CECS 130
Assignment 4
Assignment due on or before 9:00 pm Monday, July 3, 2017

1. Do challenge 1, and 2 of Chapter 9 on page 214.
2. Do challenge 2 of Chapter 10 on page 232.
3. Do challenge 1, and 2 of Chapter 11 on page 253.
4. Do challenge 1 on page 176 from the C++ Textbook.
   a. Write a c++ program that returns the length of s to support your answer.
   b. Define a string instance called s1, copy the string "Hello World" in s1 the use the string length function to determine the length of s1 an display it.
5. Do challenge 4 on page 176 from the C++ Textbook

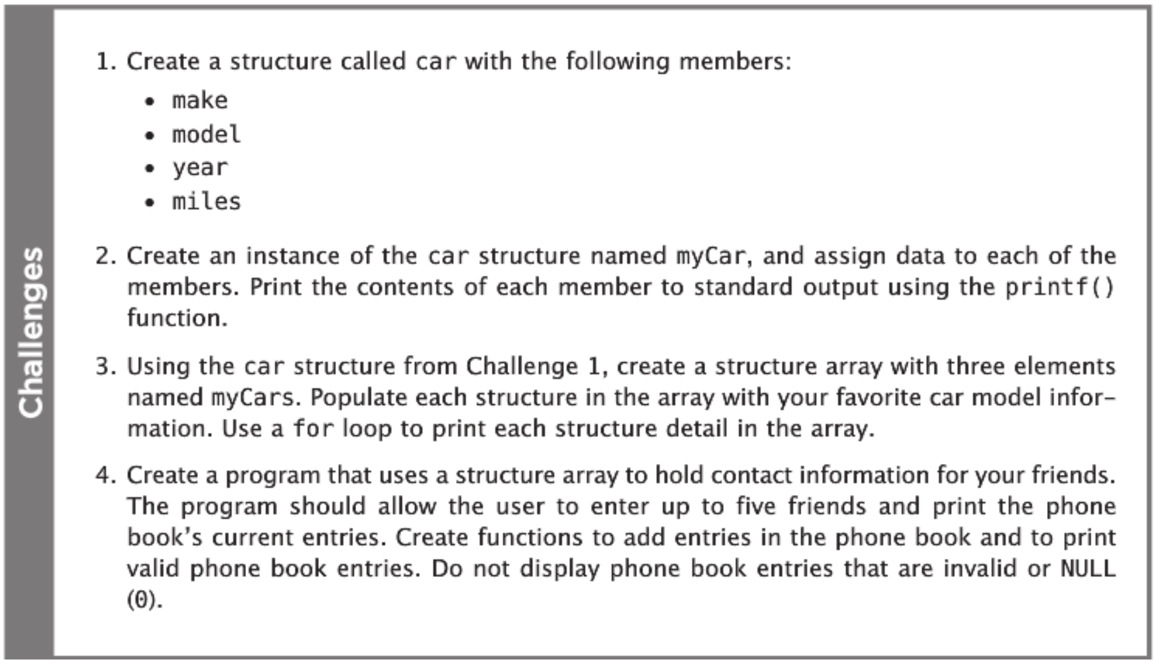
A note about assignments and reports:

Your report must conform to the following rules:

1. All reports have to be submitted as a **PDF** report that contains:
   1.1. Title page with your name, assignment number and the day you are actually submitting this report (Not the assignment due date)
   1.2. A comprehensive set of snapshots showing the inputs submitted and outputs obtained in the case of a successful output or a failure.
2. A text file that contains all source code, please concatenate all source code in one text file.
3. Make sure that you include as a comment at the top of your file your name and section:

   As an example:
   ```
   /**************************/
   /* John Q. Public        */
   /* CECS 130-11          */
   /* Assignment 35         */
   /************************/
   ```
   Failure to do this will cost you points.

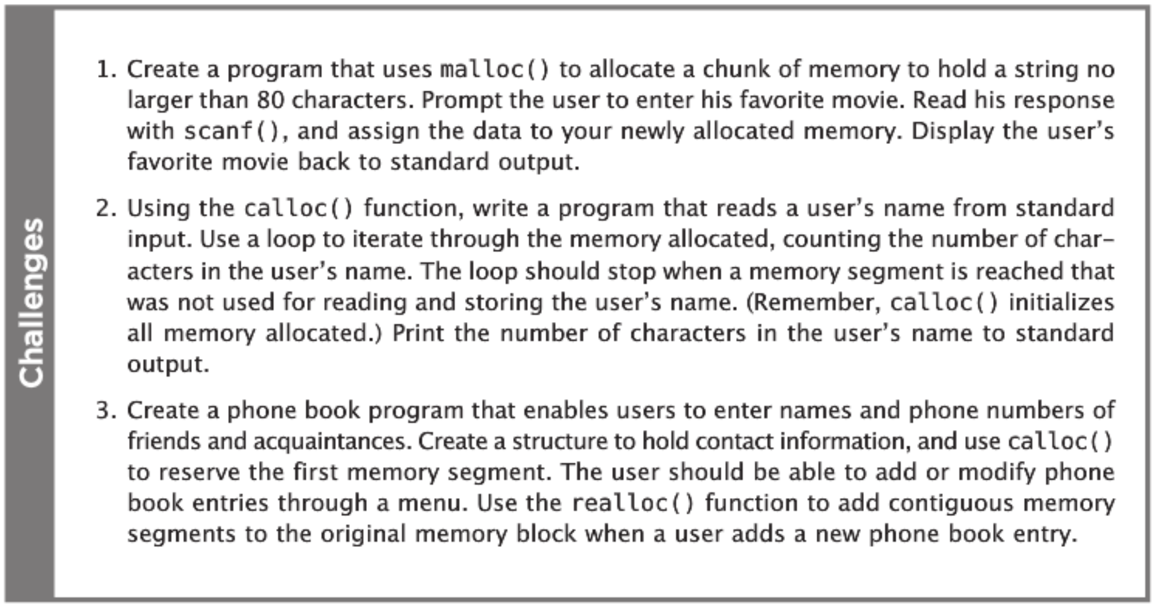
4. Please zip both the PDF document with the source code and submit one zip file.
5. Please do not submit your eclipse or bloodshed project or any IDE project that you may be using. I will be compiling and testing your source code from the text file in part 2 above to test running your applications and to verify that they run.
6. Remember that you must only access BlackBoard using section 130-01



Challenges

1. Create a structure called car with the following members:
   - make
   - model
   - year
   - miles
2. Create an instance of the car structure named myCar, and assign data to each of the members. Print the contents of each member to standard output using the printf() function.
3. Using the car structure from Challenge 1, create a structure array with three elements named myCars. Populate each structure in the array with your favorite car model information. Use a for loop to print each structure detail in the array.
4. Create a program that uses a structure array to hold contact information for your friends. The program should allow the user to enter up to five friends and print the phone book's current entries. Create functions to add entries in the phone book and to print valid phone book entries. Do not display phone book entries that are invalid or NULL (0).

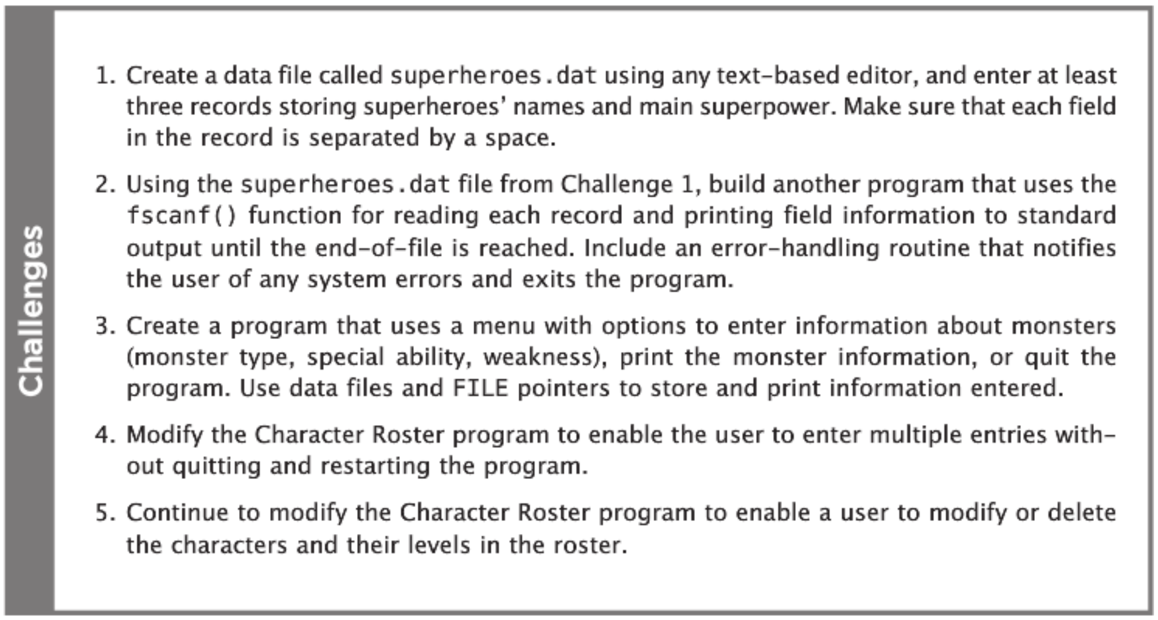
Chapter 9 page 214



Challenges

1. Create a program that uses malloc() to allocate a chunk of memory to hold a string no larger than 80 characters. Prompt the user to enter his favorite movie. Read his response with scanf(), and assign the data to your newly allocated memory. Display the user's favorite movie back to standard output.
2. Using the calloc() function, write a program that reads a user's name from standard input. Use a loop to iterate through the memory allocated, counting the number of characters in the user's name. The loop should stop when a memory segment is reached that was not used for reading and storing the user's name. (Remember, calloc() initializes all memory allocated.) Print the number of characters in the user's name to standard output.
3. Create a phone book program that enables users to enter names and phone numbers of friends and acquaintances. Create a structure to hold contact information, and use calloc() to reserve the first memory segment. The user should be able to add or modify phone book entries through a menu. Use the realloc() function to add contiguous memory segments to the original memory block when a user adds a new phone book entry.

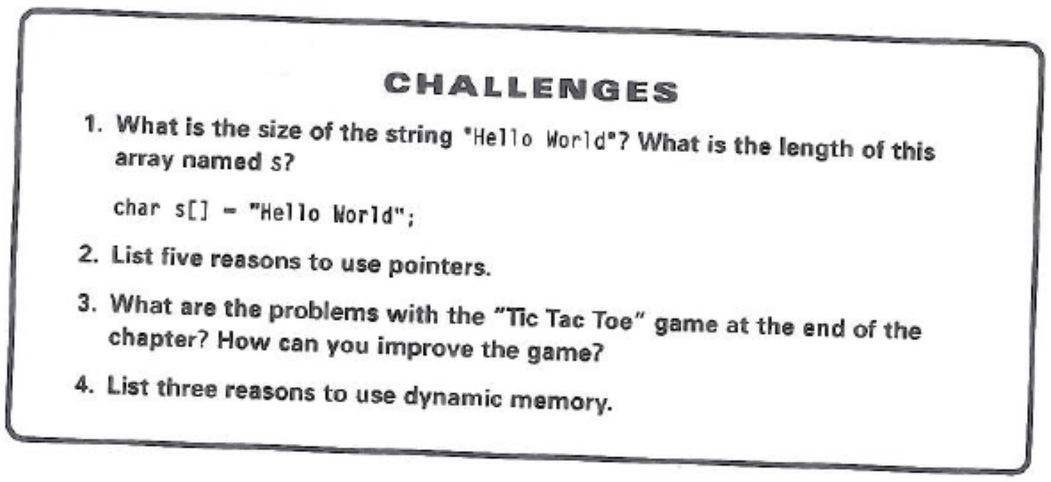
Chapter 10 page 232



Challenges

1. Create a data file called `superheroes.dat` using any text-based editor, and enter at least three records storing superheroes' names and main superpower. Make sure that each field in the record is separated by a space.
2. Using the `superheroes.dat` file from Challenge 1, build another program that uses the `fscanf()` function for reading each record and printing field information to standard output until the end-of-file is reached. Include an error-handling routine that notifies the user of any system errors and exits the program.
3. Create a program that uses a menu with options to enter information about monsters (monster type, special ability, weakness), print the monster information, or quit the program. Use data files and `FILE` pointers to store and print information entered.
4. Modify the Character Roster program to enable the user to enter multiple entries without quitting and restarting the program.
5. Continue to modify the Character Roster program to enable a user to modify or delete the characters and their levels in the roster.

Chapter 11 page 253



CHALLENGES

1. **What is the size of the string `"Hello World"`? What is the length of this array named `s`?**

```
char s[] = "Hello World";
```

2. **List five reasons to use pointers.**
3. **What are the problems with the "Tic Tac Toe" game at the end of the chapter? How can you improve the game?**
4. **List three reasons to use dynamic memory.**

Page 176 from the C++ book