Ejercicios de búsqueda y ordenación

OBJETIVOS

El objetivo de estos ejercicios es repasar los conceptos de búsqueda en arrays.

En la página de UAMX de la asignatura dispones del esqueleto del código para poder trabajar con estos ejercicios. El examen estará formado por dos ejercicios similares a estos.

Ejercicio estaOrdenado

Crear una función que compruebe si un array está ordenado.

```
int estaOrdenada(int *lista, int tamLista);
```

Ejemplo de ejecución: Tras llamar a la función con el array 3 1 2 6 5 4 7 , la función debe devolver 0.

Ejercicio búsqueda lineal

Crear una función que busque un valor en un array mediante búsqueda lineal. La función debe devolver el índice del valor en la lista si se encuentra el 'valor' en la 'lista' o -1 en caso contrario.

```
int busquedaLineal(int valor, int *lista, int tamLista);
```

Ejercicio búsqueda binaria

Crear una función que busque un valor en un array mediante búsqueda binaria. La función debe devolver el índice del valor en la lista si se encuentra el 'valor' en la 'lista' o -1 en caso contrario.

```
int busquedaBinaria(int valor, int *lista, int int, int, sup);
```

Ejercicio función hash por módulo

Crear una función hash que devuelva el valor hash de una clave numérica mediante la operación de módulo.

```
int hash_modulo(int tamanio_tabla, int clave);
```

Ejercicio función hash por el método de la multiplicación

Crear una función hash que devuelva el valor hash de una clave numérica mediante la operación de de multiplicación.

```
int hash_multiplicacion(int tamanio_tabla, int clave);
```

Ejercicio medir tiempos búsqueda lineal

Crear una función que mida el tiempo que tarda en buscar todos los valores de una tabla de tamaño 'tamanio_tabla' usando búsqueda lineal. Para ello primero se debe crear una array de números enteros y luego buscar todos. Debe devolver el tiempo empleado en la búsqueda en milisegundos (ver abajo).

```
int mide_tiempos_lineal(int tamanio_tabla);
```

Nota: Solo se debe contar el tiempo de búsqueda y no el de creación del array

Ejercicio medir tiempos búsqueda binaria

Crear una función que mida el tiempo que tarda en buscar todos los valores de una tabla de tamaño 'tamanio_tabla' usando búsqueda lineal. Para ello primero se debe crear una array de números enteros ordenados y luego buscar todos. Debe devolver el tiempo empleado en la búsqueda en milisegundos (ver abajo para saber cómo medir tiempos).

```
int mide_tiempos_binaria(int tamanio_tabla);
```

Nota: Solo se debe contar el tiempo de búsqueda y no el de creación del array

Ejercicio medir tiempos búsqueda con hash

Crear una función que mida el tiempo que tarda en buscar todos los valores de una tabla de tamaño 'tamanio_tabla' usando búsqueda lineal. Para ello primero se debe crear una array de números enteros y luego buscar todos. Debe devolver el tiempo empleado en la búsqueda en milisegundos (ver abajo).

```
int mide_tiempos_hash(int tamanio_tabla);
```

Nota: Solo se debe contar el tiempo de búsqueda y no el de creación del array

La ejecución de cada consulta mostrará el tiempo empleado en recuperar los datos (sin incluir el tiempo requerido para imprimir por pantalla). A continuación se muestra un ejemplo de cómo medir el tiempo de ejecución de una operación:

```
#include <stdio.h>
#include <sys/time.h>

int main(void)
{
    struct timeval ti, tf;
    double tiempo;

    gettimeofday(&ti, NULL);   // Instante inicial

    printf("Lee este mensaje y pulsa ENTER\n");
    getchar();

    gettimeofday(&tf, NULL);   // Instante final

    tiempo = (tf.tv_sec - ti.tv_sec)*1000 +
             (tf.tv_usec – ti.tv_usec)/1000.0;

    printf("Has tardado: %g milisegundos\n", tiempo);
}
```

Ejercicio ordenar por selección

Crear una función que ordene el array pasado como parámetro de tamaño 'tam' por el método de selección.

```
void ordenar_por_seleccion(int *tabla, int tamanio_tabla);
```

Ejercicio ordenar por inserción

Crear una función que ordene el array pasado como parámetro de tamaño 'tam' por el método de inserción.

```
void ordenar_por_insercion(int *tabla, int tamanio_tabla);
```

Ejercicio ordenar por burbuja

Crear una función que ordene el array pasado como parámetro de tamaño 'tam' por el método de la burbuja.

```
void ordenar_por_burbuja(int *tabla, int tamanio_tabla);
```