Universidad Nacional Autónoma de México

Facultad de ingeniería

Estructura de Datos y Algoritmos 1

Actividad #3

Cifrado Cesar C++

José Carlos Avalos Jasso

22/03/2021

Lenguaje C++

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#define LONGITUD_ALFABETO 26
#define INICIO_ALFABETO_MAYUSCULAS 65
#define INICIO_ALFABETO_MINUSCULAS 97
#define MAXIMA_LONGITUD_CADENA 5000
#define MOD(i, n) (i % n + n) % n // Calcular módulo positivo,
const char *alfabetoMinusculas = "abcdefghijklmnopqrstuvwxyz",
         *alfabetoMayusculas = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
void cifrar(char *mensaje, char *destino, int rotaciones);
// Mensaje al usuario, como el movimiento que se dará al codigo
void descifrar(char *mensaje, char *destino, int rotaciones);
// Obtener el valor entero de un carácter:
int ord(char c);
int main(void) {
  // El original, el cifrado y luego el descifrado
  char mensaje[MAXIMA_LONGITUD_CADENA],
mensajeCifrado[MAXIMA_LONGITUD_CADENA],
     mensajeDescifrado[MAXIMA_LONGITUD_CADENA];
  printf("Escribe un mensaje para que lo cifre [Maximo %d caracteres]:\n",
        MAXIMA_LONGITUD_CADENA - 1);
  // Esto es para obtener el mensaje y evitar que se sobrepase el limite
  fgets(mensaje, MAXIMA_LONGITUD_CADENA, stdin);
  mensaje[strcspn(mensaje, "\r\n")] = 0;
  int rotaciones;
  printf("Escribe el número de rotaciones que se darán a las letras:\n");
  scanf("%d", &rotaciones);
  // Ahora sí ciframos y descriframos
```

```
  printf("El mensaje original es: %s\n", mensaje);
  cifrar(mensaje, mensajeCifrado, rotaciones);
  printf("El mensaje cifrado es: %s\n", mensajeCifrado);
  descifrar(mensajeCifrado, mensajeDescifrado, rotaciones);
  printf("El mensaje descifrado es: %s\n", mensajeDescifrado);
  return 0;
}
void cifrar(char *mensaje, char *destino, int rotaciones) {
  /*Recorrer cadena*/
  int i = 0;
  while (mensaje[i]) {
    char caracterActual = mensaje[i];
    int posicionOriginal = ord(caracterActual);
    if (!isalpha(caracterActual)) {
      destino[i] = caracterActual;
      i++;
      continue; // Ir a la siguiente parte; por eso arriba aumentamos a i
    }
    if (isupper(caracterActual)) {
      destino[i] =
          alfabetoMayusculas[(posicionOriginal - INICIO_ALFABETO_MAYUSCULAS +
                              rotaciones) %
                             LONGITUD_ALFABETO];
    } else {

      destino[i] =
          alfabetoMinusculas[(posicionOriginal - INICIO_ALFABETO_MINUSCULAS +
                              rotaciones) %
                             LONGITUD_ALFABETO];
    }
```

```
    i++;
  }
}
void descifrar(char *mensaje, char *destino, int rotaciones) {
  /*Recorrer cadena*/
  int i = 0;
  while (mensaje[i]) {
    char caracterActual = mensaje[i];
    int posicionOriginal = ord(caracterActual);
    if (!isalpha(caracterActual)) {
      destino[i] = caracterActual;
      i++;
      continue; // Ir a la siguiente parte; por eso arriba aumentamos a i
    }
    if (isupper(caracterActual)) {
      destino[i] = alfabetoMayusculas[MOD(
          posicionOriginal - INICIO_ALFABETO_MAYUSCULAS - rotaciones,
          LONGITUD_ALFABETO)];
    } else {
      destino[i] = alfabetoMinusculas[MOD(
          posicionOriginal - INICIO_ALFABETO_MINUSCULAS - rotaciones,
          LONGITUD_ALFABETO)];
    }
    i++;
  }
}
int ord(char c) { return (int)c; }
```