This lab is designed to illustrate the following concepts:

* Use a struct as the basic element of information. At different places in the program, an individual struct may be passed (by reference), or an entire vector of structs may be passed (also by reference).
* Use a vector of structs as a database of information during the time the program is running. The database is read from a CSV file upon startup and is saved to the same CSV file upon shutdown. NOTE this means that the original inventory file will be *overwritten* each time the program runs. This means that if in the course of development you corrupt the inventory file, you will have to re-copy or re-create an uncorrupted inventory file.
* All input, both from the CSV file and from the user, is via C++strings. No C-strings are used. All input is validated. Thus the program will not crash with erroneous input, but rather the user is simply re-prompted. This requires string input to be converted into numeric values as appropriate.

This program helps manage manufacturing inventory. Each part in the manufacturing system has a description, a current inventory quantity, and a maximum and minimum number of that part to keep on hand.

When the user chooses a part and uses a certain number of that part, that number is deducted from inventory. If this causes the current inventory amount to fall below the minimum, an order is placed to re-stock the part up to the maximum amount to kept on hand.

When the program starts, the list of items, current inventory, maxima, and minima are read from a disk file. The program then repeatedly prompts the user for the part to be used, managing the inventory (deleting and re-stocking) along the way.

[Here](http://borax.truman.edu/180/lab14struct/framework.cpp) is the framework of your C++ program. Do not change any of the existing code in any way, except for the places that say “replace this”. Your assignment is to write the missing code to make to make the program work correctly. Here is a run showing one interaction that does not require a restock, followed by an interaction that does require a restock (notice the lines in red):

Item Description Current Min Max

--------------------------------------------------

0 Thermistor 50 10 50

1 Thyratron 2 1 10

2 Inclinometer 4 2 4

3 Rectifier 40 8 40

4 Darlington NPN 15 4 15

5 Yagi antenna 1 1 1

6 MOSFET 13 10 100

7 Nixie tube 5 1 5

8 DIAC 10 5 10

9 Klystron 3 5 10

10 Stepping motor 2 1 2

11 Piezoelectric crystal 4 15 50

Choose item from inventory, q to quit (0 to 11): 2

How many Inclinometers are you using? (1 to 4): 2

Item Description Current Min Max

--------------------------------------------------

0 Thermistor 50 10 50

1 Thyratron 2 1 10

2 Inclinometer 2 2 4

3 Rectifier 40 8 40

4 Darlington NPN 15 4 15

5 Yagi antenna 1 1 1

6 MOSFET 13 10 100

7 Nixie tube 5 1 5

8 DIAC 10 5 10

9 Klystron 3 5 10

10 Stepping motor 2 1 2

11 Piezoelectric crystal 4 15 50

Choose item from inventory, q to quit (0 to 11): 2

How many Inclinometers are you using? (1 to 2): 1

Restocking Inclinometer adding 3

Item Description Current Min Max

--------------------------------------------------

0 Thermistor 50 10 50

1 Thyratron 2 1 10

2 Inclinometer 4 2 4

3 Rectifier 40 8 40

4 Darlington NPN 15 4 15

5 Yagi antenna 1 1 1

6 MOSFET 13 10 100

7 Nixie tube 5 1 5

8 DIAC 10 5 10

9 Klystron 3 5 10

10 Stepping motor 2 1 2

11 Piezoelectric crystal 4 15 50

The program uses a CSV text file for persistent storage between runs. [Here](http://borax.truman.edu/180/lab14struct/inventory.txt) is a text file you can use.

The hardest part of this lab is understanding how the program works and what parts are missing that you have to fill in.