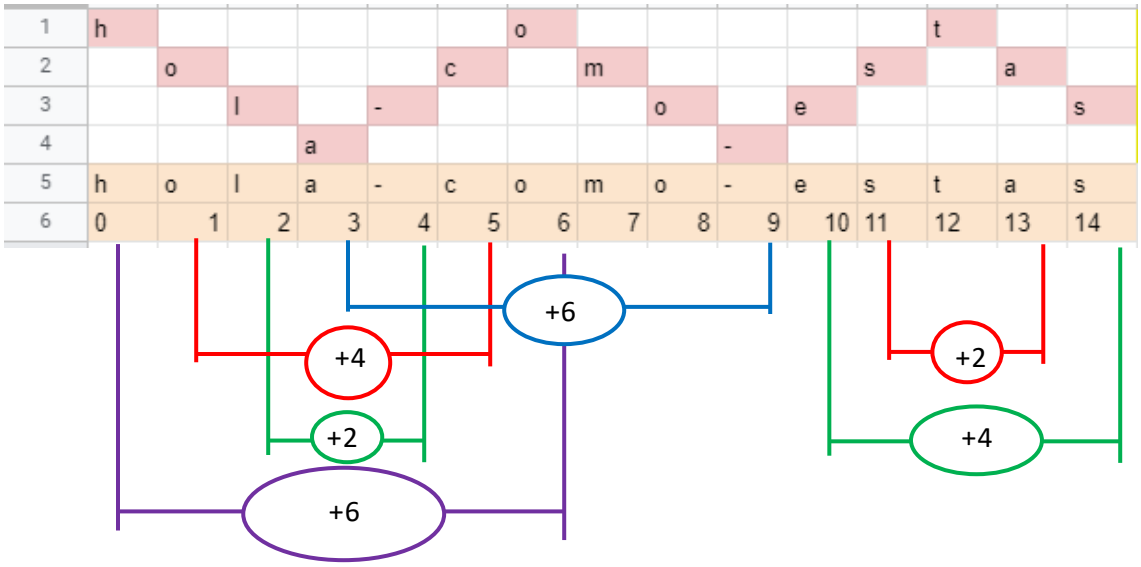
o Funciones

I. `string Ordenar (int fil, string message)`
- Función fuera de la clase, ayuda a una mejor visualización del cifrado. Reemplazando los espacios en blanco en guiones. Meramente visual, no es elemental.

II. `Send_message:: Send_message(int f,string m)`
- Constructor de la clase Send_message, llama al key (fila) y al mensaje string.

III. `void Send_message::cifrado()`
- En primer lugar llama a la función Ordenar y crea una nueva variable para guardar ese cambio.
- La variable entero “a” es la primera diferencia en mi string.Si revisamos diferentes ejemplos notaremos que al poner el mensaje en una matriz del tamaño de las filas , en la fila 0 encontramos la posición [0] y [0+(filas*2)-2];

teniendo en cuenta que se le suma (filas*2)-2) cada vez que avanza. Ejemplo en el caso de que key=4 entonces a=6.



- La variable a_1= es una variable estática.
- Nuestro for ayudara a recorrer cada fila, ya leyendo horizontalmente. Dentro del for declare un entero i que también mantendrá el valor de b, ya que b se ira moviendo a las sumas de la imagen de arriba.
- Luego tenemos un if que nos asegurará la entrada de la primera y última fila, donde usaremos un while para no traspasar la longitud del string, también usaremos la variable estática "a_1" ya que esta siempre mantendrá la suma más alta.
- En el else entraran las demás filas que de nuevo con un while se asegura que no traspase la longitud del string. Aquí si modificaremos la variable a que sumara los múltiplos de 2 subiendo de acuerdo a la fila.
- Cada for se restará la variable a dos espacios y la variable b volverá a su valor principal igualándolo con i. Cabe resaltar que cada carácter será guardado en el vector <char> criptado.

h	o	t	o	c	m	s	a	l	-	o	e	s	a	-
0	6	12	1	5	7	11	13	2	4	8	10	14	3	9

0	6+	6+		
1	4+	2+	4+	2+
2	2+	4+	2+	4+
3	6+			

- Por último, imprimiremos el descifrado como un vector.

IV. void Send_message::decifrado ()

- En nuestro descifrado haremos el mismo procedimiento que el cifrado, pero intercambiando algunas variables.

- Usaremos un puntero “ptr” que ayudara a recorrer los valores sin ser cambiados por algún while o for. Este puntero nos indicará las posiciones y recorrerá sin necesidad de un for todo el vector anterior.
- Por último, guardaremos en un array char el decifrado y lo imprimiremos.

b) Código

```
#include <iostream>
#include <string>
#include <vector>
using namespace std;
string Ordenar(int fil, string message){//uso esta funcion para los casos donde un espacio equivale a '-'
		for(int a=0;a<message.length();a++){
			if (message[a]==' ' ){message[a]='-';}
		}
	;
	return message;
}
class Send_message{
	private:
		int filas;
		string message,message_arreglado;
		vector <char> criptado;
	public:
		Send_message(int f,string m){// contructor de la clase
			filas=f;message=m;
			}
		void cifrado(){// ordenamos el mensaje e imprimimos el cifrado del mensaje
			message_arreglado=Ordenar(filas,message);
			int a=(filas*2)-2;//Primera diferencia con la que jugaremos con el valor
			int a_1=a;//Nos ayuda como valor estatico
			for(int b=0;b<filas;b++){
				int i=b;//ayuda a volver el valor de b
				if(b==0 or b==filas-1){
```

```
                                        while (b<message_arreglado.length()){
                                                criptado.push_back(message_arreglado[b]);
                                                b=b+a_1;}
                                }
                                else {
                                        criptado.push_back(message_arreglado[b]);
                                        while (b<message_arreglado.length()){
                                                        b=b+a;

        if(b>=message_arreglado.length()){break;}

        criptado.push_back(message_arreglado[b]);
                                                        b=b+(i*2);

        if(b>=message_arreglado.length()){break;}

        criptado.push_back(message_arreglado[b]);
                                                }
                                }
                                a=a-2;b=i;
                        }
                        //Imprimimos el criptado
                        cout<<"Cyphertext: ";
                        for(int j=0;j<criptado.size();j++){
                                //message[j]=cript[j];
                                cout<<criptado[j];}
                        cout<<endl;
                }
                void decifrado(){
                        int a=(filas*2)-2;//volvemos a iniciar a
                        int a_1=a;//Nos ayuda como valor estatico
                        char decriptado[criptado.size()];//creamos un array char para acceder
a sus
                        int pos=0;//para recorrer vector
```

```
                int* ptr = & pos;//para que el valor recorra libremente cambiandose
valores
                for(int b=0;b<filas;b++){
                        int i=b;//pivote de ayuda
                        if(b==0 or b==filas-1){
                                while (b<criptado.size()){
                                        decriptado[b]=criptado[*ptr];
                                        *ptr=*ptr+1;
                                        b=b+a_1;}
                        }
                        else {
                                decriptado[b]=criptado[*ptr];*ptr=*ptr+1;
                                while (b<criptado.size()){
                                                b=b+a;
                                                if(b>=criptado.size()){break;}

        decriptado[b]=criptado[*ptr];*ptr=*ptr+1;
                                                b=b+(i*2);
                                                if(b>=criptado.size()){break;}

        decriptado[b]=criptado[*ptr];*ptr=*ptr+1;
                                        }
                        }
                        a=a-2;b=i;
                }
                //Imprimimos mensaje decifrado
                cout<<"Mensaje: ";
                for(int j=0;j<criptado.size();j++){
                        if (decriptado[j]=='-'){decriptado[j]=' ';
                        }
                        cout<<decriptado[j];}
                cout<<endl;
                }
};
```

```
int main() {
	int filas;
	string mensaje;//Aqui hacemos nuestra facilidad para ingresar la clave junto al mensaje
	cout<<"Ingresa key(f):";cin>>filas;
	cout<<endl<<"Ingresa el mensaje que quieres enviar:";
	getline(cin, mensaje);getline(cin, mensaje);
	Send_message Emisor (filas,mensaje);
	Emisor.cifrado();
	Emisor.decifrado();
	return 0;
}
```