Córdoba 29 de Abril del 2021

Primer Parcial de Laboratorio

Algoritmos y Estructura de Datos II

TEMA B

Ejercicio 3

Ahora se trabajará sobre una estructura **`movie_t`** definida como

```
typedef struct s_movie_t {
  char name[MAX_NAME_LENGTH + 1u];          // Nombre de la peli
  char director[MAX_DIRECTOR_LENGTH + 1u];  // Nombre del director
  unsigned int runtime;                     // Duración en minutos
  float avg_rating;                         // Rating promedio
  float n_votes;                            // Cantidad de votos
} movie_t;
```

que guarda información sobre películas. Particularmente trabajaremos sobre datos recopilados por el sitio IMDB de películas que se estrenaron este año.

a) Completar en **`sort.c`** la definición de la funcion

```
bool goes_before(movie_t s1, movie_t s2)
```

de tal manera que devuelva `true` si y sólo si la duración en minutos de `s1` es menor o igual a la duración de `s2`.

b) Hacer una implementación de `sorted_until()` que trabaje sobre un arreglo con elementos del tipo `movie_t`, es decir que tenga el siguiente prototipo:

```
unsigned int array_sorted_until(movie_t movielist[], unsigned int size)
```

Debe basarse en el criterio de orden impuesto por `goes_before()`

c) Modificar **`main.c`** para que se muestre un mensaje indicando si la *movielist* está completamente ordenada y, de lo contrario, que indique hasta qué índice lo están.

Para verificar pueden usar las *movielist* que les incluimos siguiente resultados:

Movielist	***sorted***	***sorted_unitl***
`parcial-animation.mvl`	**`false`**	**`7`**
`zero-action.mvl`	**`false`**	**`0`**
`sorted-scifi.mvl`	**`true`**	**`9`**

Las salidas de la ejecución deben ser:

```
$ ./movielist parcial-animation.mvl
(:)
Movielist is sorted until 7
```

```
$ ./movielist sorted-scifi.mvl
(:)
Movielist is sorted
```