Ejercicio 1

Parte práctica

Annie y Hallie deciden realizar una broma a Meredith en el mismo hotel donde hicieron que se reencuentren sus padres (Elizabeth y Nicholas).

La broma resultará exitosa si:

- Annie y Hallie están cerca, es decir se encuentran en el mismo piso.
- No son rodeadas por sus padres, es decir su padre (Nicholas) y su madre (Elizabeth) no se encuentran en los pisos contiguos del piso en que están ellas, uno en cada uno.
- El padre (Nicholas) no ve la broma durante su ejecución, es decir Meredith no se encuentra en la misma habitación que Nicholas.

Si el hotel puede ser representado como una matriz de habitaciones donde una fila representa un piso, y una habitación puede ser representada con la siguiente estructura:

```
typedef struct habitacion{
    int numero;
    char ocupantes[MAX_OCUPANTES]; // inicial del nombre
    int cantidad_ocupantes;
} habitacion_t;
```

Se pide

Dada una matriz de habitaciones que representa el hotel en el instante que se ejecuta la broma, crear una función que devuelva **true** si la broma resulta exitosa y **false** en caso contrario.

Ejemplos

Con esta disposición de personas:

```
|-|-|-|-|-|M|-|
|-|-|-|-|-|-|-|
|-|-|A|-|-|-|-|
|-|-|H|-|-|-|-|
|N|-|-|-|-|-|-|
|-|-|-|E|-|-|-|
```

Devolverá **false**.

Con esta disposición de personas:

```
|-|-|--|-|-|-|-|
|-|-|--|-|-|-|-|
|-|-|--|E|-|-|-|
|-|-|HA|-|-|M|-|
|-|-|--|-|-|N|-|
|-|-|--|-|-|-|-|
```

Devolverá **false**.

Con esta disposición de personas:

```
|-|-|-|-|-|--|-|
|-|-|-|-|-|--|-|
|-|-|-|-|-|--|-|
|-|-|H|A|-|--|-|
|-|-|-|-|-|NM|-|
|-|-|E|-|-|--|-|
```

Devolverá **false**.

Con esta disposición de personas:

```
|N|-|-|-|-|M|-|
|-|-|-|-|-|-|-|
|-|-|A|-|-|H|-|
|-|-|-|-|-|-|-|
|-|-|-|-|-|-|-|
|-|-|-|-|E|-|-|
```

Devolverá **true**.

Parte teórica

1. ¿Qué son las pre condiciones de una función? ¿Qué son las post condiciones de una función? Dar ejemplos.
2. Explicar si la siguiente afirmación es Verdadera o Falsa y por qué: "Dado un vector y su tope, al acceder al elemento `vector[tope]` estoy accediendo a basura".

Ejercicio 2

Parte práctica

Luego de su misión, Bob tiene que ir al supermercado a comprar las cosas que Helen le anotó en una lista.

Como no confía mucho en Bob, Helen siempre chequea el ticket con la lista, para ver si falta algo.

Se pide

Se tienen dos archivos, uno es la lista de compras y otro el ticket, ambos con el siguiente formato:

```
PRODUCTO;CANTIDAD
```

Se debe crear programa en el lenguaje de programación **Python** que cree un archivo con los productos que estén en la lista de compras y **no** estén en el ticket (o sea, que Bob se olvidó de comprar), dicho archivo tendrá el mismo formato que los otros dos.

Aclaración

- El archivo **ticket** es pequeño y se puede asumir que entra en memoria.
- Si el producto se encuentra en el ticket, pero la cantidad que Bob compró es menor a la que estaba en la lista, debe agregarse al nuevo archivo la cantidad que falta de ese producto.
- Si hay más de un artículo en el ticket de lo que se especificó en la lista entonces no hay que incluirlo en los faltantes.

Ejemplo

Teniendo los archivos

lista_compras.csv

```
BATATA;4
YOGURT;5
LEVITE NARANJA;3
```

ticket.csv

```
BATATA;4
LEVITE NARANJA;1
```

Se obtiene el siguiente archivo:

faltantes.csv

```
YOGURT;5
LEVITE NARANJA;2
```

Parte teórica

1. ¿Cuáles son las operaciones con vectores que vimos en clase? ¿Cómo se diferencian entre sí?

Ejercicio 3

Parte práctica

Linguini debe realizar un pedido de materia prima para el Restaurante de Gusteau, pero antes debe revisar el stock actual.

El restaurante quiere tener en stock cantidad 100 de cada producto y no tienen en cuenta los productos que le queden 2 o menos días para que caduquen.

Se pide

Dado un archivo de **stock** con el siguiente formato:

```
DIAS_PARA_CADUCAR;PRODUCTO;CANTIDAD
```

Se debe crear un programa en el lenguaje de programación **Python** que cree un archivo **pedido** con los productos y cantidades necesarias para que Linguini pueda encargar el pedido.

Formato del archivo **pedido**:

```
PRODUCTO;CANTIDAD
```

Aclaración

- Para el archivo **stock** se puede asumir que no hay productos repetidos.

Ejemplo

Teniendo el archivo

stock.csv

```
4;YOGURT;67
1;BATATA;40
60;LEVITE NARANJA;100
10;PAPA;3
```

Se obtiene el siguiente archivo:

pedido.csv

```
YOGURT;33
BATATA;100
PAPA;97
```

Parte teórica

¿Cuál es la diferencia entre un vector dinámico y una lista enlazada? Mencione una ventaja y una desventaja del uso de cada una.

Ejercicio 4

Parte práctica

Mr. Increíble tiene un montonazo de supertrajes diseñados por Edna. Él se los pide siempre en color rojo porque es su color favorito, pero Edna se los hace en varios colores porque sino se aburre de usar siempre el mismo.

Mañana Mr. Increíble debe salir a una misión y quiere llevar su traje más nuevo, y tiene que ser el rojo, obvio.

Un supertraje está representado por el struct `supertraje_t` :

```
typedef struct supertraje {
    int antiguedad;
    char color[MAX_COLOR];
    bool es_ganador;
} supertraje_t;
```

Se pide:

Dado un vector de supertrajes que contiene todos los supertrajes de Mr. Increíble y su tope, crear una función **recursiva** que devuelva la posición del traje más nuevo, que sea de color rojo.

Parte teórica

1. ¿Qué elementos debe tener una función recursiva?
2. ¿Qué tipo de dato es el campo **color** del struct supertraje? ¿Qué particularidad tiene ese tipo de dato?

Ejercicio 5

Parte práctica

Minnie tiene una huerta de la que está muy orgullosa. Por eso para la cena quiere hacerle una sorpresa a Mickey y cocinarle algo con verduras y especias cultivadas por ella misma. La huerta puede ser representada como una matriz de cultivos, y cada cultivo puede representarse con el siguiente registro:

```
typedef struct cultivo {
    char nombre[MAX_NOMBRE];
    char tipo[MAX_TIPO]; // "especia", "verdura", "fruta", "flor"
    bool maduro;
    int peso;
} cultivo_t;
```

Además para cosechar, Minnie tiene una canasta, que puede ser representada como un vector de `cultivo_t` .

Aclaraciones

- Se puede asumir que la matriz de cultivos está llena.
- No es necesario borrar los cultivos recogidos de la matriz.
- Se asume que el vector canasta está vacío.

Se pide

Dada la huerta y la canasta con su tope, se pide crear un procedimiento que guarde en el vector canasta aquellos cultivos de la huerta que sean de tipo **especia** o **verdura** y que además esté maduro.

Parte teórica

¿En qué consiste la inserción ordenada? Explicar detalladamente.

Ejercicio 6

Parte práctica

Anastasia, la hermanastra de Cenicienta, se puso de novia con el panadero del pueblo que siempre le regala medialunas y la cuida. Los días más ocupados de la panadería son los fines de semana, y como Anastasia lo quiere y quiere pasar tiempo con él, decide ayudarlo a acomodar las cuentas de este fin de semana.

Se pide

Dado un archivo csv con todas las **ventas** que hizo el panadero de **viernes a domingo** (ambos inclusive), con la siguiente estructura:

```
DIA;MONTO;COBRADO
```

donde `COBRADO` puede ser "C" (cobrado) o "D" (deuda), y otro archivo con los **gastos** que tuvo esos días con el siguiente formato:

```
DIA;MONTO
```

Se pide crear otro archivo csv que contenga las ganancias que se obtuvieron por día el fin de semana con el siguiente formato:

```
DIA;MONTO
```

Aclaraciones

- No se tienen en cuenta los montos que no fueron cobrados aún.
- Se tienen en cuenta todos los gastos.
- Los montos en los archivos de ventas y gastos siempre se guardan como un número positivo.
- Los días de la semana están guardados en minúscula.

ventas.csv

```
domingo;300;D
sábado;1000;C
viernes;400;C
domingo;400;C
sábado;700;D
domingo;800;C
```

gastos.csv

```
domingo;300
viernes;500
domingo;100
sábado;600
```

cuentas.csv

```
viernes;-100
sábado;400
domingo;800
```

Parte teórica

1. Explicar cuándo el contenido de un archivo puede guardarse en su totalidad en alguna estructura como lo es un arreglo o una lista.
2. ¿Qué son las pre y post condiciones? ¿Por qué es importante usarlas? Escribir las pre y post condiciones para la siguiente función:

```
int division(int dividendo, int divisor){
    return dividendo / divisor;
}
```