C Programming Functions, File Access and Simple Array Use

For this assignment, you will implement a short C program that will perform simple searches in a file of integer data.

You will read input data from a file named `sData.txt`; here is a sample input file:

```
87  23  7  91
-1
43  73  14  91  32
18   7  90  89  33
23  30  88  17  42
```

The input file begins with a list of no more than 10 non-negative integer values which are to be searched for; call these the *sought values*. The end of this section of the file will be indicated by a *sentinel*, a single negative integer value. You must use an array to store the sought values; this is feasible since you are given an upper limit on the number of them.

Following that sentinel, there will be a list of non-negative integer values within which you will search for occurrences of the sought values; we will call this the *search space*. There is no limit on the number of values that may be given in this part of the file, so you cannot use an array to store them.

Note also that line breaks may occur at any point within the input file, so your logic for processing the file must not depend upon them.

You will write your results to an output file named `fSearch.txt`; the formatting of the file is mandatory. First, report the number of sought values, as shown in the sample output below, including the blank line. Following two header lines, as shown below, you will echo each value in the search space that matches one of the sought values, along with the position of that matching value within the search space. Positions start at 0. Once all the searching has been completed, write out a label, as shown below, and then the number of values you found in the search space.

```
Searching file for matches to 4 values.

Sought  Found at
---------------
    91         3
     7         6
    23        10
Number of items searched through in list:
15
```

You must not implement your solution as a single `main()` function; rather, you should analyze the requirements and identify a suitable collection of functions to compose into a working solution.

You may implement your solution in a single C source file, or if you are feeling adventurous, decompose it into two or more separate C files. In the latter case, you must also write suitable header files.

You may compile your program by using the following command in a Linux shell:

```
gcc –o <name for executable> –std=c99 -Wall *.c
```

`gcc` handles the wildcard and will correctly sort out dependencies; note that you do not include header files in the command.

A small amount of test data files will be supplied on the course website. You should make sure that your solution produces correct results with all the posted data before you make any submissions to the Curator.

What to Submit

You will create an uncompressed tar file containing all of your source files, and nothing else, and submit that tar file to the Curator. Note: you must do this even if your implementation uses a single source file.

This assignment will be graded automatically. You will be allowed up to ten submissions for this assignment, so use them wisely. Test your programs thoroughly before submitting them. Make sure that your programs produce correct results for every logically valid test case you can think of. Do not waste submissions on untested code.

The Curator will assign a score based on runtime testing of your submission; your best score will be counted; the TAs will later verify that your best submission meets the stated restrictions, and assess penalties if not.

The *Student Guide* and other pertinent information, such as the link to the proper submit page, can be found at:

http://www.cs.vt.edu/curator/

Pledge:

Each of your program submissions must be pledged to conform to the Honor Code requirements for this course. Specifically, you **must** include the following pledge statement in the submitted file:

```
//    On my honor:
//
//    - I have not discussed the C language code in my program with
//      anyone other than my instructor or the teaching assistants
//      assigned to this course.
//
//    - I have not used C language code obtained from another student,
//      or any other unauthorized source, either modified or unmodified.
//
//    - If any C language code or documentation used in my program
//      was obtained from an allowed source, such as a text book or course
//      notes, that has been clearly noted with a proper citation in
//      the comments of my program.
//
//    <Student Name>
```

Failure to include this pledge in a submission will result in the submission being disallowed during code review.