Fundamendez - 1C 2025 - Clase 15 (jueves)

Ejercicios de final

Ejercicio 1

Alex y Justin Russo (Los Hechiceros de Weberly Place) encontraron un libro de hechizos nuevos, donde había uno para crear portales. Abrieron portales y dentro de ellos abrían otros. Practicaron durante tanto tiempo que se perdieron.

Ayudalos a encontrar su camino a "casa", sabiendo que si recorren 20 portales se quedan sin magia, y no van a poder volver.

```
typedef struct portal{
    int col_siguiente;
    int fila_siguiente;
    bool es_seguro;
    char nombre_lugar[MAX_LUGARES]; // "casa", "selva", "supermercado"
} portal_t;
```

Se pide
Crear una función ***recursiva*** que reciba una matriz de *portal_t*, la cual representa los portales creados por los hermanos, y devuelva si pudieron volver a "casa" o no.

Aclaración: Se empieza a recorrer desde la posición (0,0) de la matriz.

Ejercicio 2

La fraternidad de Mike y Sully va a hacer una fiesta, necesitan armar la playlist. Para eso, todos los integrantes pusieron sus temas favoritos en un archivo, pero algunos tienen mal gusto musical, así que hay que filtrar esa lista para que sea cool.

Dado un archivo de canciones con el siguiente formato:

```
TITULO_CANCION;GENERO;DURACION;ARTISTA;CALIFICACION_SPOTIFY
```

Se pide crear un archivo con las canciones que sean cool. Una canción es cool cuando cumple con todas las siguientes condiciones:

- El título no contiene la palabra "triste".
- La duración es de menos de 3 minutos.
- Es de pop o de rock.
- Tiene calificación más de 7 en Spotify.
- Además, una canción siempre es cool cuando la canta María Becerra, sin importar los demás atributos.

Aclaración: La duración y la calificación son números enteros.

Ejercicio 3

Estamos en problemas, porque resulta que los recuerdos de Riley se están mezclando con los de Berni y los de Berni se están mezclando con los de Agus y… ya entendieron. Como si fuera poco, se mezclaron los recuerdos con las fantasías.

Los recuerdos y las fantasías de los humanos que estaban guardados en archivos se mezclaron todos con todos. Ahora se quiere un sistema para obtener los recuerdos de una persona.

Se pide: se tienen dos archivos, uno con las personas, con el siguiente formato:

```
ID_PERSONA;NOMBRE
```

Y otro archivo de los recuerdos con el siguiente formato:

```
ID_PERSONA;DESCRIPCION DEL RECUERDO;NIVEL_REALISMO
```

Un recuerdo pasa a ser fantasía cuando el nivel de realismo es 5 o menos.

Se pide implementar una función que, dados esos dos archivos y el nombre de una persona, busque los recuerdos (descartando las fantasías) de esa persona y los guarde en un nuevo archivo llamado recuerdos_persona.txt.

Ejemplo
Si se tienen los siguientes archivos y se quieren los recuerdos de Berni

personas.csv

```
891;Berni
172;Riley
549;Agus
```

recuerdos_fantasias.csv

```
549;Recibida;3
172;Cumpleaños de 10;9
891;Mudanza a capital;8
172;Participación en las olimpíadas;1
549;Le dan la contraseña de Google Meet;8
891;Recital de Karina;7
172;Partido de hockey;8
891;Conoce a Moria;2
```

recuerdos_persona.txt

```
Mudanza a capital
Recital de Karina
```

Ejercicio 4

En la mente de Riley se organizan los recuerdos a medida que van llegando, y se los va poniendo en un tubo para que lleguen a la memoria a largo plazo.
Un recuerdo se representa con el siguiente struct:

```
typedef struct recuerdo {
    int id;
    char tipo[MAX_NOMBRE]; //"trauma", "feliz", "triste", "indiferente"
} recuerdo_t;
```

El tubo se puede pensar como que tiene un principio y un final. Los recuerdos entran por el principio de a uno y salen por el final en el mismo orden en el que entraron, no se puede sacar un recuerdo por el principio del tubo.

Para representar esto, se tiene un TDA "tubo" que guarda los recuerdos. El TDA cuenta con las siguientes primitivas:

```
// Pre: -
// Post: Crea un TDA tubo en el heap y devuelve un puntero al
// mismo. Devuelve NULL si no lo pudo crear.
tubo_t *tubo_crear();

// Pre: El tubo recibido fue previamente creado con `tubo_crear`.
// Post: Libera la memoria que se reservó en el heap para el tubo.
void tubo_destruir(tubo_t *tubo);

// Pre: El tubo recibido fue previamente creado con `tubo_crear`.
// Post: Agrega un reporte al principio del tubo.
bool tubo_agregar_al_principio(tubo_t *vector, recuerdo_t nuevo_elemento);

// Pre: El tubo recibido fue previamente creado con `tubo_crear`.
// Post: Se eliminará el primer recuerdo en el final del tubo.
// Devuelve true si se eliminó correctamente o false si el tubo
// estaba vacío y no se pudo eliminar nada. En caso de devolver true,
// el elemento que se acaba de eliminar será devuelto por referencia
// mediante el parámetro `elemento_eliminado`.
bool tubo_eliminar_al_final(tubo_t *tubo, recuerdo_t *elemento_eliminado);
```

Hubo un problema, y se infiltró un trauma en el tubo de los recuerdos. Alegría quiere sacarlo sí o sí.

Se pide
Hacer una función que reciba un `tubo_t` creado que encuentre el trauma y lo elimine. Al finalizar, el tubo tiene que **quedar con los mismos recuerdos que con los que arrancó pero sin el trauma**.

Aclaraciones
- Solo hay un trauma en el tubo.