Guía de Ejercicios #11 - Manejo de Archivos

```
Advertencia

La resolución conjunta o grupal de los ejercicios aquí presentes no está
permitida, excepto en la medida en que puedas pedir ayuda a tus compañeros
de clase y a otras personas, y siempre que esa ayuda no se reduzca a que
otro haga el trabajo por vos.

El código fuente entregado por un estudiante debe ser escrito en su
totalidad por dicha persona.
```

Ejercicio 11.1

Escriba un programa que lea el archivo de texto “diodo.txt” y determine:

- Cantidad total de palabras
- Cantidad de veces que aparece la palabra “diodo”.

Para ello se debe escribir una función que permita buscar cualquier palabra e indique cuántas veces la encontró.

Nota: La función no debe distinguir entre mayúsculas y minúsculas.

Ejercicio 11.2

Escribir una función que reciba un archivo de texto dado y proceda a encriptar su contenido. El contenido encriptado lo escribirá en un archivo nuevo llamado ENCODED.txt que, de ya existir deberá sobreescribirse.

Nota: El algoritmo de cifrado a utilizar será muy sencillo: a cada caracter comprendido entre la a y la m se le sumará 13 y a cada caracter entre la n y la z se le restará 13.

Ejercicio 11.3

Escribir una función que reciba un archivo de texto dado y proceda a desencriptar su contenido. El contenido desencriptado lo escribirá en un archivo nuevo llamado DECODED.txt que, de ya existir deberá sobreescribirse.

Nota: El algoritmo de descifrado a utilizar será muy sencillo: a cada caracter comprendido entre la a y la m se le sumará 13 y a cada caracter entre la n y la z se le restará 13.

Ejercicio 11.4

Realizar un programa que reciba por argumentos del main un archivo de texto y dos palabras. Luego el programa analizará el contenido del archivo y reemplazará todos aquellos lugares donde aparezca la primer palabra por la segunda.

Ejercicio 11.5

Realice una función que reciba un archivo de texto y elimine tanto las vocales o las consonantes del mismo segun se indique en sus argumentos.

Ejercicio 11.6

Usando la siguiente estructura:

```
struct gente
{
char nombre[30];
char apellido[30];
int edad;
}
```

Realizar un programa que lea un archivo de texto .csv(separado por comas) y guarde esos datos en un array de estructuras de la estructura dada anteriormente. Una vez obtenidos los datos procederá a guardarlos en un archivo binario.

Nota: El nombre del archivo a leer deberá ingresarse por argumentos del main.

Ayuda: Para 'parcear' el archivo puede usar las funciones fread(); y strtok();

Ejercicio 11.7

Usando la siguiente estructura:

```
struct gente
{
char nombre[30];
char apellido[30];
int calificación;
}
```

Realizar un programa que lea un archivo de texto .csv(separado por comas) y guarde esos datos en un array de estructuras de la estructura dada anteriormente. Una vez obtenidos los datos se deberá informar cuántos chicos sacaron cada nota.

Nota: El rango de calificaciónes posibles es del 0 al 10, siendo el 0 un ausente.

Ejercicio 11.8

Utilizando la siguiente estructura:

```
typedef struct Cliente{
char usuario[30];
int clave;
}Cliente;
```

Se nos pide desarrollar un módulo de software que implemente la funcionalidad de poder volcar un arreglo estático de estructuras en un archivo binario y viceversa. Para esto se debera realizar un programa que solicite al usuario que ingrese datos de clientes por consola (como máximo 10), y luego se deberán grabar dichos datos en un archivo binario. Luego se deberá borrar el contenido del arreglo, y se volverá a cargar la información desde el archivo, para finalizar imprimiéndose en pantalla todos los datos cargados.

Ejercicio 11.9

Escribir un programa llamado mi-cat que tenga una funcionalidad similar a la del programa cat de Linux. Es decir, debe recibir el nombre de un archivo de texto como argumento por líınea de comandos (argumento del main), y debe imprimir a continuación su contenido en la consola.

Ejercicio 11.10

Para simplificar el desarrollo de una aplicación de login de usuarios, nos piden implementar las siguientes funciones:

- int leer linea(FILE* fp, char* linea);
- - Lee una línea del archivo (remueve el caracter \n y coloca \0). Retorna EXITO o ERROR en función del resultado obtenido.
- int escribir linea(FILE* fp, const char* linea);
- - Escribe una línea en el archivo al final del mismo (remueve el caracter \0 y coloca \n). Retorna EXITO o ERROR según el resultado.

Referencias

Algunos ejercicios fueron obtenidos y adaptados de:

- Guía de Trabajos Prácticos 2011 - Informática I - Departamento de Electrónica - UTN FRBA
- Guía de Trabajos Prácticos 2019 - Informática I - Departamento de Electrónica - UTN FRBA
- Guía de Ejercicios - Algoritmos y Programación I - UBA FIUBA