Ejercicio 1

La Mansión Foster para amigos imaginarios tiene cierta organización, guarda en un vector los amigos que estan presentes en la casa, y en otro, los amigos que aún están con sus creadores (es decir, no están en la casa). Sabemos que ambos vectores están ordenados por nombre, alfabéticamente.

La estructura amigo, tiene la forma:

```
typedef struct amigo {
    char nombre[MAX_NOMBRE];
    int id_amigo;
    char nombre_creador[MAX_NOMBRE];
} amigo_t;
```

SE PIDE

1. Hacer un procedimiento, que dado el vector de amigos que están en la casa, el vector de los que no están y un id, elimine el amigo del vector de los que NO están en la casa, y lo agregue en el vector en los que SI. Manteniendo el orden de ambos vectores.

Ejercicio 2

Polar es un oso callado pero le gusta mucho la música. Tiene una playlist enorme con todas las canciones que le gusta pero hay un problema con eso, cuando anda medio bajón solo quiere escuchar canciones que vayan con su estado de ánimo. Es por eso que nos pidió ayuda para armar una playlist nueva con las canciones "Pal Bajón".

Se cuenta con un **archivo binario de acceso secuencial** con todas las canciones que le gustan a Polar, con la estructura del tipo `cancion_t`:

```
typedef struct cancion{
   char nombre[MAX_NOMBRE];
   int duración; //En segundos
   char artista[MAX_ARTISTA];
   char estado_animo[MAX_ANIMO];
} cancion_t;
```

- *Aclaración:* El archivo está desordenado pero se sabe que no hay más de 50 canciones para cada estado de ánimo.

Se pide

1. Crear un segundo archivo que solo contenga las canciones con estado de animo "Bajón". **Este segundo archivo tiene que estar ordenado por el nombre de las canciones.**

Ejercicio 3

Maurice es una excelente cocinera y tiene un libro con todas sus recetas. Esta noche quiere hacer su famosa tarta de berenjenas, pero no sabe si tiene todos los ingredientes necesarios para preparar el plato.

Se cuenta con los siguientes archivos:

- `libro_recetas.dat`: Representa el libro de recetas de Maurice. Es un **archivo binario de acceso secuencial** que contiene elementos del tipo `receta_t`:

  ```
  typedef struct receta{
     char nombre[MAX_NOMBRE];
     int id_ingredientes[MAX_INGREDIENTES];
     int cantidades[MAX_INGREDIENTES];
     int tope;
     bool es_postre;
  } receta_t;
  ```

  - El campo `id_ingredientes` es un vector que contiene los ids de todos los ingredientes de la receta. El vector `cantidades` contiene las cantidades de cada ingrediente, ambos vectores son del mismo tamaño asi que comparten el tope.

- `stock_ingredientes.dat`: Contiene el stock de todos los ingredientes en la casa. Es un **archivo binario de acceso directo** que contiene elementos del tipo `ingrediente_t`:

  ```
  typedef struct ingrediente{
     int id;
     char nombre[MAX_NOMBRE];
     int cantidad;
  } ingrediente_t;
  ```

  - El archivo está ordenado por `id` de forma ascendente. Los `id` son números enteros correlativos que comienzan desde el 0.

Se pide

1. Implementar una función que devuelva `true` si Maurice tiene los ingredientes necesarios para cocinar su "Tarta de berenjenas" o `false` en caso contrario.

Ejercicio 4

Puro Hueso está un poco vago últimamente, y no está recolectando las almas que tiene como responsabilidad recolectar, es por eso que necesita un empujoncito de nuestra parte. Tiene todas las almas por recolectar en un vector de almas ordenado alfabéticamente por nombre, asi que las vamos a buscar por él.

Una alma se almacena en una estructura de la forma:

```
typedef struct alma {
    int vejez;
    int prioridad;
    char nombre[MAX_NOMBRE];
} alma_t;
```

Se pide

1. Implementar una búsqueda binaria recursiva que devuelva la posición de un alma, se puede asumir que los nombres son únicos.

Ejercicio 5

1. Qué ventaja tienen los archivos binarios frente a los de texto? y viceversa?
2. Cuáles son las limitaciones que tenemos al trabajar con vectores, que podemos solucionar con el uso de archivos? (*Nombre mínimo dos*)
3. Qué cosa nunca nos tenemos que olvidar al trabajar con archivos en **C**? Que puede pasar si finalizamos el programa sin hacer esto?