Tyler Tackett

tjtackett

CS1300 AA Fall 2019

Program 3

Classes

12 /10 / 2019

Instructions

CS1300 Programming Project 3 – Classes Fall 2019

~~Due December 4, 2019~~ Mansfield, J

Design, test, implement and test a C++ program that uses class files to implement Programming Exercise #10, page 732 (also copied below) meeting all of the specifications.

You will have a UML and other design documents.

You will have at least 3 files – these are required:

bookType.h
bookTypeImp.cpp

bookTypeUse.cpp

You should plan this program well. Include in your planning your testing methods for each required function.

Make sure to include all of your documentation for: a) planning; b) test and evaluation design; c) reflection.

Turn in all required documentation:

- Cover page
- Copy of assignment
- Design documents – including all uml diagrams; testing design and any additional algorithms
- Reflection

Additionally, drop tarred and gzipped file in cs1300.drop.

~~Due Date: 12/4/19~~

10. a. Some of the characteristics of a book are the title, author(s), publisher, ISBN, price, and year of publication. Design a `class bookType` that defines the book as an ADT.

    i. Each object of the `class bookType` can hold the following information about a book: title, up to four authors, publisher, ISBN, price, and number of copies in stock. To keep track of the number of authors, add another member variable.

    ii. Include the member functions to perform the various operations on objects of type `bookType`. For example, the usual operations that can be performed on the title are to show the title, set the title, and check whether a title is the same as the actual title of the book. Similarly, the typical operations that can be performed on the number of copies in stock are to show the number of copies in stock, set the number of copies in stock, update the number of copies in stock, and return the number of copies in stock. Add similar operations for the publisher, ISBN, book price, and authors. Add the appropriate constructors and a destructor (if one is needed).

    b. Write the definitions of the member functions of the `class bookType`.

    c. Write a program that uses the `class bookType` and tests various operations on the objects of the `class bookType`. Declare an array of 100 components of type `bookType`. Some of the operations that you should perform are to search for a book by its title, search by ISBN, and update the number of copies of a book.

Design

Problem -> Write a class and implement is operations

Files -> bookType.h

bookTypeImp.cpp

bookTypeUse.cpp

Class -> bookType

Member Variables ->

String Title

Array String Authors [Number of Authors]

String Publisher

Int ISBM

Double Price

Int Number of Copies in Stock

Int Number of Authors (up to 4)

Member Functions ->

Title Functions -> String getTitle (String Title)

Void setTitle (String& Title)

Void searchForTitle (String Title, bookType myBooks[])

Number of Copies in Stock Functions ->

Int geNumberOfCopies (int Number of Copies in Stock)

Void setNumberOfCopies (int& Number of Copies in Stock)

Authors Functions -> String getNameOfAuthors (string Authors[])

Int getNumberOfAuthors (int Number of Authors)

Void setNumberOfAuthors (int& Number of Authors)

Void setNameOfAuthors (string& Authors[])

Void SearchForAuthor(String Author, bookType myBooks[])

Publisher Functions -> String getNameOfPublisher (string Publisher)

Void setNameOfPublisher (string& Publisher)

Void searchForPublisher (String Publisher, bookType myBooks[])

ISBM Functions -> Int getISBM (int ISBM)

Void setISBM (int& ISBM)

Void searchISBM (Int ISBM, bookType myBooks[])

Price Functions -> Int getPrice (int Price)

Void setPrice (int& Price)

Void searchPrice (Int Price, bookType myBooks[])

UML ->

Book Type
- Title : String - Authors : String [] - Publisher : String - ISBM : Int - Price : Double - Number of Copies in Stock : Int - Number of Authors : Int
+ getTitle (string Title) : String + setTitle (string& Title) : Void + searchForTitle (string Title, bookType myBooks[]) : Void + getNumberOfCopies (int Number of Copies in Stock) : Int + setNumberOfCopies (int& Number of Copies in Stock) : Void + getNameOfAuthors (string Authors[]) : String + getNumberOfAuthors (int Number of Authors) : Int + setNameOfAuthors (string& Authors[]) : Void + setNumberOfAuthors (int& Number of Authors) : Void + searchForAuthor (string Author, bookType myBooks[]) : Void + getNameOfPublisher (string Publisher) : String + setNameOfPublisher (string& Publisher) : Void +searchForPublisher (string Publisher, bookType myBooks[]) : Void + getISBM (int ISBM) : Int + setISBM (int& ISBM) : Void + searchISBM (int ISBM, bookType myBooks[]) : Void + getPrice (double Price) : Double + setPrice (double& Price) : Void + searchPrice (double Price, bookType myBooks[]) : Void

Source Code ->

Header File (bookType.h)

```
#include <string>
using namespace std;
class bookType
{
   //function protoypes
   public:
      void print() const;
      string getFirstAuthorName()  const;
      string getSecondAuthorName() const;
      string getThirdAuthorName() const;
      string getfourthAuthorName() const;

      void setFirstAuthorName(string name);
      void setSecondAuthorName(string name);
      void setThirdAuthorName(string name);
      void setFourthAuthorName(string name);

      string getAuthorNum() const;
      void setAuthorNum(string num);
      bool searchAuthor(string name);

      string getPublisherName() const;
      void setPublisherName(string name);
```

```
bool searchPublisher(string name);

string getTitle() const;
void setTitle(string title);
bool searchTitle(string title);

string getCopies() const;
void setCopies(string copies);

string getYearPublished() const;
void setYearPublished(string year);
bool searchYearPublished(string year);

string getISBM() const;
void setISBM(string num);
bool searchISBM(string num);

string getPrice() const;
void setPrice(string num);
bool searchPrice(string num);

//a custom constructor
bookType(string author = "", string ISBM = "", string copies = "", string year = "",
                string price = "", string title = "", string publishers = "",
```

```
                string firstName = "", string secondName = "", string thirdName = "",
                string fourthName = "");
    //function variables
    private:
        string authorNum;
        string bookISBM;
        string stockCopies;
        string yearBookPublished;
        string bookPrice;
        string bookTitle;
        string publisherName;
        string firstAuthorName;
        string secondAuthorName;
        string thirdAuthorName;
        string fourthAuthorName;
        bool found;
};
```

Implementation File (bookTypeImp.cpp)

```
#include <iostream>
#include <string>
#include "bookType.h"
using namespace std;
//Prints all the variables
```

```
void bookType::print() const
{
   cout << "The Book Title is: " << bookTitle <<
                    ". The number of authors is " << authorNum << ". Thier name(s) are ";
   if(authorNum == "4")
   {
      cout << firstAuthorName << ", " << secondAuthorName << ", " << thirdAuthorName <<
                                                                  " and " << fourthAuthorName;
   }
   else if(authorNum == "3")
  {
      cout << firstAuthorName << ", " << secondAuthorName << " and " << thirdAuthorName;
   }
   else if(authorNum == "2")
   {
      cout << firstAuthorName << " and " << secondAuthorName;
   }
   else if(authorNum == "1")
   {
      cout << firstAuthorName;
   }
   else
   {
      cout << "unknown";
   }
```

```
   cout << ". The publisher is " << publisherName << ". The book was published in the year "
<< yearBookPublished

       << ". We have " << stockCopies << " copies of the book. Each copy is priced at " <<
bookPrice << ". The ISBM is " << bookISBM << "." << endl << endl;

}

std::string bookType::getFirstAuthorName() const

{

   return firstAuthorName;

}

std::string bookType::getSecondAuthorName() const

{

   return secondAuthorName;

}

std::string bookType::getThirdAuthorName() const

{

   return thirdAuthorName;

}

std::string bookType::getfourthAuthorName() const

{

   return fourthAuthorName;

}

string bookType::getAuthorNum() const

{

   return authorNum;
```

```
}
void bookType::setFirstAuthorName(std::string firstName)
{
   firstAuthorName = firstName;
}
void bookType::setSecondAuthorName(std::string secondName)
{
   secondAuthorName = secondName;
}
void bookType::setThirdAuthorName(std::string thirdName)
{
   thirdAuthorName = thirdName;
}
void bookType::setFourthAuthorName(std::string fourthName)
{
   fourthAuthorName = fourthName;
}
void bookType::setAuthorNum(string author)
{
   authorNum = author;
}
bool bookType::searchAuthor(std::string name)
{
   if((firstAuthorName == name)||(secondAuthorName == name)||(thirdAuthorName ==
name)||(fourthAuthorName == name))
```

```
    {
        found = true;
    }
    else
    {
        found = false;
    }
    return found;
}
std::string bookType::getPublisherName() const
{
    return publisherName;
}
void bookType::setPublisherName(std::string publisher)
{
    publisherName = publisher;
}
bool bookType::searchPublisher(std::string publisher)
{
    if(publisherName == publisher)
    {
        found = true;
    }
    else
    {
```

```
        found = false;
    }
    return found;
}
std::string bookType::getTitle() const
{
    return bookTitle;
}
void bookType::setTitle(std::string title)
{
    bookTitle = title;
}
bool bookType::searchTitle(std::string title)
{
    if(bookTitle == title)
    {
        found = true;
    }
    else
    {
        found = false;
    }
    return found;
}
```

```
string bookType::getCopies() const
{
   return stockCopies;
}
void bookType::setCopies(string copies)
{
   stockCopies = copies;
}
string bookType::getYearPublished() const
{
   return yearBookPublished;
}
void bookType::setYearPublished(string year)
{
   yearBookPublished = year;
}
bool bookType::searchYearPublished(string year)
{
   if(yearBookPublished == year)
   {
      found = true;
   }
   else
   {
      found = false;
```

```
    }
    return found;
}
string bookType::getISBM() const
{
    return bookISBM;
}
void bookType::setISBM(string ISBM)
{
    bookISBM = ISBM;
}
bool bookType::searchISBM(string ISBM)
{
    if(bookISBM == ISBM)
    {
        found = true;
    }
    else
    {
        found = false;
    }
    return found;
}
string bookType::getPrice() const
{
```

```
    return bookPrice;
}
void bookType::setPrice(string price)
{
    bookPrice = price;
}
bool bookType::searchPrice(string price)
{
    if(bookPrice == price)
    {
        found = true;    }
    else
    {
        found = false;
    }
    return found;
}
//connects the class's variables to the constructors values
bookType::bookType(string author, string ISBM, string copies, string year,
                string price, string title, string publisher, string firstName, string secondName, string thirdName, string fourthName)
{
    authorNum = author;
    bookISBM = ISBM;
    stockCopies = copies;
```

```
    yearBookPublished = year;
    bookPrice = price;
    bookTitle = title;
    publisherName = publisher;
    firstAuthorName = firstName;
    secondAuthorName = secondName;
    thirdAuthorName = thirdName;
    fourthAuthorName = fourthName;
}
```

Use File (bookTypeUse.cpp)

```
#include <iostream>
#include <fstream>
#include <string>
#include "bookType.h"
using namespace std;

int main()
{
    ifstream inData;
    inData.open("data.txt");
    bookType myBooks[5];
    bool found;
    string line;
```

```
//Checks to see if information can be read from the file
if(inData)
{
   //the loop runs 5 times in order to give all bookTypes their values
   for(int i = 0; i < 5; i++)
   {
      //getLine reads a line from the file and assings it to line
      //a member funcion is then used to assing line to a member variable
      //this process is repeated throughout the program
      getline(inData, line);
      myBooks[i].setAuthorNum(line);
      if("1" == myBooks[i].getAuthorNum())
      {
         getline(inData, line);
         myBooks[i].setFirstAuthorName(line);
      }
      else if("2" == myBooks[i].getAuthorNum())
      {
         getline(inData, line);
         myBooks[i].setFirstAuthorName(line);


         getline(inData, line);
         myBooks[i].setSecondAuthorName(line);
      }
      else if("3" == myBooks[i].getAuthorNum())
```

```
{
   getline(inData, line);
   myBooks[i].setFirstAuthorName(line);

   getline(inData, line);
   myBooks[i].setSecondAuthorName(line);

   getline(inData, line);
   myBooks[i].setThirdAuthorName(line);
}
else if("4" == myBooks[i].getAuthorNum())
{
   getline(inData, line);
   myBooks[i].setFirstAuthorName(line);

   getline(inData, line);
   myBooks[i].setSecondAuthorName(line);

   getline(inData, line);
   myBooks[i].setThirdAuthorName(line);

   getline(inData, line);
   myBooks[i].setFourthAuthorName(line);
}
else
```

```
        {
        }
            getline(inData, line);
            myBooks[i].setTitle(line);

            getline(inData, line);
            myBooks[i].setISBM(line);

            getline(inData, line);
            myBooks[i].setPublisherName(line);

            getline(inData, line);
            myBooks[i].setYearPublished(line);

            getline(inData, line);
            myBooks[i].setPrice(line);

            getline(inData, line);
            myBooks[i].setCopies(line);
    }
}
//prints all the member variables for all the bookType objects
for (int i = 0; i < 5; i++)
{
    myBooks[i].print();
```

```
}
//searches the bookType array for a matching ISBM
for (int i = 0; i < 5; i++)
{
   if(myBooks[i].searchISBM("22-99521-453-1"))
   {
      found = true;
      break;
   }
   else
   {
      found = false;
   }
}
if(found == false)
{
   cout << "ISBM not Found" << endl;
}
else
{
   cout << "ISBM Found" << endl;
}
//searches the bookType array for a matching Author
for (int i = 0; i < 5; i++)
{
```

```
    if(myBooks[i].searchAuthor("Jimmy"))
    {
        found = true;
        break;
    }
    else
    {
        found = false;
    }
}
if(found == false)
{
    cout << "Author not Found" << endl;
}
else
{
    cout << "Author Found" << endl;
}
//searches the bookType array for a matching Publisher
for (int i = 0; i < 5; i++)
{
    if(myBooks[i].searchPublisher("ABC"))
    {
        found = true;
        break;
```

```
    }
    else
    {
        found = false;
    }
}
if(found == false)
{
    cout << "Publisher not Found" << endl;
}
else
{
    cout << "Publisher Found" << endl;
}
//searches the bookType array for a matching Title
for (int i = 0; i < 5; i++)
{
    if(myBooks[i].searchTitle("Harry Johnand The Magician"))
    {
        found = true;
        break;
    }
    else
    {
        found = false;
```

```
    }
}
if(found == false)
{
    cout << "Title not Found" << endl;
}
else
{
    cout << "Title Found" << endl;
}
//searches the bookType array for a matching Year Published
for (int i = 0; i < 5; i++)
{
    if(myBooks[i].searchYearPublished("2"))
    {
        found = true;
        break;
    }
    else
    {
        found = false;
    }
}
if(found == false)
{
```

```
    cout << "Year Published not Found" << endl;
}
else
{
    cout << "Year Published Found" << endl;
}
//searches the bookType array for a matching Price
for (int i = 0; i < 5; i++)
{
    if(myBooks[i].searchPrice("52.50"))
    {
        found = true;
        break;
    }
    else
    {
        found = false;
    }
}
if(found == false)
{
    cout << "Price not Found" << endl;
}
else
{
```

```
        cout << "Price Found" << endl;
    }
    return 0;
}
```

Output ->

```
The Book Title is: C++ Programing: From Problem Analysis to Program Design. The number of authors is 5. Thier name(s) are unknown. The publisher is ABC. The bo
ok was published in the year 2000. We have 20 copies of the book. Each copy is priced at 52.50. The ISBM is 5-17-525281-3.

The Book Title is: Fuzzy Discrete Structures. The number of authors is 1. Thier name(s) are Malik, D.S. The publisher is Physica-Verlag. The book was published
 in the year 2000. We have 10 copies of the book. Each copy is priced at 89.00. The ISBM is 3-7908-1335-4.

The Book Title is: Fuzzy Mathematic in Medicine. The number of authors is 2. Thier name(s) are Malik, Davender and Mordeson, John. The publisher is Physica-Ver
lag. The book was published in the year 2000. We have 10 copies of the book. Each copy is priced at 89.00. The ISBM is 3-7908-1325-7.

The Book Title is: Harry Johnand The Magician. The number of authors is 3. Thier name(s) are Mordeson, John, Malik, Davender and Cheng, Shih-Chung. The publish
er is McArthur A. Devine Books. The book was published in the year 1999. We have 10 copies of the book. Each copy is priced at 19.95. The ISBM is 0-239-23635-0
.

The Book Title is: Dynamic InterWeb Programming. The number of authors is 3. Thier name(s) are Goof, Goofy, Pluto, Peter and Head, Mark. The publisher is GNet.
 The book was published in the year 1998. We have 25 copies of the book. Each copy is priced at 39.99. The ISBM is 22-99521-453-1.

ISBM Found
Author not Found
Publisher Found
Title Found
Year Published not Found
Price Found
```

Reflection ->

Project Summary

This project was about designing and implementing a class file; it required one header file, two cpp files, and a text file. The header file is where you create your class and state it's member variables and functions. One of the cpp files is called the Implementation file which is where all the member functions are giving definitions. The second cpp file is called the Use file which is where the class is instantiated and tested.

Throughout this project, I learned that sometimes the plan you have for a program might not work perfectly. I wasted a decent amount of time trying to stick to my initial design when I could've easily just changed the way my program worked. I also learned how to use the getline function, which was very helpful for this program.

Challenges

The only problem I ran into during this project was figuring out how to read the information from a file correctly. At first, I tried to read individual strings and concatenate them together as needed, but it became difficult. I couldn't find a flag that would consistently work when trying to figure out whether to concatenate strings or not. After all that I decided to just remove all the spaces and read the data that way. It may have looked bad but at least the data got read correctly. Then after class one day, I heard about getline, so I added the spaces back into my text file and figured out how to implement getline into my code. Now with a little help from getline everything in my program works perfectly.

Lessons Learned

The major lesson I'm taking away from this assignment is that I have a lot to learn. I didn't know getline existed and I based my whole design document off of reading in single

values at a time. By trying to read in values one at a time I wasted a lot of effort trying to do something that could've been done easily a different way.