Algoritmos y Programación I (95.11) – Curso Santisi – 4to parcialito – 27/11/2023

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

1. Se tiene una lista enlazada genérica definida con su nodo como `struct nodo { void *dato; struct nodo *sig; };` y el TDA como `typedef struct { struct nodo *prim } lista_t;`.

   Implementar una primitiva:

   ```
   lista_t *lista_clonar(const lista_t *l, void *(*clonar_dato)(const void *));
   ```

   que dada una lista `l` devuelva una copia de la misma, con sus datos también clonados.

   Se puede asumir que no va a haber fallas de memoria al realizar la operación.

2. En el lenguaje C las fechas se codifican con el tipo `time_t` el cual es un *"tipo aritmético"*, es decir, más allá de la abstracción de qué contienen los `time_t` son variables enteras que pueden ser comparadas y ordenadas.

   Se quiere organizar un evento. Para eso se le pidió a los `n` asistentes que listen las fechas en las que pueden asistir. Cada uno generó un vector de `time_t` ordenado por tiempo creciente.

   Implementar una función:

   ```
   time_t *interseccion_de_fechas(const time_t **fechas, size_t nfechas[], size_t n,
                                                                  size_t *ninterseccion);
   ```

   que reciba `fechas` un vector de `n` vectores, donde cada `fechas[i][]` tiene longitud `nfechas[i]` y devuelva un vector con la intersección de todas las fechas computado **de forma eficiente** y la cantidad de fechas en `ninterseccion`.

   Nota: La intersección de los `n` rangos de `fechas` se puede hacer de a pares.

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

¡Suerte! :)