Tutorial 7

NWEN 241
Systems Programming

Alvin C. Valera

alvin.valera@ecs.vuw.ac.nz

Content

- File I/O

- Command line arguments

Tutorial Problem

You are given data (in CSV format) about the greatest sci-fi and fantasy films of all time (taken from https://docs.google.com/spreadsheets/d/1okTb4MTllkDWlj0daHNfA8RwOPhLQkWtpQN_znJJJmk/edit?hl=en&hl=en#gid=5):

- Define a structure that can hold a record
- Declare data structure to hold records
- Read CSV file and load data to data structure
- Display contents of data structure
- Display content based on command-line argument

Similar to Tutorial 6 Problem

- Key difference is that data is not entered by hand but through a CSV file

CSV file contents (scifi.csv)

```
2001,Stanley Kubrick,1968,1,USA
Metropolis,Fritz Lang,1927,0,Germany
Blade Runner,Ridley Scott,1982,0,USA
Alien,Ridley Scott,1979,1,USA
The Wizard of Oz,Victor Fleming,1939,2,USA
ET,Steven Spielberg,1982,4,USA
Solaris,Andrei Tarkovsky,1972,0,USA
Spirited Away,Hayao Miyazaki,2001,1,Japan
Star Wars (1977),George Lucas,1977,6,USA
Close Encounters,Steven Spielberg,1977,1,USA
King Kong,Ernest B Schoedsack; Merian C Cooper,1933,0,USA
Terminator/Terminator 2,James Cameron,1984,4,USA
The Matrix,Andy & Larry Wachowski,1999,4,USA
Alphaville,Jean Luc-Godard,1965,0,France
Back to the Future,Robert Zemeckis,1985,1,USA
Planet of the Apes,Franklin J Schaffner ,1968,1,USA
Brazil,Terry Gilliam,1985,0,UK
The Lord of the Rings trilogy,Peter Jackson,2001,17,New Zealand
Dark Star,John Carpenter,1974,0,USA
Day the Earth Stood Still,Robert Wise,1951,0,USA
Edward Scissorhands,Tim Burton,1990,0,USA
Akira,Katsuhiro Otomo,1988,0,Japan
Princess Bride,Rob reiner,1987,0,USA
Pan's Labyrinth,Guillermo del Toro,2006,3,Spain
Starship Troopers,Paul Verhoeven,1997,0,USA
```


```
The Wizard of Oz,Victor Fleming,1939,2,USA
ET,Steven Spielberg,1982,4,USA
Solaris,Andrei Tarkovsky,1972,0,USA
Spirited Away,Hayao Miyazaki,2001,1,Japan
```

Define a structure that can hold a record

```
#define DEFAULT_STRLEN      100

struct movie {
      char title[DEFAULT_STRLEN];
      char director[DEFAULT_STRLEN];
      short year;
      short oscars_won;
      char origin_country[DEFAULT_STRLEN];
};

typedef struct movie movie_t;
```

Declare data structure to hold records

Use dynamically allocated memory to hold records

```
#define NUM_RECORDS     25

// Declare a pointer to movie_t
movie_t *movies;

// Allocate memory for 25 records (CSV contents)
movies = (movie_t *)calloc(NUM_RECORDS,
         sizeof(movie_t));
```

Read CSV file and load data to data structure (1/2)

C and C++

```
// Input file stream to read csv
FILE *csv;

csv = fopen("scifi.csv", "r");
if(!csv) {
    printf("Failed to open scifi.csv\n");
    return 0;
}
```

Read CSV file and load data to data structure (2/2)

C and C++

```
int i = 0;

while(!feof(csv) && i < NUM_RECORDS) {
      fscanf(csv, "%[^,],%[^,],%d,%d,%[^\n]%*c",
      movies[i].title, movies[i].director, movies[i].year,
      &movies[i].oscars_won, movies[i].origin_country);
      i++;
}

fclose(csv);
```

Display contents of data structure

```
// Display records
i = 0;
while(i < NUM_RECORDS) {
       printf("Row %d:\n", i);
       printf("Title             : %s\n", movies[i].title);
       printf("Director          : %s\n", movies[i].director);
       printf("Year of Release   : %d\n", movies[i].year);
       printf("Oscars Won        : %d\n", movies[i].oscars_won);
       printf("Country of Origin: %s\n\n", movies[i].origin_country);
      i++;
}
```

Putting it all together

- A C/C++ program to load CSV file and display contents of data structure

- See `t7a.c`

Display content based on command-line argument

- Program design
  - When no command-line argument is specified, display all rows
  - When there is command-line argument, interpret the argument as the row number to be displayed
  - Ignore command-line arguments other than the first

- See `t7a2.c`

C++

Declare data structure to hold records

Use `vector` from C++ standard template library to hold records

```
#include <vector>

// Declare a vector of movie_t for holding
// the records
vector<movie_t> movies;
```

Read CSV file and load data to data structure (1/3)

C++

```
// Input file stream to read csv
ifstream csv;

csv.open("scifi.csv");
if(!csv.is_open()) {
    printf("Failed to open scifi.csv\n");
    return 0;
}

// Variable to hold a record
movie_t m;
```

Read CSV file and load data to data structure (2/3)

C++

```
// Parse CSV file and load records
while(!csv.eof()) {
    string t;
    if(!getline(csv, t, ',')) break; // read line until ','
    strncpy(m.title, t.c_str(), t.length()+1);
    if(!getline(csv, t, ',')) break; // read line until ','
    strncpy(m.director, t.c_str(), t.length()+1);
    if(!getline(csv, t, ',')) break; // read line until ','
    m.year = atoi(t.c_str());
    if(!getline(csv, t, ',')) break; // read line until ','
    m.oscars_won = atoi(t.c_str());
    if(!getline(csv, t)) break; // read line until newline
    strncpy(m.origin_country, t.c_str(), t.length()+1);
    movies.push_back(m);
}
```

Read CSV file and load data to data structure (3/3)

C++

```
// Close
csv.close();
```

C++

Display contents of data structure

```
// Display records
int i = 0;
while(i < movies.size()) {
    cout << "Row " << i << endl;
    cout << "Title            : " << movies[i].title << endl;
    cout << "Director         : " << movies[i].director << endl;
    cout << "Year of Release  : " << movies[i].year << endl;
    cout << "Oscars Won       : " << movies[i].oscars_won << endl;
    cout << "Country of Origin: " << movies[i].origin_country << endl;
    i++;
}
```

Putting it all together

- A C++ program to load CSV file and display contents of data structure

- See `t7b.cc`