**CPP Problem Design**

|  |
| --- |
| **Contributor︰KWY** |
| **Subject：** |
| **Main testing concept：STANDARD TEMPLATE LIBRARY**   |  |  | | --- | --- | | **Basics** | **Functions** | | ■ C++ BASICS  ■ FLOW OF CONTROL  □ FUNCTION BASICS  □ PARAMETERS AND OVERLOADING  □ ARRAYS  □ STRUCTURES AND CLASSES  □ CONSTRUCTORS AND OTHER TOOLS  □ OPERATOR OVERLOADING, FRIENDS,AND REFERENCES  □ STRINGS  □ POINTERS AND DYNAMIC ARRAYS | □ SEPARATE COMPILATION AND NAMESPACES  □ STREAMS AND FILE I/O  □ RECURSION  □ INHERITANCE  □ POLYMORPHISM AND VIRTUAL FUNCTIONS  □ TEMPLATES  □ LINKED DATA STRUCTURES  □ EXCEPTION HANDLING  ■ STANDARD TEMPLATE LIBRARY  □ PATTERNS AND UML | |
| **Description：**  Please implement a library database system to store some information of books, including the title, the author, and the editions of book. You will also need to implement the following commands to manage the database.   * **Insert:** To insert a datum takes two strings, **title** and **author**, and an integer **edition** as input, adds the book datum into the database, and outputs the message of “*Insert* ***author****'s* ***title****, Edition:* ***edition****.\n*” if it does not exist. Otherwise, please output the message “*Datum already exist.\n*”. * **Delete Edition:** To delete a datum takes two strings, **title** and **author**, and an integer **edition** as input, locates and removes the specific **edition** of book datum matching the **title** and **author** from the database, and outputs the message of “*Delete* ***author****'s* ***title****, Edition:* ***edition****.\n*” if it exists. Otherwise, output the message of “*Datum doesn't exist.**\n*”. * **Delete Book:** To delete a set of data takes two strings, **title** and **author** as input, deletes all data matching the record of **title** and **author** in the database, and outputs a message of “*Delete* ***author****'s* ***title****. Edition:* ***edition****.\n*” for each deletion if there is at least one record. Otherwise, output the message of “*Book doesn't exist.\n*” * **Find Book:** To find data take two strings, **title** and **author**, as input, finds the book datum matching the **title** and **author** in the database, and output a message of “*Title:* ***title****\tAuthor:* ***author****\tEdition:* ***edition****s <E1, E2, …>\n*”, for all records where E1, E2, … are sorted based on their editions if there is at least one record. Otherwise, output a “*Book doesn't exist.\n*” * **Find Author:** To find all books written by an author takes a string, **author**, as input, finds all the book written by **author**, and output the message of “***author****'s Books:* ***title****s <T1, T2, …>\n*” where T1, T2, … are sorted with string relation operator < if there is at least one datum. Otherwise, output a message of “*No book found.\n*” * **Sort by Title:** To sort the database based on the **title** takes no input, sorts the database by comparing the **title** with string relation operator < while comparing the **author** with string operator < for those having the same **title**s, and outputs the result in the format of “*Title:* ***title****\tAuthor:* ***author****\tEdition:* ***edition****s <E1, E2, …>\n*”, for all records where E1, E2, … are sorted based on their editions. * **Sort by Author:** To sort the databased based on the **author** takes no input, sorts the database by comparing the **author** with string relation operator < while comparing the **title** with string operator < for those having the same **author**s, and outputs the result in the format of “*Title:* ***title****\tAuthor:* ***author****\tEdition:* ***edition****s <E1, E2, …>\n*”, for all records where E1, E2, … are sorted based on their editions.   **Input：**  There are seven different commands while each command is issued in a line.   1. **Insert**: Insert "title" "author" "edition" 2. **Delete Edition**: Delete Edition "title" "author" "edition" 3. **Delete Book**: Delete Book "title" "author" 4. **Find Book**: Find Book "title" "author" 5. **Find Author**: Find Author "author" 6. **Sort by Title**: Sort by "title" 7. **Sort by Author**: Sort by "author"   User can keep entering commands until reading EOF.  **Output：**   1. The output message with corresponding input command. 2. If the command doesn’t exist, output the message of “*Unknown Command.\n*”. 3. If the command is not complete, output the message of “*Incomplete Command.\n*”   See the sample output.  **Sample Input / Output :**   |  |  | | --- | --- | | **Sample Input** | **Sample Output** | | Insert "Harry" "JK" 1  Insert "Harry" "JK" 1  Insert "Rapunzel" "Unknown" 1812  Insert "Momotaro" "Unknown" 17  Insert "Harry" "JK" 2  Insert "Harry" "JK" 4  Insert "Rapunzel" "Glen Keane" 2010  Find Book "Harry" "JK"  Find Author "Unknown"  Sort by Title  Find Author "KK"  Delete Edition "Harry" "JK" 3  Delete Book "Mamamia" "ABBA"  Delete Edition "Harry" "JK" 1  Delete Book "Rapunzel" "Glen Keane"  Sort by Author  insert  Insert 5 | Insert JK's Harry, Edition: 1.  Datum already exist.  Insert Unknown's Rapunzel, Edition: 1812.  Insert Unknown's Momotaro, Edition: 17.  Insert JK's Harry, Edition: 2.  Insert JK's Harry, Edition: 4.  Insert Glen Keane's Rapunzel, Edition: 2010.  Title: Harry Author: JK Edition: 1, 2, 4  Unknown's Books: Momotaro, Rapunzel  Title: Harry Author: JK Edition: 1, 2, 4  Title: Momotaro Author: Unknown Edition: 17  Title: Rapunzel Author: Glen Keane Edition: 2010  Title: Rapunzel Author: Unknown Edition: 1812  No book found.  Datum doesn't exist.  Book doesn't exist.  Delete JK's Harry, Edition: 1.  Delete Glen Keane's Rapunzel.  Title: Harry Author: JK Edition: 2, 4  Title: Momotaro Author: Unknown Edition: 17  Title: Rapunzel Author: Unknown Edition: 1812  Unknown Command.  Incomplete Command. | |
| **□** **Easy, only basic programming syntax and structure are required.**  **■ Medium, multiple programming grammars and structures are required.**  **□ Hard, need to use multiple program structures or complex data types.** |
| **Expected solving time:**  35minutes |
| **Other notes:**  You are suggested to use STL such as std::pair, std::map, std::set and std::vector, to implement the database. |