film.h

```
#ifndef FILM_H_INCLUDE
#define FILM_H_INCLUDE

#include <stdint.h>
#include <stdbool.h>

#include "llist.h"

typedef enum rating_t
{
  R_NONE = 0,
  APPROVED,
  G,
  M,
  N_A,
  NOT_RATED,
  PASSED,
  PG,
  PG_13,
  R,
  TV_14,
  UNRATED,
  NUM_RATINGS
} Rating;
typedef enum category_t
{
  C_NONE    = 0x000001,
  ACTION    = 0x000002,
  ADVENTURE = 0x000004,
  ANIMATION = 0x000008,
  BIOGRAPHY = 0x000010,
  COMEDY    = 0x000020,
  CRIME     = 0x000040,
  DRAMA     = 0x000080,
  FAMILY    = 0x000100,
  FANTASY   = 0x000200,
```

```
  FILM_NOIR = 0x000400,
  HISTORY   = 0x000800,
  HORROR    = 0x001000,
  MUSIC     = 0x002000,
  MUSICAL   = 0x004000,
  MYSTERY   = 0x008000,
  ROMANCE   = 0x010000,
  SCI_FI    = 0x020000,
  SHORT     = 0x040000,
  SPORT     = 0x080000,
  THRILLER  = 0x100000,
  WAR       = 0x200000,
  WESTERN   = 0x400000,
  NUM_CATEGORIES = 22
} Category;
typedef uint_least32_t CategoryType;

/**
 * Returns the string representation of a particular rating
 * @param  r Rating to represent
 * @return   String representation
 */
const char* rating_toString(const Rating r);
/**
 * Returns the string representation of a particular category
 * @param  c Category to represent
 * @return   String representation
 */
const char* category_toString(const Category c);
/**
 * Internal Rating representation of a rating as a string
 * @param  str The string to get the representation for
 * @return     The internal representation, R_NONE if one isn't found
 */
Rating rating_fromString(const char* const str);
/**
 * Internal Category representation of a rating as a string
 * @param  str The string to get the representation fore
 * @return     The internal representation, R_NONE if one isn't found
 */
Category category_fromString(const char* const str);
/**
 * Converts a forward slash delimited string of categories into
 * a CategoryType, which is a bitfield of Category
 * @param  str Categories as a forward slash delimited list
 * @return     A bitfield of Category
```

```
 */
CategoryType category_fromStrings(const char* const str);


typedef struct film_t Film;

/**
 * Allocates and constructs a new film type
 * @param  title       The film's title
 * @param  year        The film's year of release
 * @param  rating      The film's age rating (MPAA-like)
 * @param  categories Bitfield of the film's categories
 * @param  runtime     The film's duration in minutes
 * @param  score       The film's ranking (presumably 0.0..10.0)
 * @return             A new film, or NULL if allocation failed
 */
Film* film_new(const char* title, uint16_t year, Rating rating,
               CategoryType categories, uint16_t runtime, double score);
/**
 * Cleans up and deallocates an instance of a film type.
 * The pointer provided is invalid after this called
 * @param film The film to delete
 */
void film_delete(Film* film);
/**
 * Prints details of the film to stdout in a formatted manner
 * @param film The film to print
 */
void film_print(Film* film);

/**
 * Retrieves a film's title, or empty string if film is NULL
 * @param  film
 * @return      Title of the film provided, or empty string
 */
const char* film_getTitle(const Film* const film);
/**
 * Retrieves a film's year of release, or 0 if the film is NULL
 * @param  film
 * @return      Year of release, or 0
 */
uint16_t film_getYear(const Film* const film);
/**
 * Retrieves a film's rating, or R_NONE if the film is NULL
 * @param  film
 * @return      Rating of the film, or R_NONE
```

```
 */
Rating film_getRating(const Film* const film);
/**
 * Retrieves a film's runtime, or 0 if the film is NULL
 * @param  film
 * @return      Runtime of the film, or 0
 */
uint16_t film_getRuntime(const Film* const film);
/**
 * Retrieves a film's score, or 0.0 if the film is NULL
 * @param  film
 * @return      Score of the film, or 0.0
 */
double film_getScore(const Film* const film);

/**
 * Tests if a film has a particularly category
 * @param  film The film to test
 * @param  cat  The category to check for
 * @return      True if the film is of this category
 */
bool film_hasCategory(const Film* const film, const Category cat);

#endif
```