



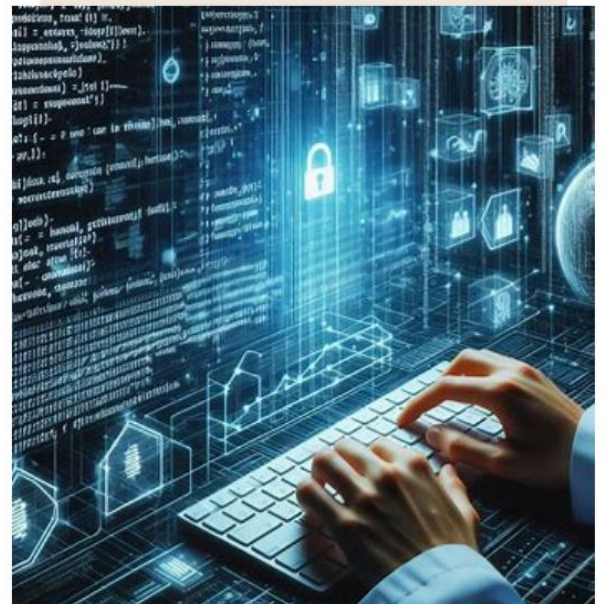
PROYECTO 8: SecureText: Sistema de Encriptación y Manipulación de Texto en C++

GRUPO 1:

- Crisóstomo Altamirano, Axl Mikel
23200165
- Barrientos Torres, José Paolo
23200145
- Aceituno Huamán, Luis Gerónimo
23200002
- Torres Chuquiyauri, Manuel Angel
23200066

PROFESOR:

- Guerra Grados, Luis Ángel



Información general:

Nombre del Proyecto	SecureText: Sistema de Encriptación y Manipulación de Texto en C++
Usuario	
Preparado por:	<ul style="list-style-type: none">• Crisostomo Altamirano, Axl Mikel• Torres Chuquiyauri, Manuel Torres• Aceituno Huaman, Luis Geronimo• Barrientos Torres, Jose Paolo
Fecha de preparación	29/05/2024
Fecha de cierre	26/06/2024
Autorizado por	Guerra Grado Luis Angel

A. Propósito del proyecto (descripción):

Descripción:

El proyecto elegido consiste en desarrollar un menú de opciones en C++ que abarca diversas funciones de manipulación y encriptación de texto. A continuación, se detallan las opciones disponibles:

1. Opción A: Este programa lee líneas de texto, obtiene las palabras de cada línea y las escribe en pantalla en orden alfabético. Se considera que el máximo número de palabras por línea es 28.
2. Opción B: Se implementa un sistema de encriptación simple que sustituye cada carácter de un mensaje por el carácter situado tres posiciones alfabéticas adelante. Se debe escribir una función que tome una cadena como parámetro y devuelva otra cadena cifrada de acuerdo a este método.
3. Opción C: Otro sistema de encriptación sustituye cada carácter del alfabeto por otro previamente decidido. La función correspondiente debe tomar como parámetros el mensaje a cifrar y una cadena que contenga las correspondencias ordenadas de los caracteres alfabéticos. La función devolverá un puntero a la cadena cifrada del mensaje.
4. Opción D: Este método de encriptación cambia todas las letras minúsculas por otras nuevas letras, según un array de caracteres cifrados proporcionado. Por ejemplo, se cambiará "a" por `cifrado[0]`, "b" por `cifrado[1]`, etc. Se debe escribir una función que reciba un texto y lo encripte utilizando este sistema.
5. Opción E: Este método de encriptación se basa en la sustitución de letras mayúsculas. Cada carácter del mensaje es sustituido por otro carácter según una correspondencia específica dada. La entrada incluye un número N, que indica la cantidad de conjuntos de datos, y para cada conjunto de datos, una línea con el mensaje codificado y otra línea con las correspondencias de los caracteres alfabéticos. La salida debe mostrar el número del conjunto de datos y el mensaje descifrado.

Propósito:

El propósito de este proyecto es desarrollar y fortalecer sus habilidades en programación en C++, enfocándose en la manipulación y encriptación de texto. A través del uso de distintos conocimientos obtenido en clase y de fuentes externas, usados para desarrollar el proyecto, pudimos hacer lo siguiente:

- **Mejoramos nuestros conocimientos en C++:** Al trabajar con distintas estructuras de datos y algoritmos, pudimos aplicar conceptos fundamentales del lenguaje C++.
- **Desarrollamos habilidades en manipulación de cadenas:** Las tareas requieren operaciones complejas de manejo de texto, incluyendo ordenamiento y transformación de cadenas.
- **Comprendimos técnicas de encriptación:** Exploramos diferentes métodos de encriptación, comprendiendo tanto sus aplicaciones como sus implementaciones.
- **Practicamos la lógica de programación:** Cada opción del menú presentaba un desafío lógico, lo cual nos ayudó a mejorar nuestras capacidades de resolución de problemas.

En resumen, este proyecto no solo nos sirvió como una herramienta de encriptación, sino también nos sirvió para aprender y practicar C++, también nos ayudó a comprender conceptos importantes de encriptación y manipulación de texto, y aplicar estos conocimientos de manera práctica y efectiva.

B. Objetivo del proyecto:

Metas del Grupo de Trabajo	Objetivos del proyecto
Desarrollar un Sistema de Menú Interactivo en C++	Crear una interfaz de menú en la consola que permita al usuario seleccionar entre diferentes opciones de procesamiento de texto.
Optimizar la Manipulación de Cadenas de Texto	Implementar una función que lea líneas de texto, separe las palabras y las ordene en orden alfabético, manejando hasta 28 palabras por línea.
Implementar Métodos Eficientes de Encriptación y Desencriptación	Desarrollar una función que encripte un mensaje sustituyendo cada carácter por el que está tres posiciones alfabéticas adelante (cifrado por desplazamiento).
Desarrollar Encriptación Personalizada	Crear una función que sustituya cada carácter del alfabeto por otro previamente decidido, utilizando una cadena de correspondencias.
Implementar la Encriptación de Minúsculas	Desarrollar una función que cambie todas las letras minúsculas por otras

	nuevas letras, según un array de caracteres cifrados.
Crear la Función de Decodificación de Mayúsculas	Desarrollar una función que descifre mensajes codificados utilizando una correspondencia específica de caracteres en mayúsculas.
Asegurar la Reutilización y Mantenibilidad del Código	Diseñar el código de manera modular y bien documentada, utilizando funciones separadas para cada tipo de encriptación y manipulación de texto.
Realizar Pruebas Unitarias y de Integración	Probar todas las funciones de manera individual y en conjunto para asegurar su correcto funcionamiento y eficiencia.
Garantizar la Consistencia de las Entradas de Datos	Implementar validaciones para asegurar que los datos ingresados por el usuario sean correctos y estén dentro del rango permitido, rechazando entradas inválidas y solicitando nuevos datos cuando sea necesario.

C. Alcance de proyecto:

1. Resultados del proyecto:

El proyecto se enfocará en el desarrollo de un menú de opciones en C++ para manipulación y encriptación de texto. Los resultados esperados son los siguientes:

Plan de trabajo:

El proyecto será dividido en 6 partes:
Creación de

Fase 1: Análisis de Requisitos y Planificación

Duración: 1 semana

Reunión de Inicio del Proyecto:

- Establecer objetivos y alcance del proyecto.
- Definir roles y responsabilidades (Programador, Tester).
- Crear el cronograma del proyecto.

Recolección de Requisitos:

- Identificar y documentar las funcionalidades del menú de opciones.
- Definir los requisitos de encriptación y manipulación de texto.
- Asegurar que los requisitos sean claros y completos.

Fase 2: Pruebas

Duración: 1 semana

Desarrollo de Casos de Prueba:

- Crear casos de prueba para cada funcionalidad implementada.

Pruebas Unitarias:

- Realizar pruebas unitarias para verificar el funcionamiento de cada función individualmente.

Pruebas de Integración:

- Realizar pruebas de integración para asegurar que todas las partes del sistema funcionen correctamente juntas.

Corrección de Errores:

- Identificar y corregir errores encontrados durante las pruebas.

Fase 3: Documentación y Entrega

Duración: 1 semana

Documentación del Proyecto:

- Elaborar la documentación del código.
- Crear un manual de usuario para la aplicación.

Revisión Final:

- Realizar una revisión final del proyecto con los stakeholders.
- Incorporar cualquier feedback final.

Entrega del Proyecto:

- Preparar y entregar el proyecto completo, incluyendo el código fuente y la documentación.

Informe Final del Proyecto:

- Redactar un informe final que resuma todo el proceso del proyecto, los logros y las lecciones aprendidas.

2. Contenido del Proyecto:**Fase 1: Análisis de Requisitos:**

- Establecer objetivos y alcance del proyecto:

1	Crear una interfaz de menú en la consola que permita al usuario seleccionar entre diferentes opciones de procesamiento de texto.
2	Implementar una función que lea líneas de texto, separe las palabras y las ordene en orden alfabético, manejando hasta 28 palabras por línea.
3	Desarrollar una función que encripte un mensaje sustituyendo cada carácter por el que está tres posiciones alfabéticas adelante (cifrado por desplazamiento).
4	Crear una función que sustituya cada carácter del alfabeto por otro previamente decidido, utilizando una cadena de correspondencias.
5	Desarrollar una función que cambie todas las letras minúsculas por otras nuevas letras, según un array de caracteres cifrados.
6	Desarrollar una función que descifre mensajes codificados utilizando una correspondencia específica de caracteres en mayúsculas.
7	Diseñar el código de manera modular y bien documentada, utilizando funciones separadas para cada tipo de encriptación y manipulación de texto.
8	Probar todas las funciones de manera individual y en conjunto para asegurar su correcto funcionamiento y eficiencia.
9	Implementar validaciones para asegurar que los datos ingresados por el usuario sean correctos y estén dentro del rango permitido, rechazando entradas inválidas y solicitando nuevos datos cuando sea necesario.

- Definir roles y responsabilidades (Programador, Tester):
Se acordó las siguientes responsabilidades para cada rol:

Programador:
<ul style="list-style-type: none"> - Implementación del menú de opciones y funcionalidades. - Validación de entradas de usuario.
Trabajo de cada programador:
<ul style="list-style-type: none"> • Aceituno Huaman Luis Geronimo: Creación del menú de opciones y de las funciones encargadas de la opción A y D. • Crisostomo Altamirano Axl Mikel: Creación de la función encargada de la opción C. • Torres Chuquiyauri Manuel Angel: Creación de la función encargada de la opción B. • Barrientos Torres Jose Paolo: Creación de la función encargada de la opción E.
Tester (Analista de Pruebas):
<ul style="list-style-type: none"> - Desarrollo de casos de prueba. - Ejecución de pruebas unitarias y de integración. - Reporte y seguimiento de errores.

- Crear el cronograma del proyecto:

Se acordó las siguientes fechas para la entrega del trabajo de cada integrante y de la revisión de estos trabajos:

Primera entrega de trabajo de cada integrante	04/06/2024
Primera revisión de las tareas de cada integrante	05/06/2024
Revisión de avance del proyecto	12/06/2024
Segunda entrega de trabajo de cada integrante	18/06/2024
Entrega del proyecto final	26/06/2024

- Recolección de Requisitos:

Después de la reunión entre los integrantes, se pudo recolectar los requisitos y sobre todo se pudo entender la parte del trabajo que le tocaba a cada uno. A continuación, se muestra lo recolectado:

Parte del proyecto	Requisitos
Función menú de opciones	Para la creación de menú se considera que se debe imprimir las 5 opciones mostradas en el proyecto, también el menú debe servir como un puente que

	sirva para que el usuario pueda llamar de manera directa a la función que se requiera. Adema se debe asegurar que el usuario ingrese una opción correcta para evitar errores
Función de la opción A	Para la creación de esta función, se debe pedir al usuario un texto que tenga hasta 27 palabras, luego mediante algoritmos se debe separar cada palabra, ordenarlos de manera alfabética. Se debe asegurar que el usuario no coloque más de 27 palabras, ya que es una condición del problema.
Función de la opción B	Para crear esta función, primero toma un mensaje de texto ingresado por el usuario y lo cifra desplazando cada letra tres posiciones hacia adelante en el alfabeto. Por ejemplo, la letra 'A' se convierte en 'D', la letra 'B' se convierte en 'E', y así sucesivamente. El programa utiliza un bucle para iterar a través de cada letra del mensaje y aplica la lógica de cifrado correspondiente. Finalmente, muestra el mensaje cifrado al usuario.
Función de la opción C	Para crear la función de cifrado, primero se leerá el número de conjuntos de datos a analizar. Para cada conjunto, se obtendrá el mensaje codificado y la cadena de caracteres que define la correspondencia entre las letras del alfabeto. Luego, para cada carácter del mensaje codificado, se encontrará su correspondiente en la cadena de mapeo y se reemplazará por el carácter original del alfabeto. Los mensajes decodificados se almacenarán y finalmente se imprimirán con su respectivo número de conjunto.
Función de la opción D	Para la creación de esta función, se debe almacenar mediante un array los 26 caracteres que reemplazaran a las letras del alfabeto en minúscula. Se debe priorizar que solo se reemplaza las letras minúsculas, también los caracteres ingresados en el array no se deben repetir.
Función de la opción E	Para la creación de esta función, primero lee el número de conjuntos de datos a analizar. Para cada conjunto, obtiene el mensaje codificado y la cadena de caracteres que define la correspondencia entre las letras del alfabeto. Luego, para cada carácter del mensaje codificado, encuentra su correspondiente en la cadena de mapeo y lo reemplaza por el carácter original del alfabeto. Los mensajes decodificados se almacenan y finalmente se imprimen con su respectivo número de conjunto.

Fase 2: Pruebas:

a) Función menú:

Para la función menú, primero declaramos la función opciones() antes de la función principal main(). Esta se define como una función void que no retorna ningún valor. A continuación, se muestra su declaración:

void opciones();

Dentro de la función opciones(), se presenta un menú con diversas opciones para el usuario, las cuales son:

- Lee líneas de texto, obtiene las palabras y las muestra en orden alfabético (máximo 28 palabras por línea).
- Encripta un mensaje sustituyendo cada carácter por el que está tres posiciones adelante en el alfabeto.
- Encripta un mensaje sustituyendo cada carácter según una cadena de correspondencias predefinida (mediante punteros).
- Encripta un texto sustituyendo letras minúsculas por otras según un array cifrado.
- Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- Salir.

El usuario debe elegir una opción ingresando una letra minúscula. La opción seleccionada se guarda en una variable y se maneja mediante un switch, permitiendo al usuario acceder a las funciones correspondientes. Se han definido cinco casos específicos:

- a: Accede a la función asociada a la opción A y luego retorna al menú de opciones.
- b: Accede a la función asociada a la opción B y luego retorna al menú de opciones.
- c: Accede a la función asociada a la opción C y luego retorna al menú de opciones.
- d: Accede a la función asociada a la opción D y luego retorna al menú de opciones.
- e: Accede a la función asociada a la opción E y luego retorna al menú de opciones.
- f: Finaliza el menú y termina el programa.

En caso de que el usuario no ingrese una opción válida, se activa el caso por defecto (default) del switch, solicitando nuevamente al usuario que ingrese una opción válida mediante una llamada recursiva a opciones().

Aquí está la implementación de la función opciones():

```
void opciones() {  
    char opcion;
```



```

do {
    cout << "\nMenu de Opciones: \n\n";
    cout << "a. Lee lineas de texto, obtiene las palabras y las muestra en orden
alfabetico (maximo 28 palabras por linea).\n";
    cout << "b. Encripta un mensaje sustituyendo cada caracter por el que
esta tres posiciones adelante en el alfabeto.\n";
    cout << "c. Encripta un mensaje sustituyendo cada caracter segun una
cadena de correspondencias predefinida (punteros).\n";
    cout << "d. Encripta un texto sustituyendo letras minusculas por otras
segun un array cifrado.\n";
    cout << "e. Encripta un mensaje sustituyendo cada letra por otra en todo el
mensaje.\n";
    cout << "f. Salir.\n\n";
    cout << "Elija que opcion desea (solo con letras minusculas): ";
    cin >> opcion;
    cin.ignore(); // Para limpiar el buffer de entrada

    switch (opcion) {
        case 'a':
            opcion_A();
            break;
        case 'b':
            opcion_B();
            break;
        case 'c':
            opcion_C();
            break;
        case 'd':
            opcion_D();
            break;
        case 'e':
            opcion_E();
            break;
        case 'f':
            cout << "Presione ENTER para cerrar la aplicacion.\n";
            break;
        default:
            cout << "\n\nIngrese una opcion valida: \n";
            break;
    }
} while (opcion != 'f');
}

```

también se adjuntará las pruebas que se le hizo a la función, cabe aclarar que las pruebas serán cuando no se ingresa una opción correcta, cuando seleccionas una opción y cuando te pide seleccionar otra opción.

Opción A:

Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas): a

Acacaba de ingresar a la opcion a.

Ingrese otra opcion: Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas):

Opción B:

Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas): b

Acacaba de ingresar a la opcion b.

Ingrese otra opcion: Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas):

Opción C:

Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas): c

Acacaba de ingresar a la opcion c.

Ingrese otra opcion: Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas):

Opción D:

Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas): d

Acacaba de ingresar a la opcion d.

Ingrese otra opcion: Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas): _

Opción E:

Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas): e

Acacaba de ingresar a la opcion e.

Ingrese otra opcion: Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas): _

Opción F:

Menu de Opciones:

- a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
- b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
- c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
- d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
- e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
- f. Salir.

Elija que opcion desea (solo con letras minusculas): f

Process exited after 3.162 seconds with return value 0
Presione una tecla para continuar . . . _

Opción Inválida:

```

Menu de Opciones:
a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
f. Salir.

Elija que opcion desea (solo con letras minusculas): Z

Ingrese una opcion valida:
Menu de Opciones:
a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
f. Salir.

Elija que opcion desea (solo con letras minusculas): _

```

//Explicar.

b) Función A:

Primero declaramos la funcion antes de la función principal main():

```
void opcion_A();
```

Luego definimos la función:

```

void opcion_A(){
    int e=0;
    string mayuscula="ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    string minuscula="abcdefghijklmnopqrstuvwxyz";
    string texto;
    string cadena[27];
    fflush(stdin);
    cout << "\nIngrese un texto para dividir: ";
    getline(cin,texto);
    for(int i=0;i<texto.length();i++){
        for(int k=0;k<26;k++){
            if(texto[i]==mayuscula[k]){
                texto[i]=minuscula[k];
            }
        }
    }
    istringstream palabra(texto);
    string salida;
    while (palabra >> cadena[e]) {
        e++;
        continue;
    }
    //Ordenar
    sort(cadena,cadena +e );
    for(int i=0;i<e;i++){
        cout<<i+1<<"..."<<cadena[i]<<endl;
    }
}

```

Explicación del código:

1. Primero, inicializamos la variable `e` a 0, la cual contará las palabras almacenadas en el vector `cadena`, que tiene un tamaño máximo de 28 palabras.
2. Definimos dos variables `string`: `mayusculas` y `minusculas`, que contienen todas las letras del alfabeto en mayúsculas y minúsculas, respectivamente. Además, creamos la variable `texto` para almacenar el texto ingresado por el usuario.
3. Utilizamos `getline(cin, texto)` para capturar todo el texto ingresado por el usuario, incluyendo espacios y caracteres especiales.
4. Un bucle `for` itera a través de cada carácter del texto (`texto.length()`), reemplazando cualquier letra mayúscula encontrada por su equivalente en minúscula, utilizando las variables `mayusculas` y `minusculas`.
5. Creamos un objeto `istringstream` llamado `palabra(texto)`, que convierte el texto en un flujo de entrada que podemos procesar palabra por palabra.
6. Usando un bucle `while` (`palabra >> cadena[e]`), leemos y almacenamos cada palabra del texto en el vector `cadena`. Este proceso continúa hasta que no hay más palabras para procesar.
7. Utilizamos `sort(cadena, cadena + e)` para ordenar alfabéticamente las palabras almacenadas en el vector `cadena`. Esto asegura que las palabras estén dispuestas en orden ascendente.
8. Finalmente, usamos un bucle `for` para imprimir cada palabra del vector `cadena`. Cada palabra se imprime junto con su posición en el orden ordenado.

Pruebas de la función A:

```
Menu de Opciones:
a. Lee líneas de texto, obtiene las palabras y las muestra en orden alfabético (maximo 28 palabras por línea).
b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
f. Salir.

Elija que opcion desea (solo con letras minusculas): a

Ingrese un texto para dividir: El sol dorado ilumina suavemente el mar tranquilo. las olas bailan bajo el cielo azul. Gaviotas planean en busca de su comida entre las rocas del acantilado. Gracias
1...acantilado.
2...azul.
3...bailan
4...bajo
5...busca
6...cielo
7...comida
8...de
9...del
10...dorado
11...el
12...el
13...el
14...en
15...entre
16...gaviotas
17...gracias
18...ilumina
19...las
20...las
21...mar
22...olas
23...planean
24...rocas
25...sol
26...su
27...suavemente
28...tranquilo.
```

c) Función B:

1. Se incluyen las bibliotecas necesarias para la entrada/salida y el manejo de cadenas de caracteres.
2. Se define la función `cifrarMensaje` que toma una cadena como parámetro y devuelve una cadena cifrada.
3. Se inicializa una cadena vacía donde se almacenará el mensaje cifrado.
4. Se recorre cada carácter del mensaje original utilizando un bucle `for`.

5. Se verifica si el carácter actual es una letra mayúscula.
6. Si el carácter es una letra mayúscula, se aplica una transformación de cifrado desplazando el carácter 3 posiciones en el alfabeto.
7. Se verifica si el carácter actual es una letra minúscula.
8. Si el carácter es una letra minúscula, se aplica una transformación de cifrado desplazando el carácter 3 posiciones en el alfabeto.
9. Si el carácter no es una letra, se añade sin cambios a la cadena cifrada.
10. Se devuelve la cadena cifrada una vez que todos los caracteres han sido procesados.
11. En la función main, se solicita al usuario que introduzca un mensaje.
12. Se lee el mensaje completo de la entrada estándar.
13. Se llama a la función cifrarMensaje con el mensaje introducido por el usuario.
14. Se muestra el mensaje cifrado en la salida estándar.

```
Menu de Opciones:
a. Lee líneas de texto, obtiene las palabras y las muestra en orden alfabético (máximo 28 palabras por línea).
b. Encripta un mensaje sustituyendo cada carácter por el que está tres posiciones adelante en el alfabeto.
c. Encripta un mensaje sustituyendo cada carácter según una cadena de correspondencias predefinida (punteros).
d. Encripta un texto sustituyendo letras minúsculas por otras según un array cifrado.
e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.

Elija que opción desea (solo con letras mayúsculas): b

Introduce el mensaje a cifrar: MANUEL
Mensaje cifrado: PDQXHO

-----
Process exited with return value 0
Press any key to continue . . . |
```

d) Función C:

Explicación código:

1. Inicializamos un array de caracteres mapaCorrespondencias con tamaño 26.
2. Utilizamos un bucle for para llenar mapaCorrespondencias con los caracteres de claveCorrespondencia.
3. Declaramos una estructura Indice con dos campos: index y length.
4. Inicializamos indice.index a 0 y indice.length con la longitud de mensajeOriginal.
5. Creamos un array dinámico mensajeCifrado con un tamaño igual a indice.length + 1.
6. Utilizamos un bucle for para iterar a través de cada carácter en mensajeOriginal:
 - Si el carácter está entre 'a' y 'z', lo reemplazamos con el carácter correspondiente en mapaCorrespondencias.
 - Si el carácter no es una letra minúscula, lo copiamos tal cual en mensajeCifrado.
7. Terminamos el array mensajeCifrado agregando un carácter nulo ('\0') al final.
8. Retornamos el puntero mensajeCifrado.
9. En la función opcion_C, solicitamos al usuario ingresar mensajeOriginal y claveCorrespondencia utilizando getline.
10. Llamamos a la función cifrarMensaje con los parámetros mensajeOriginal y claveCorrespondencia.

11. Imprimimos el mensaje cifrado y la ubicación del puntero mensajeCifrado.
12. Liberamos la memoria asignada a mensajeCifrado usando delete[].

Pruebas de la función C:

```
Menu de Opciones:
a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
f. Salir.

Elija que opcion desea (solo con letras minusculas): c
Ingrese el mensaje codificado: apruebame gf
Ingrese la clave de correspondencia: qwertyuiopasdfghjklzxcvbnm
Mensaje cifrado: 'qhktwqdt uy'
Ubicación del puntero: 0xcc5b00

-----
Process exited after 22.59 seconds with return value 0
Presione una tecla para continuar . . . |
```

e) Función D:

Primero declaramos la función antes de la función principal main():

```
void opcion_D();
```

Luego definimos la función:

```
void opcion_D(){
    string palabra,alfabeto="abcdefghijklmnopqrstuvwxyz";
    char codigo[26];
    for(int i=0;i<26;i++){
        cout<<"\nIngrese el caracter #"<<i+1<<" que sirvan para encriptar:
";cin>>codigo[i];
        for(int k=0;k<i;k++){
            if(codigo[i]==codigo[k]){
                while(codigo[i]==codigo[k]){
                    cout<<"\nIngrese un caracter no repetido:
";cin>>codigo[i];
                }
            }
            else{continue;}
        }
    }
    fflush(stdin);
    cout<<"Ingrese el texto que desea encriptar: ";getline(cin,palabra);
    //Sustitucion
    for(int i=0;i<99;i++){
        for(int k=0;k<26;k++){
            if(palabra[i]==alfabeto[k]){
                palabra[i]=codigo[k];
                break;
            }
            else{continue;}
        }
    }
    cout<<"\nEl texto encriptado es: "<<palabra;
}
}
```

Explicación del código:

- Declaramos una cadena alfabeto que contiene todas las letras del alfabeto en minúsculas.
- Definimos un arreglo codigo de caracteres con tamaño 26, donde se almacenarán los caracteres de sustitución para la encriptación.
- Utilizamos un bucle for para pedir al usuario que ingrese cada uno de los caracteres (codigo[i]) que se utilizarán para encriptar, comenzando desde el caracter 1 hasta el 26.
- Para asegurar que no haya caracteres repetidos en el arreglo codigo, dentro del bucle for anidado comprobamos si el caracter ingresado (codigo[i]) ya existe en las posiciones anteriores del arreglo. Si es así, pedimos al usuario que ingrese un caracter no repetido hasta que se cumpla esta condición.
- Solicitamos al usuario que ingrese el texto (palabra) que desea encriptar utilizando getline(cin, palabra);. Esto nos permite capturar todo el texto, incluyendo espacios y caracteres especiales.
- Utilizamos dos bucles for anidados para iterar a través de cada caracter del texto (palabra).
- En el bucle exterior (for(int i=0; i<99; i++)), recorremos cada posición del texto hasta un máximo de 99 caracteres (asumiendo que el texto no excederá esta longitud).
- En el bucle interior (for(int k=0; k<26; k++)), comparamos cada caracter del texto con las letras del alfabeto (alfabeto[k]). Cuando encontramos una coincidencia, sustituimos el caracter original por su correspondiente en el arreglo codigo.
- Una vez que encontramos la correspondencia y hacemos la sustitución, rompemos el bucle interno con break; para pasar al siguiente caracter del texto.
- Imprimimos en pantalla el texto encriptado utilizando cout<<"\nEl texto encritado es: "<<palabra;.

Pruebas de la función D:

```

Menu de Opciones:
a. Lee lineas de texto, obtiene las palabras y las muestra en orden alfabetico (maximo 28 palabras por linea).
b. Encripta un mensaje sustituyendo cada caracter por el que esta tres posiciones adelante en el alfabeto.
c. Encripta un mensaje sustituyendo cada caracter segun una cadena de correspondencias predefinida (punteros).
d. Encripta un texto sustituyendo letras minusculas por otras segun un array cifrado.
e. Encripta un mensaje sustituyendo cada letra por otra en todo el mensaje.
f. Salir.

Elija que opcion desea (solo con letras minusculas): d
Ingrese el caracter #1 que sirvan para encriptar: p
Ingrese el caracter #2 que sirvan para encriptar: d
Ingrese el caracter #3 que sirvan para encriptar: m
Ingrese el caracter #4 que sirvan para encriptar: g
Ingrese el caracter #5 que sirvan para encriptar: l
Ingrese el caracter #6 que sirvan para encriptar: s
Ingrese el caracter #7 que sirvan para encriptar: v
Ingrese el caracter #8 que sirvan para encriptar: x
Ingrese el caracter #9 que sirvan para encriptar: k
Ingrese el caracter #10 que sirvan para encriptar: b
Ingrese el caracter #11 que sirvan para encriptar: e
Ingrese el caracter #12 que sirvan para encriptar: i
Ingrese el caracter #13 que sirvan para encriptar: o
Ingrese el caracter #14 que sirvan para encriptar: f

Ingrese el caracter #15 que sirvan para encriptar: j
Ingrese el caracter #16 que sirvan para encriptar: q
Ingrese el caracter #17 que sirvan para encriptar: u
Ingrese el caracter #18 que sirvan para encriptar: r
Ingrese el caracter #19 que sirvan para encriptar: c
Ingrese el caracter #20 que sirvan para encriptar: t
Ingrese el caracter #21 que sirvan para encriptar: h
Ingrese el caracter #22 que sirvan para encriptar: z
Ingrese el caracter #23 que sirvan para encriptar: a
Ingrese el caracter #24 que sirvan para encriptar: w
Ingrese el caracter #25 que sirvan para encriptar: y
Ingrese el caracter #26 que sirvan para encriptar: n
Ingrese el texto que desea encriptar: ABC def GHI jkl MN

El texto encriptado es: ABC gls GHI bei MN

```

f) Función E:

Esta función pretende decodificar un mensaje con una cadena de referencia proporcionada por el usuario.

```
#include<iostream>
#include<stdlib.h>
#include<string>
#include<cstring>
#include <conio.h>

using namespace std;

int main(){
char ALFABETO[26] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U',
'V', 'W', 'X', 'Y', 'Z'};
char CODIGO[99];
char CADENA[99];
string DECO[99];
int longitud=0;
int NUM;

    cout<<"Ingresa el numero de conjuntos que quiere analizar: "<<endl; cin>>NUM;
    for(int h=0; h<NUM;h++){
        fflush(stdin);
        cout<<"\nIngresa el mensaje codificado: "<<endl; cin.getline(CODIGO,99);
        fflush(stdin);
        cout<<"\nIngresa la cadena: "<<endl; cin.getline(CADENA,99);

        longitud = strlen(CODIGO);

        for(int i=0; i<longitud;i++){
            for(int k=0;k<27;k++){
                if(CODIGO[i]==CADENA[k]){
                    CODIGO[i]=ALFABETO[k];
                    break;
                }
            }
            else {
                continue;
            }
        }
    }

    DECO[h]=CODIGO;
    for(int u=0; u<=99;u++){
        CODIGO[u]=' ';
    };

    longitud=0;
}

    for(int h=0; h<NUM;h++){
```

```

        cout<<h+1<<"-"<<DECO[h]<<endl;
    }

    getch();
    return 0;
}

```

Declaración de variables y arreglos:

- Se declara un arreglo `ALFABETO` que contiene las letras del alfabeto en mayúsculas.
- Se declaran los arreglos `CODIGO` y `CADENA` para almacenar el mensaje codificado y la cadena de referencia, respectivamente.
- Se declara un arreglo de strings `DECO` para almacenar los mensajes decodificados.
- Se declaran variables `longitud` para almacenar la longitud de la cadena `CODIGO` y `NUM` para el número de conjuntos que se van a analizar.

Solicitud de entrada al usuario:

- Se solicita al usuario que ingrese el número de conjuntos que desea analizar mediante `cin >> NUM`.

Bucle principal:

- Se inicia un bucle `for` que itera `NUM` veces para procesar cada conjunto.
- Se limpia el búfer de entrada usando `fflush(stdin)` para evitar problemas con la entrada de datos.
- Se solicita al usuario que ingrese el mensaje codificado y la cadena de referencia para cada conjunto.

Decodificación del mensaje:

- Se calcula la longitud de la cadena `CODIGO` utilizando `strlen(CODIGO)`.
- Se inicia un bucle anidado que recorre cada carácter de la cadena `CODIGO`.
- Para cada carácter, se busca su posición en la cadena `CADENA`.
- Si se encuentra la letra en la cadena `CADENA`, se reemplaza el carácter en `CODIGO` con la letra correspondiente del alfabeto.
- Se utiliza un segundo bucle `for` para limpiar el arreglo `CODIGO` después de decodificar el mensaje.

Almacenamiento del mensaje decodificado:

- El mensaje decodificado se almacena en el arreglo `DECO`.
- Se limpia el arreglo `CODIGO` para el próximo conjunto.

Impresión de los mensajes decodificados:

- Se imprime cada mensaje decodificado para cada conjunto.

Espera de entrada del usuario y finalización del programa:

- Por último, se utiliza `getch()` para esperar a que el usuario presione una tecla antes de finalizar el programa.

Se utiliza un enfoque de sustitución de letras para realizar la decodificación. Ahora se adjunta el arreglo de la función:

- a. Se pide el número de conjuntos que se desea analizar, en este caso como se pedía el ejercicio se haran 2 conjuntos

```
Ingresa el numero de conjuntos que quiere analizar:
2

Ingresa el mensaje codificado:
|
```

- b. Ingresamos el mensaje codificado y su cadena respectivamente. Recalcando que cada mensaje que usemos tiene su cadena específica.

```
Ingresa el mensaje codificado:
HPC PJVYMIY

Ingresa la cadena:
HLPRMF BASODGCVJZKWIYUNXQTE|
```

```
Ingresa el mensaje codificado:
FDY GAI BG UKMY

Ingresa la cadena:
CUOHYTSDBRAKEZPLJWGF MVQXIN
```

- c. Y como resultado se imprime el mensaje decodificado para cada conjunto respectivamente.


```
1-ACM CONTEST
2-THE SKY IS BLUE
|
```

3. Hipótesis o Suposiciones:

Se asume que los usuarios finales tendrán conocimientos básicos de uso de la consola y podrán interactuar correctamente con el menú de opciones. Además, se presupone que no habrá restricciones de recursos de hardware para la ejecución del programa.

4. Restricciones:

La única restricción del proyecto es que el menú de opciones debe funcionar en sistemas operativos Windows, sin considerar otras plataformas.

D. Planeamiento Inicial del Proyecto al alto nivel:

Estimación de recursos requeridos:

- 1 líder de proyecto
- 1 sublíder de proyecto
- 4 programadores
- 2 Tester

Estimación de fechas a programar:

- Fecha de inicio: 29/05/2024
- Fecha de entrega del avance del proyecto: 12/06/2024
- Fecha de entrega del proyecto final: 25/06/2024

E. Funciones del proyecto:

Función-Usuario

- Selección de Opciones del Menú: **Permitir al usuario seleccionar entre diferentes opciones de manipulación y encriptación de texto desde un menú interactivo en la consola.**
- Lectura y Ordenamiento de Texto: **Leer líneas de texto ingresadas por el usuario, extraer las palabras y mostrarlas en orden alfabético.**
- Encriptación por Desplazamiento: **Encriptar un mensaje sustituyendo cada carácter por el que está tres posiciones alfabéticas adelante, según la selección del usuario.**
- Encriptación Personalizada: **Encriptar un mensaje utilizando una cadena de correspondencias decidida de antemano, permitiendo al usuario ingresar el mensaje y la cadena de mapeo.**
- Encriptación de Minúsculas: **Encriptar un mensaje cambiando todas las letras minúsculas por otras letras nuevas definidas en un array proporcionado por el usuario.**
- Decodificación de Mayúsculas: **Decodificar mensajes codificados mediante una correspondencia específica de caracteres en mayúsculas, ingresada por el usuario.**

Función-Estado	
<ul style="list-style-type: none"> ○ Facilitación de Educación Tecnológica: Utilizar el proyecto como una herramienta educativa en instituciones públicas para enseñar conceptos básicos de programación y encriptación. ○ Promoción de la Alfabetización Digital: Fomentar el uso de herramientas tecnológicas entre la población, promoviendo la alfabetización digital y el aprendizaje de nuevas habilidades. ○ Seguridad y Privacidad: Desarrollar y promover métodos de encriptación sencillos para mejorar la seguridad y privacidad de la información en comunicaciones y datos personales. 	
Función-Empresa	
<ul style="list-style-type: none"> ○ Capacitación de Empleados: Usar el proyecto como base para la creación de recursos de capacitación básica para empleados en departamentos de TI. ○ Desarrollo de Prototipos: Implementar el proyecto como base para el desarrollo de prototipos de sistemas más complejos de encriptación y manipulación de datos. ○ Auditoría y Pruebas de Seguridad: Utilizar el proyecto como base para la creación de funciones de encriptación y decodificación como herramientas de auditoría para probar y mejorar la seguridad de sistemas de información corporativos. 	

F. Hitos del proyecto:

Hitos o evento significativo	Fecha programada
Repartimiento de tareas a cada integrante del grupo	29/05/2024
Primera revisión de las tareas de cada integrante	05/06/2024
Revisión del avance del proyecto	12/06/2024
Segunda revisión de las tareas de cada integrante	19/06/2024
Exposición y entrega del proyecto final	26/06/2024





G. Autoridad del proyecto:

- **Líder de proyecto: Crisóstomo Altamirano Axl Mikel**

H. Integrantes del equipo del proyecto, Roles y Responsabilidades:

1. Líder del proyecto: Crisóstomo Altamirano Axl Mikel
2. Sublíder del proyecto: Aceituno Huamán Luis Gerónimo
3. Programador 1: Crisostomo Altamirano Axl Mikel
4. Programador 2: Torres Chuquiyauri Manuel Torres
5. Programador 3: Barrientos Torres Jose Paolo
6. Programador 4: Aceituno Huaman Luis Geronimo
7. Tester 1: Torres Chuquiyauri Manuel Torres
8. Tester 2: Aceituno Huaman Luis Geronimo

I. Firmas:

Nombres			Códigos	Firmas
Crisostomo	Altamirano	Axl	23200165	
Mikel				
Barrientos	Torres	Jose Paolo	23200145	
Aceituno	Huaman	Luis	23200002	
Geronimo				
Torres	Chuquiyauri	Manuel	23200066	
Angel				