Purpose

The purpose of this assignment is to have you work on an array of structures and File I/O. The input data must be read programmatically using File I/O. User has to be prompted to enter the input data file name.

Scenario

Microbiologists estimating the number of bacteria in a sample that contains bacteria that do not grow well on solid media may use a statistical technique called the most probable number (MPN) method. Each of five tubes of nutrient medium receives 10 ml of the sample. A second set of five tubes receives 1 ml of sample per tube, and in each of a third set of five tubes, only 0.1 ml of sample is placed. Each tube in which bacterial growth is observed is recorded as a positive, and the numbers for the three groups are combined to create a triplet such as 5-2-1, which means that all five tubes receiving 10 ml of sample show bacterial growth, only two tubes in the 1-ml group show growth, and only one of the 0.1-ml group is positive. A microbiologist would use this combination –of-positive triplet as an index to a table like the one below to determine that the most probable number of bacteria per 100 ml of the sample is 70, and 95 percent of the samples yielding this triplet contain between 30 and 210 bacteria per 100 ml.

Table of Bacterial Concentrations for Most Probable Number Method

Combination of Positives	MPN Index/100 ml	95 percent Confidence Limits	
		Lower	Upper
4-2-0	22	9	56
4-2-1	26	12	65
4-3-0	27	12	67
4-3-1	33	15	77
4-4-0	34	16	80
5-0-0	23	9	86
5-0-1	30	10	110
5-0-2	40	20	140
5-1-0	30	10	120
5-1-1	50	20	150
5-1-2	60	30	180
5-2-0	50	20	170
5-2-1	70	30	210
5-2-2	90	40	250
5-3-0	80	30	250
5-3-1	110	40	300
5-3-2	140	60	360

Problem

Define a structure type to represent one row of the MPN table. The structure will include one string component for the combination-of-positives triplet and three integer components in which to store the associated most probable number and the lower and upper bounds of the 95 percent confidence range.

Write a program to implement the following algorithm for generating explanations of combination-of-positives triplets.

1) Load the MPN table from a file into an array of structures called mpn_table
2) Print the entire MPN table using the data that from the array of structures that you filled in from the step above
3) Repeatedly get from the user a combination-of-positives triplet, search for it in the combination-of-positives components of mpn_table, and then generate a message such as:

```
2-1-0, MPN = 20, 95% of samples contain between 9 and 45 bacteria per ml
1-0-2, MPN = 17, 95% of samples contain between 5 and 35 bacteria per ml
Combination of Positives not found for 1-3-5
Combination of Positives not found for 1-10
Combination of Positives not found for 3-3-0
5-1-0, MPN = 60, 95% of samples contain between 30 and 180 bacteria per ml
```

Each of these operations must be implemented in separate functions. The main function should call corresponding functions for the first 2 operations, and then loop repeatedly to perform the 3rd operation until the user input is "000". See the Requirements section below for more specifics on these functions.

Input

The input MPN data is coming from a file whose name is entered by the user, but the search string/data comes from the user. You should read the data into your array. Your array should be capable of holding a maximum of 50 records. After reading and printing the records, you need to repeatedly request a search string from the user which is the combination-of-positives triplet until the user enters 000 to stop the program..

Output

All output is directed to the screen or standard output.

First print out information on the reading of the data. As the user repeatedly searches for records, the matching record details (or a message indicating record was not found) must be printed. Your output does not have to look exactly like mine but it should be as complete and easy to read.

Requirements

- You must use at least the following functions (all of them must be commented):
    - Function load_mpn_table() takes as parameters the name of the input file and the mpn_table array. Function opens the file, fills the mpn_table array, and closes the file. Then it returns the actual array size as the function result. If the file contains too much data, the function should store as much data as will fit into the array maximum size.
    - Function print_mpn_table() takes as parameters the mpn_table array and its actual size. Function prints each record in the array one at a time and returns. Function returns no value. Printing is done to standard output
    - Function search_mpn_table() takes as parameters the mpn_table array, its actual size, and a target string representing a combination-of-positives triplet. Returns the subscript/index of the structure whose combination-of-positives component matches the target or -1 if not found. The main function accepts the triplet combination string from the user and then calls search_mpn_table() which should return the proper subscript/index. The main function should print the details of this search using the index returned (or a message indicating that no match was found). All printing is done to the standard output or display (not to a file). No printing must be done in the search function.

- The program must be split into multiple files:
    - The main must be the only function in one code file (.c)
    - All the other functions should be in a second code file (.c)
    - The other functions should be forward declared (prototype declarations) in a header file (.h) and included in both code files
    - **Be sure to submit all three files!**

- All function definitions must have a block comment describing what they do.

- Work on the program using the modularity of the functions, for example write a program that reads the data and prints it out. Once you have these working you can move on to the other functions

- As always test your program carefully and follow good programming style.

Other Details

- You must follow all the coding style rules as specified in our "coding guidelines". In particular:
    - You must put your name enclosed in a comment box at the top, and keep any other comments that are already there.
    - Keep lines to the point of making your code easily readable. You must use good names for any variables you create (a full word that describes what it is there for).
- Details that you do not follow are penalized after other scored items are added up, so even if you got a 100 for the functionality of your program, you can still get a lower score because you did not follow all the other requirements for the assignment.

Submission

Submit this assignment with the code 10L followed by the names of all 3 files:
For example:

```
submit 10L 10L.c 10Lfunctions.c 10Lheader.h
```