Database Search

Brief Description

First let us start with a brief definition of the Database concept. A *database* is a collection of data records stored in a computer in a systematic way. In this way, a computer program can be used to answer queries about the data stored in the database by searching the records.
In this homework, you will write a program that will search the books written by an author published in a certain range of years. The database records that you will deal with are stored in two text files (see "Structure of Database (Input) Files" section for details). Your program will basically search and retrieve the books written by an author published in a certain range of years all entered by the user. Resulting book list will be stored in a separate output file (see "Name and Structure of Output File" section for details).

Name and Structure of Database (Input) Files

As mentioned before, database is stored in two different text files. One is only for book records, the other one is for author-book pair records. These two files will be input files to your program (i.e. your program will read some data from both files) and they are generated outside of the program using Notepad (we provide example input files within the zip package of this homework).
Name of both input files will be entered by the user of your program. At the beginning of your program, user will be asked to enter an input file name for the author records. If user enters a name for a file that does not exist, your program should display an appropriate message and ask for the name of a new input file name until the file is successfully opened. After first input file is opened successfully, your program will ask for another input file name for book records file. The same file opening control will be performed for this file too. If user enters a name for a file that does not exist, your program should display an appropriate message and ask for the name of a new input file name until the file is successfully opened.

Books Database File

The books input file contains several information fields for each book. For a single book, there are three columns of information on the same line. Each line corresponds to the *record* of a book. The record structure for each book is described as follows:
Published_year Price Book_title

- Record line starts with the published year of the book.

  - *Published_year* is a single integer used to define the year the book is published. Such as; 2008, 2009, etc.

- It is followed by the price of the book;*Price*.

  - *Price* is a single double value. Such as; 35.50, 50, etc.

- Last data item of the record is the name of the book title;*Book_title*.

- *Book_title* may consist of more than one word. You may assume that the book title has at least one word in it, but there is no upper limit on the number of words. In the context of this homework, a *word* is defined as a sequence of non-space characters. A word may contain letters, digits, or any punctuation marks and symbols. Words that make up the book title may be separated by one or multiple white spaces. You should handle it.

The abovementioned record structure, which contains 3 different data items in one line, is just for one book. The input file contains information about several books. Therefore, there are several such records in the input file. You may assume there is a record in each line of input file, no line is empty.

Authors Database File

The authors database file contains author-book pairs written by that author. For a single *author record*, there are 2 pieces of information on the same line. The record structure for each book-author pair is described as follows:

Author_firstname *Author_lastname* *Book _title*

Author_firstname Author_lastname Book _title

Author_firstname is the first name of the author and contains exactly one word. For example, *Yasar*.

Author_lastname is last name of the author and contains exactly one word. For example, *Kemal*.

Book_title is the name of the book title and contains one or more words. For example, *Ince Memed*.

We assume that no author-book pair appears more than once in authors database file. For example, if there is a pair such as *Author_firstname_1 Author_lastname_1 Book11*, then there will not be another pair such as *Author_firstname_1 Author_lastname_1 Book11*.

There might be any number of white spaces (i.e. blanks) between and after each data item in both files. You should handle it.

There is no specific order of records in both input files. That means you cannot assume that the records of the input files are ordered according to *Book_title, Author* or any other data attribute.

The structure of the input files and the associated rules and assumptions explained above are fixed such that you cannot change them or make any other assumptions. No format check for the contents of the input files is needed. You may assume that all data are in the correct format. You may examine the sample input files (booktitles.txt, authors.txt) provided in the zip package for this homework (hw6.zip).

Database Search/Filter

Database search will basically be searching the books database file for all books of a given author between a given start year and an end year. The first and last names of the author to be searched will be entered by the user of your program using the keyboard. After the author's first and last names, the user is asked to enter a start year and an end year also. This year range will be used to select the books published after the start year and before the end year including the start and end years ($start\_year \leq Published\_year \leq end\_year$). Your program should also validate that the start and end years are nonnegative and also start year is smaller than or equal to the end year. Your program should keep asking for start and end years until the user enters valid numbers. If the start year is equal to the end year, then that would mean books published on that year.

For the author entered via the keyboard, your program will search each book written by that author through the books database file. To do so, first the authors database file must be searched from the beginning to find out all records with *Author* field that matches the author's "*first name"*+"*last name"* entered by the user the program. For each of such records, your program should then search the record with that *Book_title* in the books database. If the *Book_title* is found in the books database, your program will select that book only if the published year of the book is greater than or equal to start year and less than or equal to end year.
Then the information of the selected book will be output to an output file. If no such book is found, nothing will be written to the output file. The structure of the output file will be explained in "Name and Structure of Output File" section.
When search is finished, display a message (to the screen) that informs the user about the end of the process and the name of the output file. The message format is:
"The search results are in the file: *output_file_name*"

Name and Structure of Output File

There will be a single output file. The name of this file will be derived from the search inputs; author first and last name, start year and end year. In order to determine the output file name, simply concatenate author first and last names, start year and end year with "_" between them. Moreover, put ".*txt*" to the end of the output file name. For example, if the author name is entered as *Yasar Kemal*, and the start and end years are *2003* and *2009*, then name of the output file will be *Yasar_Kemal_2003_2009.txt*
As mentioned above, all the search results are stored in a single output file. In this file, you should output one line for each book found. The structure of this line is below.
Published_year Price Book_title
Published_year, Price and *Book_title* entries are the data extracted from the books database file. There is one blank between each of these entries.
There are any number of such lines, one for each book found, in the output file.
There is one more important rule about the output file content. While describing input file structure, it is noted that words that make up the book titles may be separated by one or more white spaces. However, in the output file, the words that make up these names must be separated by only one white space (see the sample output files provided below and in the zip package of this homework for some examples.). If you utilize input string streams while reading the input file, this rule can easily be applied since the blanks are used as delimiters while reading words and multiple or single blanks functionally behave as the same.
Here please remark that the structure and format of the output file lines are extremely important since we are going to test the output of your programs automatically. Even a single character mistake may cause your program to fail our tests.

Some Hints

Maybe you have already realized that the simplest way of processing the input file is reading the data line by line and parsing each line. In order to perform such a processing, the ideal mechanism is to use a combination of `getline` function and input string streams. We have given/will give examples of such a processing in the lecture and in recitations.

Just a reminder: When you are done with an output file, do not forget to close it.

Note that string comparisons are case sensitive. That means *ankara* and *Ankara* are not equal to each other.

And some common problems/questions and solutions/hints:

Question 1: I enter a correct input filename, but my program does not open corresponding files.

Answer 1: If you run your program in the VC++ environment, then the input file must be in the same folder as the .vcproj file of your project. Or, you should give the exact path for the file ("C:\cs201\hw6\file.txt"). Otherwise it may not be found while it is opened.

Question 2: I formed a loop in order to check if the input file is opened successfully or not. However, after a file name is entered wrong, even if I enter a correct name, it does not accept it.

Answer 2: The reason, most probably, is not clearing the input file stream before trying to re-open it. In order to re-open of file stream object after a failed open you have to call the clear member function on this file stream.

Question 3: My program works fine if I enter the correct file names. However, if I enter a wrong file name first and then the correct file name, the output files are either empty or the results are problematic.

Answer 3: The reason, most probably, is redefinition of the file stream object in the loop that you check whether the files are opened or not. Do not redefine the file streams in the loops. As mentioned in the question above, clear them if needed.

Question 4: I have written my program, then tested it. The problem is that it is producing empty output files.

Answer 4: If your code does not have any bugs, there is still a possibility to get empty output files. One is that, you may be searching a book title which does not exist in the Books Database File. If that is not the case, second possibility is, you may be searching with upper/lower case letters (different than the original type). Consider that, searches are case sensitive, which means Ankara and ankara are not equal to each other.

Important Remarks

Your program must be modular and you should avoid code duplication. Thus you have to show your ability to use functions in an appropriate way. This will affect your grade. In general, if your main function or any user-defined function is too long and if you do everything in main or in another user-defined function, you grade may be lowered.

Try to use parametric and non-void functions wherever appropriate. Do NOT use any global variables (variables defined outside the functions) to avoid parameter use.

In this homework (and in the coming ones) you are not allowed to use instructions such as “exit” and “goto”. These cause difficulty to control the flow of your programs. Thus we do not approve using them. You are also not encouraged to use “break” and “continue”. The use of “break” and “continue” prevent you from forming good readable loop conditions and hence prevent you from learning how to form good loops. Think cleverly in order not to use any of these instructions. If you don't know these commands, do not even try to learn them (we explained “break” in class). Please remark that the only inputs are input file names, author first and names, start and end years to be searched. No other input is allowed (such as a name for the introduction or anything else not mentioned in this homework specification). Since your submissions are processed automatically, extra inputs cause problems and delays in the processing and grading. If you do not follow this rule, your grade may be lowered.

Sample Input and Output Files

In the zip file of this homework, you will find two input files (*books.txt* and *authors.txt*) and four output files that correspond to the following sample runs.

Sample Runs

We do not provide a greetings part in the following sample runs, but you are expected to display an informative greeting. Command Line Window (keyboard inputs are written in ***red***, ***bold*** and ***italic***)

Sample Run 1

```
This program bla bla bla....
Please enter a filename for author names and book titles database:authors.txt
Please enter a filename for year, price and book titles
database:booktitles.txt
Please enter name and last name of the author you are looking for: Hop hop
Please enter start and end year you are looking for: 2005 2008
The search results are in the file: hop_hop_2005_2008.txt
Press any key to continue . . .
```

Contents of Output File (*hop_hop_2005-2008.txt*)

YEAR PRICE
BOOKTITLE
2005 15.5 Altintop
2008 14 Bundan Baska
2006 25.3 Oyun Yok

Sample Run 2

```
This program bla bla bla....
Please enter a filename for author names and book titles database:authors.txt
Please enter a filename for year, price and book titles
database:booktitles.txt
Please enter name and last name of the author you are looking for: Can Canan
Please enter start and end year you are looking for: 2002 2009
The search results are in the file: can_canan_2002_2009.txt
Press any key to continue . . .
```

Contents of Output File (*can_canan_2002_2009.txt*)

Sample Run 3

```
This program bla bla bla....
Please enter a filename for author names and book titles database:authors.txt
Please enter a filename for year, price and book titles
database:booktitles.txt
Please enter name and last name of the author you are looking for: Kah Kah
Please enter start and end year you are looking for: 2009 0
ERROR! Invalid start or end year.
Please enter start and end year you are looking for: 2009 2008
ERROR! Invalid start or end year.
Please enter start and end year you are looking for: 2009 2009
The search results are in the file: kah_kah_2009_2009.txt
Press any key to continue . . .
```

Contents of Output File (*kah_kah_2009_2009.txt*)

Sample Run 4

```
This program bla bla bla....
Please enter a filename for author names and book titles database:auto.txt
ERROR! Cannot open input file.
Please enter a filename for author names and book titles database :
authors.txt
Please enter a filename for year, price and book titles database:booklet.txt
ERROR! Cannot open input file.
Please enter a filename for year, price and book titles database:books.txt
ERROR! Cannot open input file.
Please enter a filename for year, price and book titles
database:booktitles.txt
Please enter name and last name of the author you are looking for: Ali Kemal
Please enter start and end year you are looking for: 1 2010
The search results are in the file: ali_kemal_1_2010.txt
Press any key to continue . . .
```

Contents of Output File (*ali_kemal_1_2010.txt*)
YEAR PRICE
BOOKTITLE
1992 5.4 Kop
2007 11.6 Olmek

What and where to submit (PLEASE READ, IMPORTANT)?

You should prepare (or at least test) your program using MS Visual C++ 6.0 (part of MS Visual Studio 6.0). We will use standard C++ compiler and libraries of the abovementioned platform while testing your homework.