COSC2430 Hw3: Database

1. Introduction

You will create a C++ program to manage a database. Read a file that has a lot of items that stand for the information of a name list of certain class. You will need to add into the database and you will also need to delete records in the database. Then sort the records by specific attributes to alphabetical ascending order, and output to a plain text file.

2. Input and Output

a. Input file

1) The input file has multiple records (number > 1). Your program should read the records one by one from the beginning of the file to the end.
2) Each record has uniform attributes (columns), the attributes appear in a fixed order, a record should contain all the keys, no empty values part will be given. But the input file may contain duplicated records. If two records have same id value, they can be recognized as duplicated records. You should NOT update record with the latter one. Other than useful information in records, no other character or space will be given.
3) Each record takes one line (ends in '\n'), and one line only contains one record.
4) The id value always uses 5 digits. DOB values always with the format MM

DD YYYY, which are reasonable date. Position is always one word. Salary will always be an integer. Years worked will always be an integer.

5) All the keys are case sensitive and with the exact words shown below in example. All the values should be outputted exactly as input, you should not change them to upper or lower case or truncate them.
6) When adding a record, use the information of that record. When deleting a record, use "delete id". For one record, the id is the only key word, which cannot appear twice in your output file. Do not add duplicates. Deleted record should not be shown in output file unless it is added again. All records should be processed sequentially from begin to end.
7) An array is not allowed. A linked list must be used.

b. command file

1) The command file only contains commands sort (sort id, sort DOB and etc.) add at end, add at beginning, add after (index here), add before (index here), delete (id), delete (index) to (index), delete at beginning, delete at end, output original. Each line ends in '\n'. Notice that each line only contains one command, and one command only contained in one line.
2) The outputted result should base on the sequence of commands, the commands should be executed from top to bottom sequentially. The latter commands should always base on the former commands' output.
3) Every time you execute a command you should output the modified database into the output file.

c. output file

   1) The output file should be clean, one record in one line, the line should end in ‘\n’. No space in records.
   2) The outputted records in the output file should keep the same keys and values of input file, the only differences are the outputted file is ordered.

3. Program specification and Examples

The main C++ program will become the executable to be tested by the TAs. The result file should be write to another text file (output file), provided with the command line. Notice the input and output files are specified in the command line, not inside the C++ code. All the necessary parameters will be in the command line when calling your program, so you don’t need to worry about the missing file name problem. When calling your program, the format would be one of the two standard types as below, no mixed calling type will be given. Notice also the quotes in the program call, to avoid Unix/Windows get confused.

The general call to the executable is as follows:

```
database “input=input1.txt;output=output1.txt;command=command1.txt”
```

Call example with another command line type.

```
database input=input1.txt output=output1.txt command=command1.txt
```

Example 1 of input and output

input11.txt
```
18513 Lorene Cox 01 11 1978 25 148200 Manager
28931 Edwin Porter 12 24 1999 1 42931 Secretary
98753 Van Goodman 03 15 1998 2 50181 Janitor
```

command11.txt
```
sort salary
add after 1
31230 Donald Trump 12 01 1945 20 1000000 President
delete 2 to 3
```

Command line:
```
sort input=input11.txt output=output11.txt sort=sort11.txt
```

output11.txt
```
28931 Edwin Porter 12 24 1999 1 42931 Secretary
98753 Van Goodman 03 15 1998 2 50181 Janitor
18513 Lorene Cox 01 11 1978 25 148200 Manager

28931 Edwin Porter 12 24 1999 1 42931 Secretary
98753 Van Goodman 03 15 1998 2 50181 Janitor
31230 Donald Trump 12 01 1945 20 100000 President
18513 Lorene Cox 01 11 1978 25 148200 Manager

18513 Lorene Cox 01 11 1978 25 148200 Manager
28931 Edwin Porter 12 24 1999 1 42931 Secretary
```

Example 2 of input and output

input12.txt
```
51249 Harvey Byrd 11 11 1911 90 1 CEO
23192 Cheryl Abbott 10 31 1990 2 500000 Clown
49512 Karl Walsh 07 26 1979 10 481822 Manager
10559 Mike Gutierrez 02 19 1964 15 125821 Developer
```

command12.txt
```
sort position
add before 3
08823 Julian Munoz 08 17 1994 3 109000 Memer
delete 08823
output original
```

Command line:
sort “input=input12.txt;output=output12.txt;command=command12.txt”

output12.txt

```
51249 Harvey Byrd 11 11 1911 90 1 CEO
23192 Cheryl Abbott 10 31 1990 2 500000 Clown
10559 Mike Gutierrez 02 19 1964 15 125821 Developer
49512 Karl Walsh 07 26 1979 10 481822 Manager

51249 Harvey Byrd 11 11 1911 90 1 CEO
23192 Cheryl Abbott 10 31 1990 2 500000 Clown
10559 Mike Gutierrez 02 19 1964 15 125821 Developer
08823 Julian Munoz 08 17 1994 3 109000 Memer
49512 Karl Walsh 07 26 1979 10 481822 Manager

51249 Harvey Byrd 11 11 1911 90 1 CEO
23192 Cheryl Abbott 10 31 1990 2 500000 Clown
10559 Mike Gutierrez 02 19 1964 15 125821 Developer
49512 Karl Walsh 07 26 1979 10 481822 Manager

51249 Harvey Byrd 11 11 1911 90 1 CEO
23192 Cheryl Abbott 10 31 1990 2 500000 Clown
49512 Karl Walsh 07 26 1979 10 481822 Manager
10559 Mike Gutierrez 02 19 1964 15 125821 Developer
```

Example 3 of input and output

input13.txt

```
10001 Anna Lawson 01 14 1956 23 1000001 Janitor
10002 Darlene Daniel 02 14 2001 1 86752 Developer
10003 Marco Rios 07 12 1977 3 49123 Maintenance
10004 Noel Walker 09 12 1987 1 1 CEO
10005 Sally Jackson 05 03 1988 5 223011 Accountant
10006 Winston Moran 05 15 1988 7 41412 Manager
10007 Guillermo Dunn 06 21 1998 12 78865 Cook
10008 May Marshall 07 12 1987 14 41231 Doctor
10009 Harvey Willis 08 13 1995 11 132311 Dentist
10010 Patti Haynes 12 25 2000 15 1337 Mascot
```

command13.txt
sort first name
add at beginning
23192 Cheryl Abbott 10 31 1990 2 500000 Clown

```
10007 John Smith 01 01 2000 1 10 1000 Construction
delete 23192
delete 10007
delete 5 to 8
add at end
23192 Cheryl Abbott 10 31 1990 2 500000 Clown
10007 John Smith 01 01 2000 1 10 1000 Construction
```

Command line:
sort input=input13.txt output=output13.txt sort=sort13.txt

output13.txt

```
10001 Anna Lawson 01 14 1956 23 1000001 Janitor
10002 Darlene Daniel 02 14 2001 1 86752 Developer
10007 Guillermo Dunn 06 21 1998 5 78865 Cook
10009 Harvey Willis 08 13 1995 11 132311 Dentist
10003 Marco Rios 07 12 1977 3 49123 Maintenance
10008 May Marshall 07 12 1987 14 41231 Doctor
10004 Noel Walker 09 12 1987 1 1 CEO
10010 Patti Haynes 12 25 2000 15 1337 Mascot
10005 Sally Jackson 05 03 1988 5 223011 Accountant
10006 Winston Moran 05 15 1988 7 41412 Manager

23192 Cheryl Abbott 10 31 1990 2 500000 Clown
10001 Anna Lawson 01 14 1956 23 1000001 Janitor
10002 Darlene Daniel 02 14 2001 1 86752 Developer
10007 Guillermo Dunn 06 21 1998 5 78865 Cook
10009 Harvey Willis 08 13 1995 11 132311 Dentist
10003 Marco Rios 07 12 1977 3 49123 Maintenance
10008 May Marshall 07 12 1987 14 41231 Doctor
10004 Noel Walker 09 12 1987 1 1 CEO
10010 Patti Haynes 12 25 2000 15 1337 Mascot
10005 Sally Jackson 05 03 1988 5 223011 Accountant
10006 Winston Moran 05 15 1988 7 41412 Manager

10001 Anna Lawson 01 14 1956 23 1000001 Janitor
10002 Darlene Daniel 02 14 2001 1 86752 Developer
10007 Guillermo Dunn 06 21 1998 5 78865 Cook
10009 Harvey Willis 08 13 1995 11 132311 Dentist
10003 Marco Rios 07 12 1977 3 49123 Maintenance
10008 May Marshall 07 12 1987 14 41231 Doctor
10004 Noel Walker 09 12 1987 1 1 CEO
10010 Patti Haynes 12 25 2000 15 1337 Mascot
10005 Sally Jackson 05 03 1988 5 223011 Accountant
```

```
10006 Winston Moran 05 15 1988 7 41412 Manager

10001 Anna Lawson 01 14 1956 23 1000001 Janitor
10002 Darlene Daniel 02 14 2001 1 86752 Developer
10009 Harvey Willis 08 13 1995 11 132311 Dentist
10003 Marco Rios 07 12 1977 3 49123 Maintenance
10008 May Marshall 07 12 1987 14 41231 Doctor
10004 Noel Walker 09 12 1987 1 1 CEO
10010 Patti Haynes 12 25 2000 15 1337 Mascot
10005 Sally Jackson 05 03 1988 5 223011 Accountant
10006 Winston Moran 05 15 1988 7 41412 Manager

10001 Anna Lawson 01 14 1956 23 1000001 Janitor
10002 Darlene Daniel 02 14 2001 1 86752 Developer
10009 Harvey Willis 08 13 1995 11 132311 Dentist
10003 Marco Rios 07 12 1977 3 49123 Maintenance
10008 May Marshall 07 12 1987 14 41231 Doctor

10001 Anna Lawson 01 14 1956 23 1000001 Janitor
10002 Darlene Daniel 02 14 2001 1 86752 Developer
10009 Harvey Willis 08 13 1995 11 132311 Dentist
10003 Marco Rios 07 12 1977 3 49123 Maintenance
10008 May Marshall 07 12 1987 14 41231 Doctor
23192 Cheryl Abbott 10 31 1990 2 500000 Clown
10007 John Smith 01 01 2000 1 10 1000 Construction
```

4. Requirements summary

- C++:

  It is encouraged you do not use existing STL, vector classes since you will have to develop your own C++ classes and functions in future homework.

  Your C++ code must be clear, indented and commented.

  You must create a struct or a class to save each whole record and the elements of the record. The whole record is used for eliminating duplicate ones while the elements of the record are used for sorting.

There are 2000 records at most, so if you want to create a fixed memory space, that is the number. Every time when deleting a record, you should adjust the pointer to its next, so that you won't lost all record followed.

Include comments in each source code file.

Your program will be tested with GNU C++. Therefore, you are encouraged to work on Unix. You can use other C++ compilers, but the TAs cannot provide support or test your programs with other compilers.

- Output:

  The output file must contain the result, in the same format as input. Pay attention to the order of attributes, disordered attributes would be treated as wrong answer.

  Your program should write error messages to the standard output (cout, ptintf).

  Your program should output result within 5 seconds, after will be killed by system.

5. Turn in your homework

Homework 3 need to be turned in to our Linux server, follow the link here http://www2.cs.uh.edu/~rizk/homework.html.

Make sure to create a folder under your root directory, name it hw3 (name need to be lower case), only copy your code to this folder, no testcase or other files needed.

ps. This document may have typos, if you think something illogical, please email TAs or Teacher for confirmation.

Questions and Answers:

Question: My code has a segmentation fault and I do not know why.

Answer: It is likely because you forgot to assign your pointers to nullptr or you are accessing a part of a dynamic array that has not been initialized or not been allocated.

Question: If we are told to delete an id in the command file but the ID is not in the database what should we do?

Answer: You will not be asked to delete an ID not in the database.

Question: If we delete the entire array and we are asked to add before 1, what do we do?

Answer: Adding before 1 will not be given in an empty array

Question: What are the commands?

Answer: The commands are:

sort id

sort first name

sort last name

sort DOB (there is no sort month, no sort day, no sort year) sort years worked

sort position

add before (index)

add after (index)

delete (index) to (index), this command does not only delete what is in between

the indexes but deletes the indexes as well

delete (id)

delete at beginning

delete at end

output original