Proyecto 2- Cifrado cesar

Grupo:

Juan Camilo Riaño

Antonio ortega

Introducion

La criptografía, una disciplina ancestral que ha evolucionado en paralelo con la historia de la humanidad, juega un papel fundamental en la seguridad de la información. Entre sus múltiples métodos de cifrado, el cifrado César destaca como uno de los más simples y, a su vez, fascinantes. Originario de la antigua Roma, este método se basa en una sutil trasposición de caracteres que ha perdurado a lo largo de los siglos.

En este informe, exploraremos el cifrado César aplicado en el contexto moderno utilizando el lenguaje de programación C. Nuestro objetivo es desentrañar un mensaje encriptado, aprovechando las capacidades de este lenguaje para implementar una solución eficiente y comprender los fundamentos detrás de este método milenario de cifrado.

A través del análisis y la implementación práctica en C, nos sumergiremos en la esencia del cifrado César, descubriendo cómo una simple transposición de caracteres puede ocultar información de manera temporalmente segura. Este viaje nos permitirá no solo comprender la mecánica del cifrado César, sino también apreciar la conexión entre la tradición y la modernidad en el vasto campo de la criptografía.

Desafio

Descifre el siguiente mensaje:

GEUDOPPSJPDDKDBJQBNYXGKKFYKTGMIWIHLGNDMGBEJOMKXICRECGMYHDT

Y proporcione la respuesta que se solicita en el mensaje oculto.

Pista:

Número pseudo aleatorio = 4
ROTOR 1 = BDFHJLCPRTXVZNYEIWGAKMUSQO
ROTOR 2 = AJDKSIRUXBLHWTMCQGZNPYFVOE
ROTOR 3 = EKMFLGDQVZNTOWYHXUSPAIBRCJ

Codigo en lenguaje c

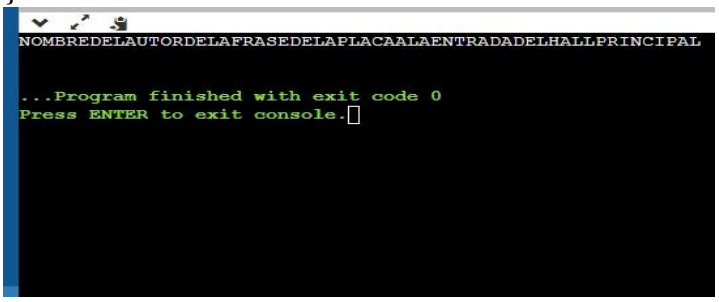
```
#include <stdio.h>
#include <string.h>

char rotor1[] = "BDFHJLCPRTXVZNYEIWGAKMUSQO";
char rotor2[] = "AJDKSIRUXBLHWTMCQGZNPYFVOE";
char rotor3[] = "EKMFLGDQVZNTOWYHXUSPAIBRCJ";

char findOriginal(char c, char* rotor) {
    for(int i = 0; i < 26; i++) {
        if(rotor[i] == c) {
            return 'A' + i;
        }
    }
    return ' ';
}

void decipher(char* message, int pseudoRandomNumber) {
    int len = strlen(message);
    for(int i = 0; i < len; i++) {
        char c = message[i];
        c = findOriginal(c, rotor3);
        c = findOriginal(c, rotor2);
        c = findOriginal(c, rotor1);
        c = ((c - 'A' - pseudoRandomNumber - i) % 26 + 26) % 26 + 'A';
        printf("%c", c);
    }
    printf("\n");
}

int main() {
    char message[] =
"GEUDOPPSJPDDKDBJQBNYXGKKFYKTGMIWIHLGNDMGBEJOMKXICRECGM
YHDT";
    decipher(message, 4);
    return 0;
}
```



Este código implementa un descifrador de mensajes utilizando un cifrado de la máquina Enigma, un dispositivo de cifrado electromecánico utilizado por Alemania durante la

Segunda Guerra Mundial. La máquina Enigma constaba de rotores, reflectores y un conjunto de reglas para cifrar y descifrar mensajes. En este caso, se simulan tres rotores (rotor1, rotor2 y rotor3) y se realiza el proceso de descifrado.

Primero Se definen tres rotores, cada uno representado por una cadena de caracteres que especifica la correspondencia entre las letras de entrada y salida.

Segundo función ´char´ toma un carácter ´c´ y un rotor, y devuelve el carácter original antes de ser procesado por ese rotor.

Tercero pasa el mensaje sobre cada carácter del mensaje, lo pasa por los tres rotores en orden inverso, y luego aplica un ajuste según un pseudo número aleatorio y la posición del carácter en el mensaje. Finalmente, imprime el carácter descifrado.

Por ultimo En la función principal, se define un mensaje cifrado y se llama a la función `decipher` con ese mensaje y un pseudo número aleatorio de 4 para realizar el descifrado. El resultado se imprime en la consola.

Concluion

En esta exploración del cifrado inspirado en la máquina Enigma mediante el uso del cifrado César, hemos desentrañado los misterios de un mensaje encriptado. A través de la implementación en lenguaje C, hemos recreado la complejidad de los rotores y su influencia en el proceso de cifrado y descifrado.

La simplicidad aparente del cifrado César se ha transformado en una experiencia fascinante al incorporar la mecánica de la máquina Enigma, evidenciando cómo un algoritmo aparentemente sencillo puede ganar complejidad y robustez al combinarse con otros elementos.