film.c

```
#include "film.h"

#include <stdio.h>
#include <stdint.h>
#include <string.h>

#include "alloc.h"

const char* const rating_n[] = {
  "NONE",
  "APPROVED",
  "G",
  "M",
  "N/A",
  "NOT RATED",
  "PASSED",
  "PG",
  "PG-13",
  "R",
  "TV-14",
  "UNRATED",
  "X"
};

const char* rating_toString(const Rating r)
{
  return rating_n[r];
}
Rating rating_fromString(const char* const str)
{
  for (size_t i = APPROVED; i < NUM_RATINGS; ++i)
  {
    if (strcmp(str, rating_n[i]) == 0)
      return (Rating)i;
  }
  return R_NONE;
```

```
}

const char* const category_n[] = {
  "None",
  "Action",
  "Adventure",
  "Animation",
  "Biography",
  "Comedy",
  "Crime",
  "Drama",
  "Family",
  "Fantasy",
  "Film-Noir",
  "History",
  "Horror",
  "Music",
  "Musical",
  "Mystery",
  "Romance",
  "Sci-Fi",
  "Short",
  "Sport",
  "Thriller",
  "War",
  "Western"
};

const char* category_toString(const Category c)
{
  for (size_t i = 0; i < NUM_CATEGORIES; ++i)
  {
    if ((Category)(1 << i) == c)
      return category_n[i];
  }
  return category_n[0];
}
Category category_fromString(const char* const str)
{
  for (size_t i = 0; i < NUM_CATEGORIES; ++i)
  {
    if (strcmp(str, category_n[i]) == 0)
      return (Category)(1 << i);
  }
  return C_NONE;
}
```

```
CategoryType category_fromStrings(const char* const str)
{
  CategoryType ret = 0x0;
  size_t p = 0;
  // <= to not trim out the last one
  // could use strtok, but that requires either a full copy or mutating the
  // original data
  for (size_t i = p; i <= strlen(str); ++i)
  {
    if (str[i] == '/' || i == strlen(str))
    {
      char buf[64];
      memcpy(buf, str+p, i-p);
      buf[i-p] = '\0';
      p = i+1;
      ret |= category_fromString(buf);
    }
  }
  return ret;
}

typedef struct film_t
{
  char* title;
  uint16_t year; // assuming 32k years is enough?
  Rating rating;
  CategoryType categories;
  uint16_t runtime; // in minutes
  double score; // how good (or bad) is this film
} Film;

Film* film_new(const char* title, uint16_t year, Rating rating,
               CategoryType categories, uint16_t runtime, double score)
{
  Film* film = (Film*)mt_malloc(sizeof(Film));

  if (film)
  {
    film->title = mt_malloc(strlen(title)+1);
    strcpy(film->title, title);
    film->year = year;
    film->rating = rating;
    film->categories = categories;
    film->runtime = runtime;
    film->score = score;
  }
```

```
  return film;
}

void film_delete(Film* film)
{
  if (!film) // No film? No work
    return;
  mt_free(film->title);
  mt_free(film);
}

void film_print(Film* film)
{
  if (!film)
    return;

  printf("%s\n\tYear: %u\n\tRating: %s\n\tCategories:",
        film->title, film->year, rating_toString(film->rating));

  for (size_t i = 0; i < NUM_CATEGORIES; ++i)
  {
    if ((1 << i) & film->categories)
      printf("\n\t %s", category_toString(1 << i));
  }

  printf("\n\tRun time: %u minutes\n\tScore: %.1f\n",
        film->runtime, film->score);
}

const char* film_getTitle(const Film* const film)
{
  return film ? film->title : "";
}
uint16_t film_getYear(const Film* const film)
{
  return film ? film->year : 0;
}
Rating film_getRating(const Film* const film)
{
  return film ? film->rating : R_NONE;
}
uint16_t film_getRuntime(const Film* const film)
{
  return film ? film->runtime : 0;
}
```

```
double film_getScore(const Film* const film)
{
  return film ? film->score : 0.0;
}

bool film_hasCategory(const Film* const film, const Category cat)
{
  if (!film)
    return false;

  return cat & film->categories;
}
```