```
#include <iostream>
#include "structFile.h"
#include <fstream>
#include <string>
#include <iomanip>
#include <algorithm>

using namespace std;

bool book::salesGreaterThan(int otherSalesNum)  ///This member function checks if the sales of one book struct is greater than the sales of the input book struct
{
   if(totalSales > otherSalesNum)
      return true;
   else
      return false;
}

bool author::dateOlderThan(int otherDate) ///This member function checks if the year of one date struct is less than the date of the input date struct
{
   if(d.year < otherDate)
      return true;
   else
      return false;
}

void book::sortBookSales(book* b, int &n) ///This member function sorts the books by sales form highest to lowest
{
```

```
string title1 = " ";
string name1 = " ";
int year1 = 0;
string lang1 = " ";
string genre1 = " ";
int sales1 = 0;
for(int i = 0; i < n; i++)
{
  for(int j = 0; j < n; j++)
  {
    if(b[i].salesGreaterThan(b[j].totalSales)) ///If the current sales are greater than the given one, it places the higher number above the lower in the array
    {
      title1 = b[j].bookTitle;
      name1 = b[j].authorName;
      year1 = b[j].englishPubYear;
      lang1 = b[j].originalLang;
      genre1 = b[j].genre;
      sales1 = b[j].totalSales;
      b[j].bookTitle = b[i].bookTitle;
      b[j].authorName = b[i].authorName;
      b[j].englishPubYear = b[i].englishPubYear;
      b[j].originalLang = b[i].originalLang;
      b[j].genre = b[i].genre;
      b[j].totalSales = b[i].totalSales;
      b[i].bookTitle = title1;
      b[i].authorName = name1;
      b[i].englishPubYear = year1;
      b[i].originalLang = lang1;
```

```
            b[i].genre = genre1;
            b[i].totalSales = sales1;
        }
    }
  }
}

void book::printBooksToFile(book* b, int &n) ///This function prints the sorted book information to an output file called "OrderedBooks.txt"
{
  ofstream outFile("OrderedBooks.txt");
  outFile << left << setw(45) << " " << "Books" << endl << endl;
  outFile << left << setw(35) << "Title" << setw(25) << "Author" << setw(20) << "Release Year" << setw(20) << "Language"
  << setw(30) << "Genre" << setw(20) << "Total Sales" << endl << endl;
  for (int i = 0; i < n; i++)
  {
    outFile << left << setw(35) << b[i].bookTitle << setw(30) << b[i].authorName << setw(15) << b[i].englishPubYear <<
    setw (20) << b[i].originalLang << setw(20) << b[i].genre << setw(20) << b[i].totalSales << endl;
  }
  outFile << endl;
}

void author::printAuthorsToFile(author* a, date* d, int &n) ///This function prints the sorted author information to an output file called "OrderedAuthors.txt"
{
  ofstream outFile("OrderedAuthors.txt");
  outFile << left << setw(30) << " " << "Authors" << endl << endl;
```

```
   outFile << left << setw(22) << "Name" << "Date of Birth" << right << setw(22) << "Notable Work" << endl << endl;
   for (int i = 0; i < n; i++)
   {
      outFile << left << setw(25) << a[i].name;
      outFile << d[i].month << "/" << d[i].year;
      outFile << right << setw(30) << a[i].mostNoteBook << endl;
   }
   outFile << endl;
}



void library::initializeLibrary(library* l, struct book* b1, struct author* a1, struct date* d1, int &n, int &m)
{
   for(int i = 0; i < n; i++)
   {
      l->b[i].authorName = b1[i].authorName;
      l->b[i].bookTitle = b1[i].bookTitle;
      l->b[i].englishPubYear = b1[i].englishPubYear;
      l->b[i].genre = b1[i].genre;
      l->b[i].originalLang = b1[i].originalLang;
      l->b[i].totalSales = b1[i].totalSales;
   }


   for(int j = 0; j< m; j++)
   {
      l->a[j].name = a1[j].name;
      l->a[j].mostNoteBook = a1[j].mostNoteBook;
```

```
      l->a[j].nationality = a1[j].nationality;
      l->a[j].d.day = a1[j].d.day;
      l->a[j].d.month = a1[j].d.month;
      l->a[j].d.year = a1[j].d.year;
   }
}
```

```
void sortAuthorDate(struct author* a, struct date* d, int &n) ///This function sorts the authors by oldest date of birth
{
   string name1 = " ";
   string book1 = " ";
   string nationality1 = " ";
   int day1 = 0;
   int month1 = 0;
   int year1 = 0;
   for(int i = 0; i < n; i++)
   {
      for(int j = 0; j < n; j++)
      {
         if(a[i].dateOlderThan(a[j].d.year)) ///if the current date of birth is older than another, it places the older date of birth above the newer
         {
            name1 = a[j].name;
            book1 = a[j].mostNoteBook;
            nationality1 = a[j].nationality;
            day1 = a[j].d.day;
            month1 = a[j].d.month;
```

```
            year1 = a[j].d.year;
            a[j].name = a[i].name;
            a[j].mostNoteBook = a[i].mostNoteBook;
            a[j].nationality = a[i].nationality;
            a[j].d.day = a[i].d.day;
            a[j].d.month = a[i].d.month;
            a[j].d.year = a[i].d.year;
            d[j].day = a[i].d.day;
            d[j].month = a[i].d.month;
            d[j].year = a[i].d.year;
            a[i].name = name1;
            a[i].mostNoteBook = book1;
            a[i].nationality = nationality1;
            a[i].d.day = day1;
            a[i].d.month = month1;
            a[i].d.year = year1;
            d[i].day = day1;
            d[i].month = month1;
            d[i].year = year1;
        }
      }
    }
}

void initializeBookArray(struct book* b, int &n) ///This function is used to initialize the array of books
{
   ifstream input("Books-2.txt"); ///Read the given text file
   if(!input) ///If the input fails, indicate this to the user
   {
```

```
    cout << "Input failed." << endl;
  }
  input >> n;
  for(int i = 0; i < n; i++) ///Input the given information from the file into each attribute at the given index
  {                  ///of the array of structs
    input >> b[i].bookTitle;

    input >> b[i].authorName;

    input >> b[i].englishPubYear;

    input >> b[i].originalLang;

    input >> b[i].genre;

    input >> b[i].totalSales;

    replace(b[i].bookTitle.begin(), b[i].bookTitle.end(), '_', ' '); ///Replace the underscores with spaces in the
    replace(b[i].authorName.begin(), b[i].authorName.end(), '_', ' '); ///appropriate positions
    b->sortBookSales(b, n);
  }
}

void printBookArray(struct book* b, int &n) ///This function will print the given array of books in the proper format
{
  cout << left << setw(45) << " " << "Books" << endl << endl;
```

```
   cout << left << setw(35) << "Title" << setw(25) << "Author" << setw(20) << "Release Year" <<
setw(20) << "Language"
   << setw(30) << "Genre" << setw(20) << "Total Sales" << endl << endl;
   for (int i = 0; i < n; i++)
   {
      cout << left << setw(35) << b[i].bookTitle << setw(30) << b[i].authorName << setw(15) <<
b[i].englishPubYear <<
      setw (20) << b[i].originalLang << setw(20) << b[i].genre << setw(20) << b[i].totalSales << endl;
   }
   cout << endl;
   b->printBooksToFile(b, n); ///runs the function to print the book information to the output file
}

void initializeAuthorArray(struct author* a, struct date* d, int &n) ///This function will initialize both
the author
{                                          ///and date arrays
   char dummy;
   ifstream input("Authors-2.txt");
   if(!input)
   {
      cout << "Input failed." << endl;
   }
   input >> n;
   for(int j = 0; j < n; j++) ///input the given information into the correct attributes within the given
array of structs
   {
      input >> a[j].name;
      input >> a[j].d.month >> dummy >> a[j].d.day >> dummy >> a[j].d.year;
      input >> a[j].mostNoteBook;
      d[j].month = a[j].d.month;
```

```
    d[j].day = a[j].d.day;
    d[j].year = a[j].d.year; ///Copies the information from the date struct within the author struct into the date array
    replace(a[j].mostNoteBook.begin(), a[j].mostNoteBook.end(), '_', ' ');
    replace(a[j].name.begin(), a[j].name.end(), '_', ' '); ///Replaces the underscores with spaces in the
  }
    sortAuthorDate(a, d, n);     ///appropriate positions
}

void printAuthorArray(struct author* a, struct date* d, int &n) ///This function will print the given array of authors
{                                    ///in the proper format
  cout << left << setw(30) << " " << "Authors" << endl << endl;
  cout << left << setw(22) << "Name" << "Date of Birth" << right << setw(22) << "Notable Work" << endl << endl;
  for (int i = 0; i < n; i++)
  {
    cout << left << setw(25) << a[i].name;
    cout << d[i].month << "/" << d[i].year;
    cout << right << setw(30) << a[i].mostNoteBook << endl;
  }
  cout << endl;
  a->printAuthorsToFile(a, d, n); ///runs the function to print the author information to the output file
}

void matchAuthors(struct book* b, struct author* a)///This function checks to see if any of the authors in the books array
{                                    ///match any of the authors in the authors array
  cout << left << setw(22) << " " << "Authors Matched" << endl << endl;
```

```
   cout << left << setw(20) << "   Name" << setw(30) << "Nationality" << setw(25) << "Book" << endl
<< endl;

   for (int i = 0; i < 11; i++)

   {

      for(int j = 0; j < 7; j++)

      {

         if(b[i].authorName == a[j].name)///If the authors match, it prints their name, their nationality
(the original langage

         {                        ///of their books), and the title of the book that appears on the books array

            cout << left << setw(20) << b[i].authorName << setw(25) << b[i].originalLang << setw(25) <<
b[i].bookTitle << endl;

         }

      }

   }

   cout << endl;

}
```