Assignment 3 –Structures in C

1. Allowed libraries: stdio.h, string.h
2. We will be creating a structure for storing DVDs in a library.
3. Create a Header file, which is a text file, named ***structures.h***. Do the following in ***structures.h***.
    a. Create two constants:
        i. ***TEXT_SIZE*** and set it to 80.
        ii. ***NUM_DVDS*** set it to 4.
    b. Create a typedef structure named ***dvd_t*** which has a field for the title which is ***TEXT_SIZE*** in length called ***title***, a field for director which is ***TEXT_SIZE*** in length called ***director***, and an integer for play time of the DVD in minutes named ***playTime***.

    The fields MUST be named exactly as shown above otherwise the tests will fail.
    You MUST create a "typedef struct" as shown in the notes or your tests will fail.

    c. Add prototypes for the methods created below. A prototype is simply the function declaration without the body. You can wait till you finish the functions to add them if you wish.

    Here is an example prototype.

    ```
    int getMask(int  start_bit,  int  end_bit,  int  bit_value);
    ```

4. Create a C program file named ***structures.cc*** which contains the functions, with bodies, listed below. Put the assignment and your name as comments as the first two lines like this.
    //Joel Swanson
    //Programming Assignment x – Structures

    DO NOT PUT #DEFINES IN structures.cc.
    (You can use your own #defines in structures.cc, but not the #defines listed above.)
    DO NOT PUT PROTOTYPES IN structures.cc.
    DO NOT PUT ANY TYPEDEFS IN structures.cc.
    ALL OF THE ABOVE ITEMS SHOULD BE IN structures.h.
    ONLY CREATE THE FUNCTIONS LISTED. NO HELPER FUNCTIONS SHOULD BE REQUIRED.

    a. Include the header file include below your other included files.

    ```
    #include <stdio.h>
    #include <string.h>
    #include "structures.h"
    ```

Note the quotes instead of angle brackets. This tells the compiler to look in the current project directory instead of the default library directory to find structures.h.

b. Create a function named ***getOneDVD*** which takes a **reference** to a ***dvd_t*** as a parameter. This function should get a single title, director, and play time from the terminal using scanf and put that data into the ***dvd_t*** structure. Use **EXACTLY** the prompts as shown in the sample output. Notice there is ONE single space after the colon.

   NOTE ON SCANF AND SPACES. You have to tell scanf to allow spaces otherwise it will stop scanning when it hits a space. The following should work:

```
char test[11];
scanf("%10[^\n]s", test);
char c;
scanf("%c",  &c);  //Of course you have to get rid of the newline now.
```

   The above shows reading a string which is at the most 10 characters in length. Yours will be a different length.

   The [^\n] means not a newline. It will accept characters up to, but not including a newline. This will allow you to have spaces in the input string.

   The scanf stops when the newline is reached (i.e. the enter key is pressed) so you must remove the newline from the scanf queue which is what the second scanf above demonstrates.

   When reading only primitive types, this isn't a problem since the scanf skips whitespace and simply starts where the text starts. **But** when reading a string AFTER reading an a primitive type, that newline is still there.

   If you find scanf skipping inputs, it is because of this issue and you must read the newline character after the previous scanf but before the one being skipped.

c. Create a function named ***printOneDVD*** which takes which takes a dvd_t (by value, not by reference) as a parameter. This function should print the title, director and play time from the structure in the following format Title:Director:PlayTime

   Assuming Eraserhead, David Lynch, and 85 were the title, director, and play time in the structure printOneDVD should print the following with a single newline at the end. Note that the separating colons have no spaces before or after.

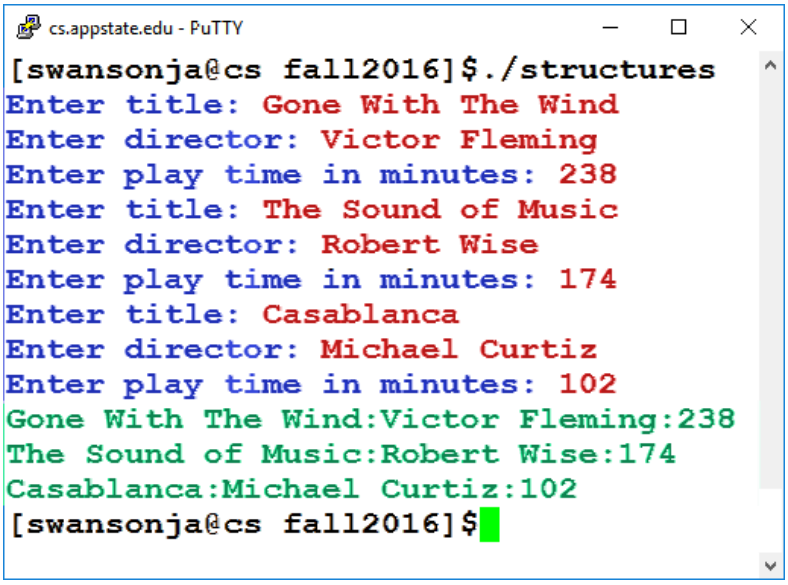
Eraserhead:David Lynch:85

d. Create a function named ***getCollection*** which takes an array of ***dvd_t*** structures and a number of dvds to get. Call ***getOneDVD*** for each ***dvd_t*** in the array

e. Create a function named ***printCollection*** which takes an array of ***dvd_t*** structures and a number of dvds to print. This method should call ***printOneDVD*** for that many ***dvd_t*** objects from the array.

5. Copy ***main.cc*** from the AsULearn page and put it in the same folder as ***structures.cc*** and ***structures.h***.

   Use main to test your code. Test thoroughly before submitting. Below is a running of main.cc. The blue text is your prompt output. The red text is data entered from the keyboard. The green text is the output after all of the data has been entered.



6. Compile and make sure your output matches the output shown above for the given input (of course yours will be black text and not red, blue, and green).

   Compile command (Spaces have been exaggerated.) Note that you don't have to specify the .h file. It will be assumed from the #include.
   g++ -Werror -Wall -o structures main.cc structures.cc

7. Submit your ***structures.cc*** on Web-CAT.

Do not submit a file with a main function.
DO NOT submit structures.h.
DO NOT submit main.cc.
File cannot be submitted more than 2 days late.
Late submissions will suffer a 10 point penalty per day.
Each submission over 5 will be -1 point.