Algoritmos y Programación I (95.11) – Curso Essaya – 4to parcialito – 28/11/2022

Resolver los siguientes problemas en forma clara y legible en código ISO-C99.

1. Teniendo representado el TDA lista genérica como `typedef struct {struct nodo *prim;} lista_t;` y el nodo `struct nodo {struct nodo *sig; void *dato;};` implementar una primitiva

   ```
   void *lista_borrar(lista_t *l, void *e, int (*cmp)(void *a, void *b));
   ```

   que busque la primera ocurrencia del elemento `e` en la lista y elimine ese nodo devolviendo el elemento. La función `cmp` devuelve `0` si dos elementos son iguales.

2. Se inventariaron todos los mármoles de la facultad y se les asignó un número a cada uno de ellos. Todos los mármoles que necesitan ser reparados se guardaron en un vector `marmoles_peligrosos` ordenados por número. Al final de cada día se elabora un vector `marmoles_reparados` con los mármoles que se repararon ese día también ordenado. Queremos obtener el listado de los mármoles que siguen sin repararse **de forma eficiente**, para eso necesitamos una función:

   ```
   int *marmoles_todavia_peligrosos(int marmoles_peligrosos[], size_t n_marmoles_peligrosos,
       int marmoles_reparados[], size_t n_marmoles_reparados,
       size_t *n_marmoles_todavia_peligrosos);
   ```

   que devuelva el vector y la cantidad de mármoles que todavía son peligrosos.

   Nota 1: No se puede asumir que `n_marmoles_reparados` siempre valga 0, este es un ejercicio ficticio.

   Nota 2: Por si alguien ya no se acuerda, el problema de los mármoles era el que teníamos antes del problema de las inundaciones.

3. Implementar una función `float sumar_elementos(float v[], size_t n);` que utilizando recursividad calcule la suma de los elementos de un vector `v` de `n` flotantes.

¡Suerte! :)