Ej1- Análisis palabras fichero

Realiza un programa que analice un fichero en búsqueda de palabras y vuelque el resultado en otro fichero.

La llamada al programa se debe realizar con los siguientes parámetros:

```
./analiza.exe <Nombre de fichero> <Palabras a buscar>
```

Condiciones del programa:

- El parámetro <palabras a buscar> es una secuencia de palabras separadas por espacio, de un número indeterminado. Usar las propiedades de argc/argv para almacenarlas correctamente.
- El fichero de salida tendrá el nombre del fichero de entrada añadiendo ".out" antes de la extensión. Si el nombre no tiene punto, se añadirá al final del nombre. (Ayuda, si no se consigue crear el nombre fichero o no se puede abrir, se volcará a la salida estándar). Por ejemplo:
    - El nombre del fichero de salida de lineas.txt sería lineas.out.txt
- Los errores se deberán dirigir a la salida de error (fichero ya abierto en programa stderror).
- Para cada palabra encontrada, el programa escribirá en el archivo de resultados:
    - La palabra
    - El número de ocurrencias
    - Números de las líneas en las que se encuentra la palabra.
- Se deberá utilizar memoria dinámica tanto para leer las líneas del fichero como para almacenar las estructuras y la información que sea necesaria de estas estructuras. No hay que almacenar el contenido de todo el fichero, solo la información relevante.
- El programa deberá liberar la memoria dinámica cuando ya no la necesite.

Se pide utilizar las siguientes dos estructuras:

Estructura para almacenar la **información de las palabras** que se van a buscar (`palabraInfo_t`):

Palabra

Número de ocurrencias (inicialmente, 0)

Lista de números de línea (array dinámico, inicialmente tamaño 0)

Estructura para almacenar la **lista de información (**`listaPalabras_t`**):**

Lista de palabras con su información (array con la estructura descrita anteriormente)

Número de palabras (tamaño útil del array almacenado en esta estructura)

Se pide utilizar las siguientes funciones:

```
//Esta función recibe una línea y el número de línea y modifica la lista de palabras para añadir la información encontrada.
```

```
void examinaLinea(char *lineaAux, int numLinea, listaPalabras_t
listaPalabras)

//Esta función obtiene la siguiente línea de un fichero ya abierto y devuelve
//un puntero a la cadena con memoria reservada dinámicamente.

char *leeLineaDinamicaFile(FILE *fd)


//Abre el fichero de escritura obteniendo el nombre a partir del nombre del
//fichero origen y escribe para cada palabra encontrada: la palabra, el
número //de ocurrencias y las líneas en las que aparece, si una palabra
aparece //varias veces en una línea aparecerá duplicada en la lista

void escribirResultado(listaPalabras_t lista, char *nombreFichOrigen)
```

Ejemplo de ejecución:

```
./AnalisisFichero buscar.txt que palabras noestan
```

Fichero de entrada (buscar.txt)

```
Aqui un fichero que podria ser mas largo pero que no lo es
con distintas lineas
algunas mas cortas que otras
y distintas palabras para poder buscar coincidencias
podemos alargarlo pero creo que esta claro
```

Fichero de salida (buscar.out.txt)

```
que. Numero de ocurrencias: 4. Lineas: 1 1 3 5
palabras. Numero de ocurrencias: 1. Lineas: 4
```

Ej2- Estrechar fichero

Crea un programa que copia un fichero de entrada a otro acortando la longitud de las líneas en el número de caracteres que se pasa como parámetro de entrada. Condiciones del programa:

- Utilizar fgets para leer las líneas del fichero, sin salirse de memoria (el tamaño máximo de una línea es el tamaño total del fichero).
- Parámetros de entrada (argc/argv):

```
<nombre_del_fichero_entrada> <nombre_del_fichero_salida> <numero_de_caracteres_maximo>
```

Las reglas de recorte son las siguientes:

- Una línea es una secuencia de caracteres acabada en espacio seguido de '\n'
- Se leen líneas completas de un fichero de entrada, el tamaño máximo de una línea en fichero es el tamaño total del fichero
- Dado el tamaño máximo de una línea, pasada por parámetros (argc/argv) al programa, si hay líneas con más caracteres que las indicadas, se dividen en dos líneas:
    - La línea cortada, el carácter siguiente al que corresponda con el tamaño máximo se sustituye por un '\n'
    - Los caracteres sobrantes se añaden al inicio de la línea siguiente
    - Si la siguiente línea más los caracteres añadidos sobrepasan el tamaño máximo de línea, se repite la operación.
- Condiciones de presentación de líneas
    - Si la línea cortada no termina en espacio, es posible que se hayan cortado palabras. Hay que seguir las reglas de párrafos cortados:
        - Si la continuación empieza por espacio, se omite el espacio (no se han cortado palabras)
        - Si la continuación no empieza por espacio, se entiende que la palabra se ha cortado y se añade un '-' antes del salto de línea de la línea cortada.

Ejemplo:

Fichero entrada: quijote.txt

```
En un lugar de la Mancha, de cuyo nombre
no quiero acordarme, no ha
mucho tiempo que vivia
un hidalgo de los de lanza en astillero,
adarga antigua, rocin flaco y galgo corredor.
```

```
./estrechar.exe quijote.txt quijote_recortado.txt 10
```

Fichero salida: quijote_recortado.txt

```
En un luga-
r de la Ma-
ncha, de c-
uyo nombre
no quiero
acordarme,
```

no ha muc-
ho tiempo
que vivia
un hidalgo
de los de
lanza en a-
stillero,
adarga ant-
igua, roci-
n flaco y
galgo corr-
edor.

Ej3-Registo de alumnos

Crea un programa que guarde en un fichero información de los alumnos. Las líneas del archivo deberán estar definidas de la forma:

Nombre:<nombre del alumno>,Apellido:<apellido del alumno>,Edad:<edad>

Ejemplo:

Nombre: Marta, Apellido: Alonso, Edad: 20
Nombre: Sergio, Apellido: Gomez, Edad: 19

El programa comenzará leyendo la información de un archivo llamado “alumnos.txt”:

- Puede estar vacío
- Si no existe, el programa lo creará

Por cada línea del archivo, siguiendo el formato mostrado previamente, leerá los datos de un alumno y los guardará en una lista redimensionada dinámicamente de alumnos. La estructura para almacenar los datos de un alumno seguirá el siguiente esquema:

- Nombre: Cadena de datos alfanuméricos
- Apellido: Cadena de datos alfanuméricos
- Edad: Dato numérico entero
- Estado: Dato que puede tener dos valores: guardado o sin guardar.

Se pide además, crear un menú para poder trabajar con los datos leídos de fichero y actualizarlos. Ofrecerá las siguientes opciones:

- Listar los alumnos
    - Mostrará el estado actual de alumnos en memoria de programa (los cargados de fichero y creados nuevos)
- Añadir alumno
    - Pide los datos de un nuevo alumno, y lo guarda al final de la lista de alumnos “en memoria” (aquí no se actualiza el fichero)
- Guardar alumnos
    - Actualiza el archivo de datos de alumnos usando la lista de alumnos cargados en memoria. Añade al fichero los nuevos alumnos (estado “sin guardar”) y cambia el estado del nuevo alumno escrito a “guardado”
- Buscar alumno
- Pide una cadena que puede ser tanto el nombre como el apellido (completos o parciales) de un alumno, y muestra todos los alumnos que coincidan con esa búsqueda