Ejercicio 1

Coraje tuvo una buena caza de huesos durante sus vacaciones. Recolectó taaantos taaaantos huesos, que tuvo que hacer un archivo de texto llamado *huesitos.csv* para poder guardar la información sobre estos. El archivo se encuentra ordenado por zona descendentemente, luego por fecha ascendentemente, y finalmente por hueso alfabéticamente. Ahora que ya está tranquilo en su casa, antes de enterrarlos, quiere clasificarlos para tener un mayor control, y saber hasta cuando los puede comer sin que le caigan mal. Cómo anduvo por muchos lados, primero le gustaría clasificarlos por lugar y luego por fecha.

Se pide

1.Clasificar a los huesos agrupados primero por zona en dónde los encontró, y luego por fecha, como se muestra a continuación:

```
---------------------------------------
Palermo
    01/01/21
        Hueso blanco
        Hueso duro
        ..
    05/01/21
        Hueso caracu
        Hueso chico
        Hueso de oro
    ..
---------------------------------------
Adrogue
    02/02/21
        Hueso de colección
        Hueso grande
        ...
    05/04/21
        Hueso negro
    ...
```

Ejercicio 2

Las chicas superpoderosas quieren tener en su cuarto un rompecabezas de la ciudad de Saltadilla. Ya sacaron todas las fichas y las pusieron, de manera desordenada, dentro de un marco para que no se les pierda ninguna pieza.

Justo cuando estaban por empezar a armarlo, el alcalde las llamó por una emergencia, por lo cual te piden a vos si lo podes armar.

Teniendo una matriz donde cada celda representa una pieza, y la estructura de la pieza definida como:

```
typedef struct pieza {
    int fil;
    int col;
    int id;
} pieza_t;
```

donde *fil* corresponde a la fila en donde debería estar, y *col* a la columna donde debería estar:

Se pide

1. Crear una matriz nueva donde quede el rompecabezas completamente armado.

Ejercicio 3

Todos los directores de Cartoon Network se juntaron para definir quién o quiénes fueron los que más combatieron el crimen, para darles el Cartoon Award del año. Como son tantos, volcaron de manera ascendente en un archivo de texto *heroes.csv* todos los nombres por los cuales son reconocidos los que alguna vez salvaron al mundo. Por otra parte, tienen un archivo binario *identidades.dat* donde se revela la verdadera identidad de ese héroe. Este archivo se encuentra ordenado de manera ascendente por nombre del héroe, con la siguiente estructura:

```
typedef struct heroe {
    char nombre_real [MAX_NOMBRE];
    char nombre_heroe [MAX_NOMBRE];
    int crimenes_combatidos;
} heroe_t;
```

Se pide

1. Crear un nuevo archivo de texto *heroes_revelados.txt* donde se muestre a los héroes como se especifica a continuación:

```
nombre_real;crimenes_combatidos
```

2. Si los directores quisieran saber el total de crímenes combatidos en todo Cartoon Network, ¿Podrían hacerlo? Si es así, ¿Qué tendrías que agregar al código para poder llevarlo a cabo?

Ejercicio 4

Tom y Jerry están tratando de recordar qué contraseña habían puesto a su caja fuerte, para poder sacar sus más preciados tesoros que tienen guardados. Como ambos tienen muy mala memoria, pensaron que sería una buena idea que cada uno hiciera un vector con las contraseñas que se les ocurrieran para probar todas ellas.

Luego de escribir ambos vectores, se acordaron que la contraseña que habían puesto, no superaba los 10 caracteres.

Se pide

1. Sabiendo que ambos vectores están ordenados de manera ascendente, crear un nuevo vector que **una** todas las posibles contraseñas, sin contar las repetidas, ni las que tienen más de 10 caracteres.

Justo cuando les quedaba un último intento antes de que se bloquee la caja fuerte, encontraron un papel que tenía escrito algo que podía ser, o no, la contraseña. Antes de intentar con esa y arriesgarse al bloqueo, decidieron primero fijarse si ya no la habían probado.

2. Teniendo el vector creado en el punto anterior, crear una función que devuelva *true* en caso de ya haberla probado, o *false* en caso de que no.

Ejercicio 5

1. Explicar el ordenamiento por inserción, detallando paso a paso con la palabra **MANDY**. No se puede utilizar **ninguna** palabra reservada por C.
2. Si un vector está desordenado y solo se tiene que buscar un elemento... ¿Es conveniente ordenarlo? ¿Por qué?
3. ¿Qué diferencias hay entre la búsqueda binaria y la búsqueda lineal? ¿En qué casos se aplica cada una?