CISC 1110 Assignment 8 Spring, 2017 Due May 24, 2017

Write a complete C++ program to do the following: The program will maintain a database of something you own or love, storing information on at least 5 objects. For example, you might create a database of your CD collection or your DVD collection, your books, your family members, or your clothes. Your program will create a class and define an array of objects of that class. The class must have at least two fields, and at least one field must be another nested class. You may use my example below if you do not want to create your own database.

For example, for a CDcollection class, you might store the following information: for each title, the CD title, the artist name (a class, divided into first and last), and the genre (rock, rap, heavy metal, folk, world music, classical, opera). You can store the information with a different organization or include other information, like label (Decca, DG, Sony) or year produced (like 2016).

For example, I might use the following class structure for my CD collection, with the information shown for my first entry:

CD [0]
- title: "Vocalises"
- artist: (a subclass)
  - first name: "Natalie"
  - last name: "Dessay"
- genre: classical vocal
- CDpublisher: (a subclass)
  - label: EMI
  - year: 1998

The description below uses the fields from my sample class above, but if you use a different class, use your own names.

Define the class above main so that you can use the class in all the functions. Declare the actual array of objects in the main program.

readdata

The program will begin by reading in the data from a file. Call a function **readdata** sending it two parameters: the array of objects and n as a reference parameter. The function will read in n and then info on n objects (items). For each item, read in a value for each of the fields in the class. Put the data in the order of the members of the class (as shown above). If any titles or names are more than one word (like "Hits of 2012"), put underbars between the words ("Hits_of_2012") so that you can read everything with the extraction operator (filein >>....).

printall

Next, the program will print all the data. Call a function **printall**, sending it two parameters: the array of objects and n, the number of filled positions in the array. The function should print n, together with an appropriate message. Then, for each item in your collection, the function will print all the info (for example, CD title, artist name, genre, label, and year). As you print, provide labels, like "Title:" Each item (for example, each CD) should start after a blank line, but you may spread the member values over several lines. Before printing the list, print a heading like "Original Database."

sort

Next the program will call a function **sort**, sending it two parameters, the array of objects and n, the number of filled positions in the array. The function will sort the array of objects in ascending order by ONE of

the fields (in my example, I might choose artist last name or title). Make sure to carry along all the other information, as you did in Program 6. Use object assignment so that you can copy one entire object to another object in one statement and sort in only 3 statements. (What data type should temp have?)

Next the program will call **printall** again to print the sorted data. Before printing the data, skip a line and print a heading, like "Database Sorted by Title" (or whatever field you've chosen).

search

Next the program will read in a value and call a search function to find that item in the array of objects. You need to write only one search function, but you may write more if you want to. Choose an appropriate name; in my example, I might call it **findartist**, **findtitle**, **findgenre**, **findlabel**, or **findyear**.

Make the value you read in match the function you've chosen; that is, if you've written **findartist** for the data shown above, read in "Dessay" or some other artist name. If you've written **findlabel**, read in "EMI" or "Virgin" or some other label.

Send your chosen function three parameters: the array of objects, n (the number of filled positions in the array), and the value to search for. The search function will first **print the item searched for**, together with a message like "Searching title for Vocalises", and then search for the item in the array.

IMPORTANT NOTE: The search function will find ALL the occurrences of the requested item, not just one. Don't leave the function until you've processed the entire array.

printone

Each time the item is found, the search function will print a message like "Vocalises found" and call a function **printone** to print all of the information for the object in the array that matched the value searched for. For example, if you were searching for "EMI", you'd call **printone** once to print the artist name, title, genre, label and year for CD[0], and once for any other CD's on the EMI label. If the item is not found, the search function should print "not found". The only parameter sent to **printone** is the single element from the array of objects to be printed, such as **myCDs[0]**. (You could send a subscript, but I want you to practice passing this type of parameter.)

Summary of Functions:

readdata: Two parameters: the array of objects and n, a reference parameter, both changed in function.

printall: Two parameters: the array and n, number of filled positions in the array.

printone: One parameter: a single object from the array of objects.

One search function, like **findartist**, **findtitle**, **findgenre**, **findlabel**, or **findyear,** that finds all occurrences of an item in your database. Three parameters: the array of objects; n, the number of filled positions in the array; and value, an item to search for.

sort: Two parameters: the array and n, number of filled positions in the array.

DATA: Your arrays should have room for up to 15 items. To test the program, have a set of data with at least 5 items. Make sure at least 2 items repeat the field you search for (for example, if you search for genre, make sure you have two CDs with the same genre).