UT 5: Estructuras de datos dinámicas

Tarea 2

1. Declara dos punteros a float. Reserva memoria para cada uno de los punteros. Asigna valores para cada uno de los punteros, imprime tanto la dirección apuntada por el puntero como el valor, libera la memoria de ambos punteros. Imprime la dirección de memoria y el valor apuntado por ambos punteros, ¿Qué ocurre?
2. Repite el ejercicio anterior y justo después de hacer el free, asigna NULL a ambos punteros. Imprime la dirección de memoria y el valor apuntado por ambos punteros. ¿Qué ocurre?Declara un puntero a float. Declara un float y asignale un valor. Haz que el puntero apunte a la variable e imprime su valor. Decrementa en uno el valor apuntado por el puntero e imprime la variable float. ¿Es necesario asignar y liberar espacio para este puntero?
3. Declara dos punteros d1 y d2, ambos a float. Reserva memoria para 10 valores apuntados sólo para d1. Haz que d2 recorra todos los valores reservados para asignar a todos ellos el valor de los 10 primeros números pares. Una vez termines de inicializar d1, vuelve a recorrer d1 utilizando d2 imprimiendo todos los valores. Libera la memoria al finalizar el programa. ¿Cuántos free() tienes que realizar?
4. Repite el ejercicio anterior pero una vez que has rellenado los valores imprime por pantalla en orden inverso los valores por pantalla.
5. Realiza un programa en el que declares un array (estático) de 10 floats. Inicialízalo con valores 40, 41, ...49. Declara también un puntero a un número en coma flotante. Recorre el array estático con el puntero que has creado para que imprima todos los datos..
6. Dadas las estructuras de datos siguientes implementa un programa que tengas las siguientes opciones:
    a. Introducir datos del día: Sólo se guardará una medición meteorológica.
    b. Mostrar datos registrados: Comprobar antes si existen datos.
    c. Quitar datos:Liberar espacio en caso de que existan datos.
    d. Salir

    La estructura de datos:

```
// Estructura para guardar una hora
 typedef struct
 {
   int hora;
   int min;
 } Hora;

 // Estructura para guardar los datos meteorológicos de un día
 typedef struct {
   float tempMedia;
   Hora salidaSol;
   Hora puestaSol;
 } DatosMeteo;
```

Notas a tener en cuenta:

- Declara un puntero a la estructura de datos.
- Sólo se guardará una medición meteorológica.