Serendipity Booksellers Software Development Project— Part 11: A Problem-Solving Exercise

For this chapter's assignment you are going to replace the arrays you added in Chapter 7 with a single array of structures. You will also modify the functions so they work with this single array instead of the multiple related arrays. This will simplify the program and set the stage for the next two chapters' assignments.

1. **Create the structure declaration.**
   Create a `BookData` structure declaration with the following members:

`bookTitle`:	An array of characters. The array should have 51 elements. This makes it large enough to store a book title of up to 50 characters in length.
`isbn`:	An array of characters. The array should have 14 elements. This makes it large enough to store an ISBN entry of up to 13 characters in length.
`author`:	An array of characters. The array should have 31 elements. This makes it large enough to store an author's name of up to 30 characters in length.
`publisher`:	An array of characters. The array should have 31 elements. This makes it large enough to store the name of a publisher with up to 30 characters.
`dateAdded`:	An array of characters that will hold the date the book was added to the inventory. The array should have 11 elements. This makes it large enough to store a date, as a string, with 11 characters. The date should be stored in the form MM-DD-YYYY. For example, April 2, 2012 would be stored as 04-02-2012.
`qtyOnHand`:	An integer. This member will hold the quantity on hand of a book.
`wholesale`:	A `double`. This member will hold the wholesale price of a book.
`retail`:	A `double`. This member will hold the retail price of a book.

2. **Add the following functions to the program.**

`setTitle`:	This function accepts as arguments a pointer to a string and an integer that will act as a subscript into the array of `BookData` structures. It copies the string to the `bookTitle` member of the array element specified by the subscript. Return value: `void`.
`setISBN`:	This function accepts as arguments a pointer to a string and an integer that will act as a subscript into the array of `BookData` structures. It copies the string to the `ISBN` member of the array element specified by the subscript. Return value: `void`.
`setAuthor`:	This function accepts as arguments a pointer to a string and an integer that will act as a subscript into the array of `BookData` structures. It copies the string to the `author` member of the array element specified by the subscript. Return value: `void`.
`setPub`:	This function accepts as arguments a pointer to a string and an integer that will act as a subscript into the array of `BookData` structures. It copies the string to the `publisher` member of the array element specified by the subscript. Return value: `void`.
`setDateAdded`:	This function accepts as arguments a pointer to a string and an integer that will act as a subscript into the array of `BookData` structures. It copies the string to the `dateAdded` member of the array element specified by the subscript. Return value: `void`.
`setQty`:	This function accepts as arguments an integer (holding a quantity) and an integer that will act as a subscript into the array of `BookData` structures. It copies the quantity parameter to the `qtyOnHand` member of the array element specified by the subscript. Return value: `void`.
`setWholesale`:	This function accepts as arguments a `double` and an integer that will act as a subscript into the array of `BookData` structures. It copies the `double` to the `wholesale` member of the array element specified by the subscript. Return value: `void`.
`setRetail`:	This function accepts as arguments a `double` and an integer that will act as a subscript into the array of

	`BookData` structures. It copies the `double` to the `retail` member of the array element specified by the subscript. Return value: `void`
`isEmpty`:	This function accepts an integer that will act as a subscript into the array of `BookData` structures as its argument. The function will return an `int`, indicating whether the structure specified by the subscript is empty (1 = empty, 0 = not empty). If the first character of the `bookTitle` member is zero (the null terminator), the function returns 1. Otherwise a 0 is returned.
`removeBook`:	This function accepts an integer that will act as a subscript into the array of `BookData` structures as its argument. The function will remove a book from inventory. It does so by simply setting the first character of the `bookTitle` member to zero (the null terminator).

3. Replace the multiple arrays with a single array of structures.

The program currently uses the following arrays:

```
bookTitle
isbn
author
publisher
dateAdded
qtyOnHand
wholesale
retail
```

Remove these arrays from the program and replace them with a global array of 20 of the `BookData` structures you declared in Step 1.

4. Modify the `addBook` function.

Change the `addBook` function so it works with the array of structures you created in Step 2 above. When a new book is added to the inventory, the program will step through the array, calling the `IsEmpty` function of each structure. When it finds an empty structure, it will ask the user for the book's data. The function will then call the appropriate functions to set the structure's members to the new data. (See step 2 for a description of the functions.)

5. **Modify the `lookUpBook` function.**

   The `lookUpBook` function should be changed to search the structure array for a book whose title matches the user's input. When a book is found, its data should be passed to the `bookInfo` function.

6. **Modify the `editBook` function.**

   The `editBook` function should be changed to search and modify data in the structure array. When it finds a book whose data the user wishes to modify, it should pass the new data to the appropriate functions. (See Step 2 for a description of the functions).

7. **Modify the `deleteBook` function.**

   The `deleteBook` function should be changed to work with the structure array. When a book is to be removed from inventory, this function should search for it in the structure array, then call the `removeBook` function to delete it.

8. **Modify the `cashier` function.**

   The `cashier` function should be modified to work with the structure array. When a sale takes place, the function should search the structure array for the needed data and retrieve it. All other operations of the `cashier` function will remain the same.