UTM
UNIVERSITI TEKNOLOGI MALAYSIA
Malaysia-Japan International Institute of Technology (MJIIT)

SEMESTER 3 2020/2021

SECJ 2013

Data Structure and Algorithm

Mini Project

NAME	Matric No.	Signature
1. Md Yusuf Bin Forkan	A19MJ0178	*yusuf*
2. Ruhul Quddus Tamim	A19MJ0182	*tamim*
3. Shafi Ahmed	A19MJ0184	*shafi*
Name of Lecturer	**SECTION**	**GROUP**
Dr Zaltul Alwani	15	1
Dr Nies Hui Wen	15	1

Introduction :

Book shop management system is the computerized application to automate all kinds of activity in a book shop. The main aim of this project is to manage the books in the book store. It can also manage the order processing, stock management and accounts management. This project will be very helpful for maintaining the record of sales and purchase. This Book shop Management System is a type of computer which provides the facility to the customer of the shop to purchase the books from the shop without any problem. This system keeps all the records of books, sales, inventory, stock and publication.

Functionalities :

- ⇨ Book Shop Management System provides the searching facilities based on various factors such as book stock, bills, purchase, sales, and books.
- ⇨ Book Shop Management System also tracks all the information such as details of books, stocks, bills, purchase sales and books.
- ⇨ It increases the efficiency of managing the customers.
- ⇨ It deals with monitoring the information of payments and transactions.
- ⇨ Editing, adding and updating of book records is improved which results in proper resource management of Book Shop Management System data.
- ⇨ It integrates all the records of books and its details including the up-to-date availability of the stock.

Figure 1.1 illustrates the whole functionality of the system in a visual way. In this system we have two users and one of them is the buyer(Customer) and the other is the seller (Admin & Staff). The rectangular box represents the whole system, and the circular shapes represent the functionalities in the system. Both the user will interact with different part of functions within the system. Customers can search for any books the need and will be able to know if the book is available or not. This process includes other three sub-processes. Customers can view if there is enough stock available and the price of the books. If the buyer’s needed quantity doesn’t match with the available stock, the system will notify the user about the insufficient stock. Buyers can purchase and see any books using this system. Another user of this system will be the seller. Seller will interact with different functions such as updating the price and stock of the books. Seller will manage the sales where he can get all the transaction details it helps him for the proper accounts management.

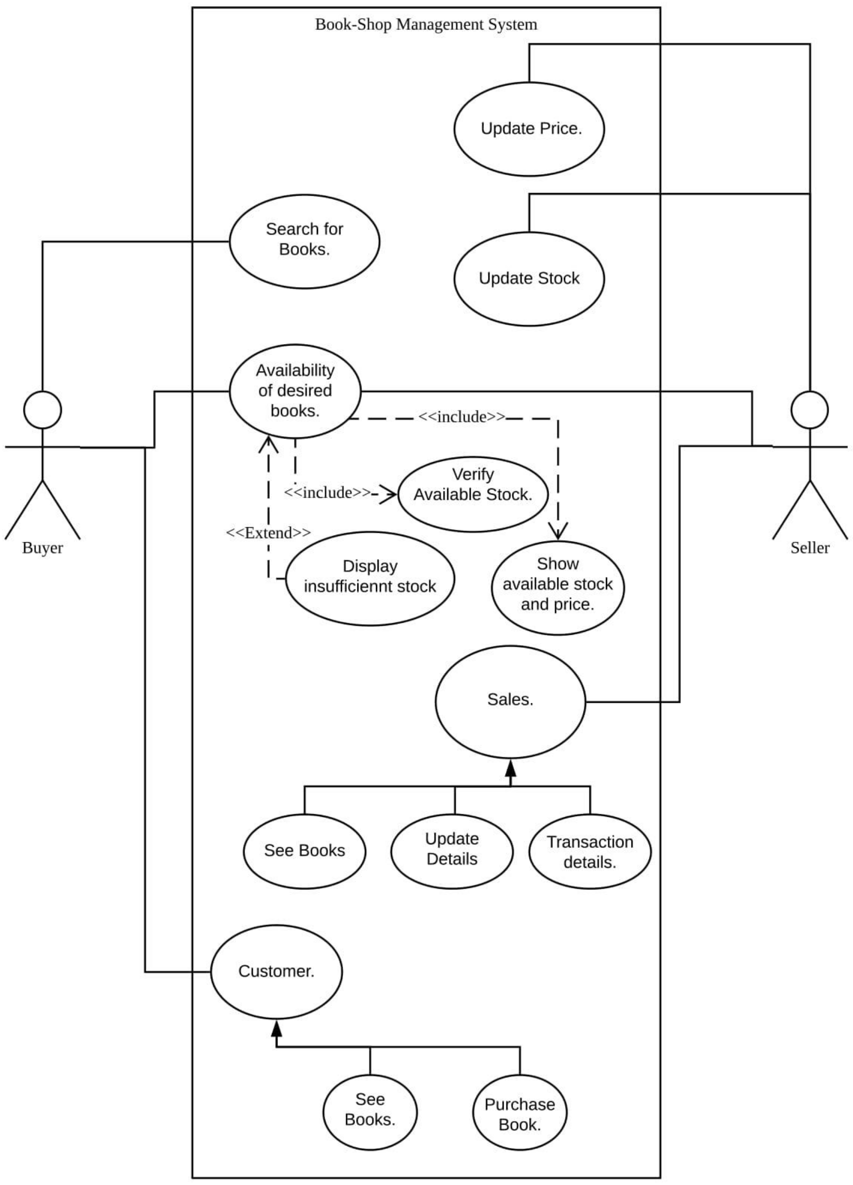


Figure 1.1: USE CASE Diagram for BOOK SHOP MANAGEMENT SYSTEM.

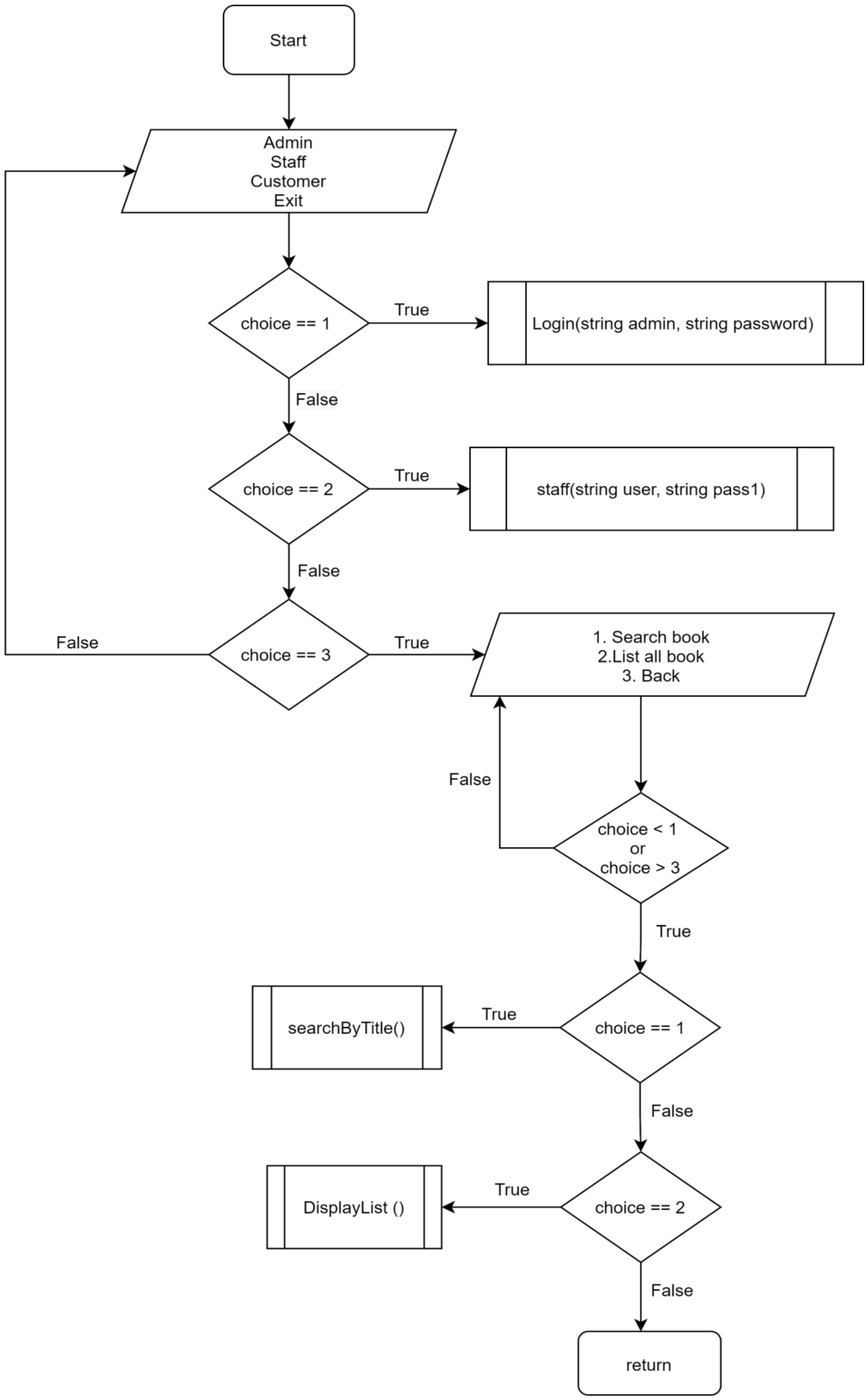
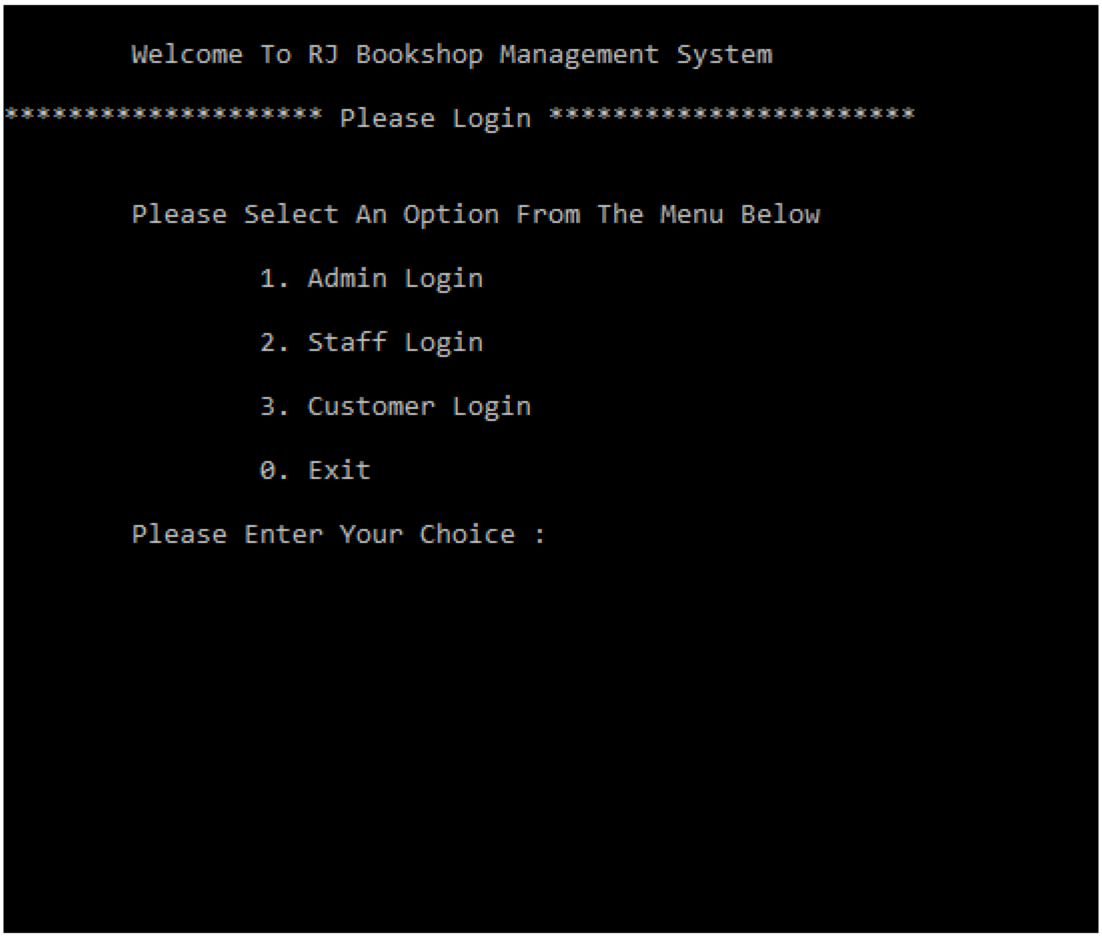


Figure 1.2: Flow Chart for BOOK SHOP MANAGEMENT SYSTEM.

Main Interface :



```
        Welcome To RJ Bookshop Management System

******************** Please Login ***********************


        Please Select An Option From The Menu Below

                1. Admin Login

                2. Staff Login

                3. Customer Login

                0. Exit

        Please Enter Your Choice :
```

Objective for Assignment 1

- ⇨ Create class book that contains all the attributes for books and make an array of objects that contains the list of books.
- ⇨ Use any sorting technique to sort the array on certain key

Synopsis for Assignment 1

1. Class Book

The source code for the class Book is as followed.

```
class book {

private:
    int bookCode;
    string author;
    string title;
    double price;
    string publisher;
    int stock;

    //Function To Update the existing book price
    void priceUpdate()
    {
        cout << "\nEnter The New Price: ";
        cin >> price;
        cout << "Price Successfully Updated!\n" << endl;
    }

    //Function to update the existing book stock
    void stockUpdate()
    {
        cout << "\nEnter The New Stock: ";
        cin >> stock;
        cout << "Stock Successfully Updated!\n" << endl;
    }

public:

    book() {};

    //class constructor with auguments
    book(int _bookCode, string _author, string _title, double _price, string _publisher, int _stock)
    {
        bookCode = _bookCode;
        author = _author;
        title = _title;
        price = _price;
        publisher = _publisher;
        stock = _stock;
    }

    //class constructor used for searching books
    book(string nAuthor, string nPublisher) {
        author = nAuthor;
        publisher = nPublisher;
    }
```

```
//class constructor used for searching bookcode
book(int nBookCode) {
    bookCode = nBookCode;
}

//Search Book Function
int search(book b)
{
    transform(author.begin(), author.end(), author.begin(), ::toupper);
    transform(b.author.begin(), b.author.end(), b.author.begin(), ::toupper);
    transform(publisher.begin(), publisher.end(), publisher.begin(), ::toupper);
    transform(b.publisher.begin(), b.publisher.end(), b.publisher.begin(), ::toupper);

    if (author == b.author && publisher == b.publisher)
        return 1;
    else
        return 0;
}

//To search bookcode
int searchCode(book b)
{
    if (bookCode == b.bookCode)
        return 1;
    else
        return 0;
}

//Number of books the customer want
void noOfCopies()
{
    int n;
    cout << "\nBook Details:\n" << endl;
    showDetail();
    cout << "Enter Required Number Of Copies: ";
    cin >> n;

    if (n > stock)
    {
        cout << "Required Copies Is Not In Stock, Sorry!\n" << endl;
        Transaction(0);
    }

    else
    {
        cout << "Total Cost of The Books: " << (n*price) << endl;//for the total price calculation
        stock = stock - n;
        cout << endl;
        cout << "Remaining Stock: " << stock << endl;//Define the remaining stock
        Transaction(1);
        cout << endl;
    }
}

//Function that will show all the details of each book
void showDetail()
{
    cout << "Book Code: " << bookCode << endl;
    cout << "Author: " << author << endl;
    cout << "Title: " << title << endl;
    cout << "Publisher: " << publisher << endl;
```

```
        cout << "Price: " << price << endl;
        cout << "Availabe Stocks: " << stock << endl;
        cout << endl << endl;
    }

    //Update existing books price and stock
    void update()
    {
        int x;
        cout << "\nSelect What To Update\n\n1. Price\n2. Stock\n" << endl;
        cout << "Input: ";
        cin >> x;

        if (x == 1)
            priceUpdate();
        else
            stockUpdate();
    }

    //Update file after changing book price and stock
    void fileUdate(int t)
    {
        ofstream newFile;

        if (t == 0) {
            newFile.open("list.txt");
            newFile << bookCode << "," << author << "," << title << "," << price << ".00" << "," << publisher << "," <<
stock;
        }

        if (t > 0) {
            newFile.open("list.txt", fstream::app);
            newFile << endl << bookCode << "," << author << "," << title << "," << price << ".00" << "," << publisher <<
"," << stock;
        }
    }

    //Sorts book price
    void priceSort(int bCount, int counter1, int ch3)
    {
        int sortBookCode[2000];
        string temp2;
        string temp3;
        double sortPrice[2000];
        string temp4;
        int temp5;

        int count = 0;
        string comma_string;

        ifstream classFile;
        classFile.open("list.txt");

        //Reading books data from file
        while (classFile) {

            classFile >> sortBookCode[count];//Getting bookcode value from file
            getline(classFile, comma_string, ',');
            getline(classFile, temp2, ',');
            getline(classFile, temp3, ',');
            classFile >> sortPrice[count];//Getting price value from file
        getline(classFile, comma_string, ',');
        getline(classFile, temp4, ',');
        classFile >> temp5;
        getline(classFile, comma_string, '\n');

        count++;
    }
    count--;

    pair<double, int> AB[2000];//Pairing two arrays

    //Getting value of price and bookcode to the paired array
    for (int i = 0; i < bCount + 1; i++)
    {
        AB[i].first = sortPrice[i];
        AB[i].second = sortBookCode[i];
    }

    sort(AB, AB + (bCount + 1));//Sorting from low to high

    if (ch3 == 2)
        reverse(AB, AB + (bCount + 1));//After sorting price reversed from high to low

    //Getting value of price and bookcode from the paired array after sorting
    for (int i = 0; i < bCount + 1; i++)
    {
        sortPrice[i] = AB[i].first;
        sortBookCode[i] = AB[i].second;
    }

    //Comparing price and bookcode together to show book details after sorting
    if (sortPrice[counter1] == price && sortBookCode[counter1] == bookCode)
    {
        cout << "Book Code: " << bookCode << endl;
        cout << "Author: " << author << endl;
        cout << "Title: " << title << endl;
        cout << "Publisher: " << publisher << endl;
        cout << "Price: " << price << endl;
        cout << "Availabe Stocks: " << stock << endl;
        cout << endl << endl;
    }
}
```

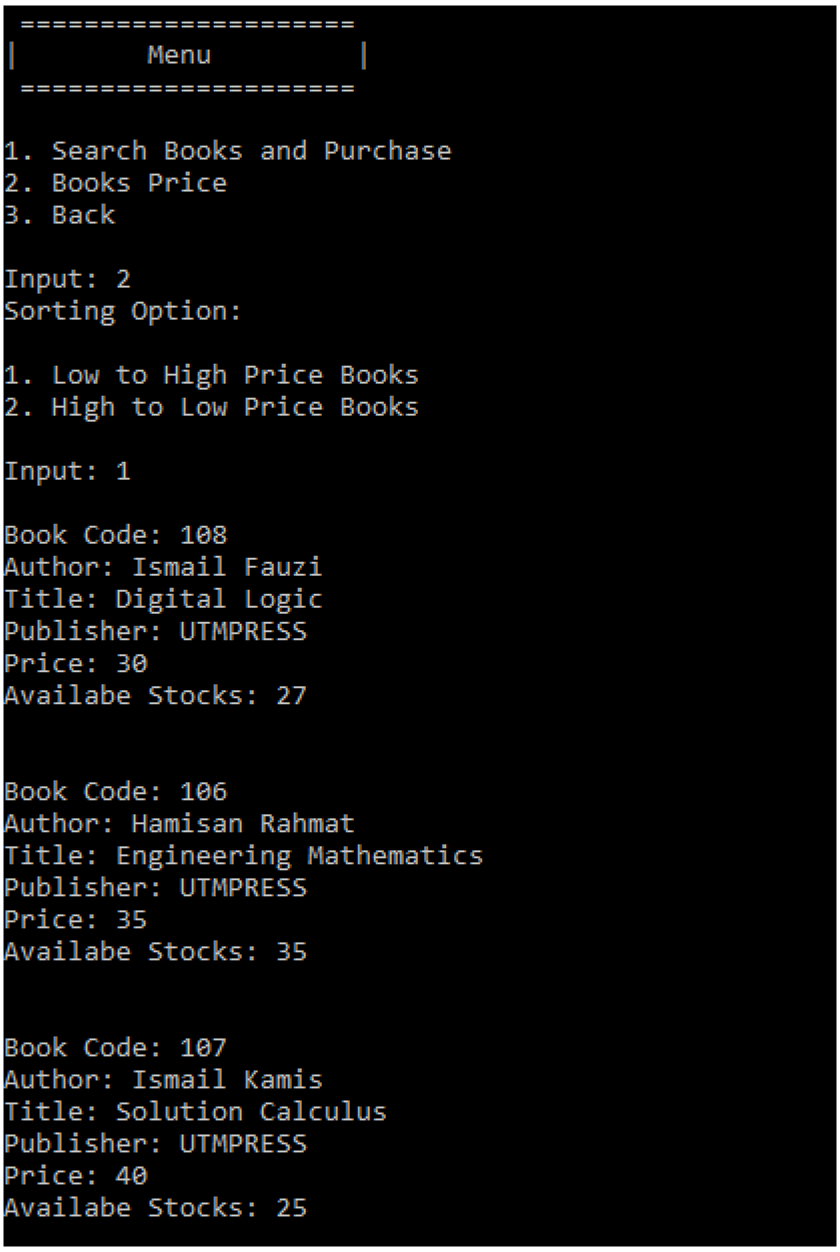
2. Sorting technique.

For sorting data from existing file, we implemented bubble sort method. Source codes are given as followed.

```
//Sorts book price
void priceSort(int bCount, int counter1, int ch3)
{
    int sortBookCode[2000];
    string temp2;
    string temp3;
    double sortPrice[2000];
    string temp4;
    int temp5;

    int count = 0;
    string comma_string;

    ifstream classFile;
    classFile.open("list.txt");

    //Reading books data from file
    while (classFile) {

        classFile >> sortBookCode[count];//Getting bookcode value from file
        getline(classFile, comma_string, ',');
        getline(classFile, temp2, ',');
        getline(classFile, temp3, ',');
        classFile >> sortPrice[count];//Getting price value from file
        getline(classFile, comma_string, ',');
        getline(classFile, temp4, ',');
        classFile >> temp5;
        getline(classFile, comma_string, '\n');

        count++;
    }
    count--;

    pair<double, int> AB[2000];//Pairing two arrays

    //Getting value of price and bookcode to the paired array
    for (int i = 0; i < bCount + 1; i++)
    {
        AB[i].first = sortPrice[i];
        AB[i].second = sortBookCode[i];
    }

    sort(AB, AB + (bCount + 1));//Sorting from low to high

    if (ch3 == 2)
        reverse(AB, AB + (bCount + 1));//After sorting price reversed from high to low

    //Getting value of price and bookcode from the paired array after sorting
    for (int i = 0; i < bCount + 1; i++)
    {
        sortPrice[i] = AB[i].first;
        sortBookCode[i] = AB[i].second;
    }

    //Comparing price and bookcode together to show book details after sorting
    if (sortPrice[counter1] == price && sortBookCode[counter1] == bookCode)
    {
        cout << "Book Code: " << bookCode << endl;
        cout << "Author: " << author << endl;
        cout << "Title: " << title << endl;
        cout << "Publisher: " << publisher << endl;
        cout << "Price: " << price << endl;
        cout << "Availabe Stocks: " << stock << endl;
        cout << endl << endl;
    }
}
```

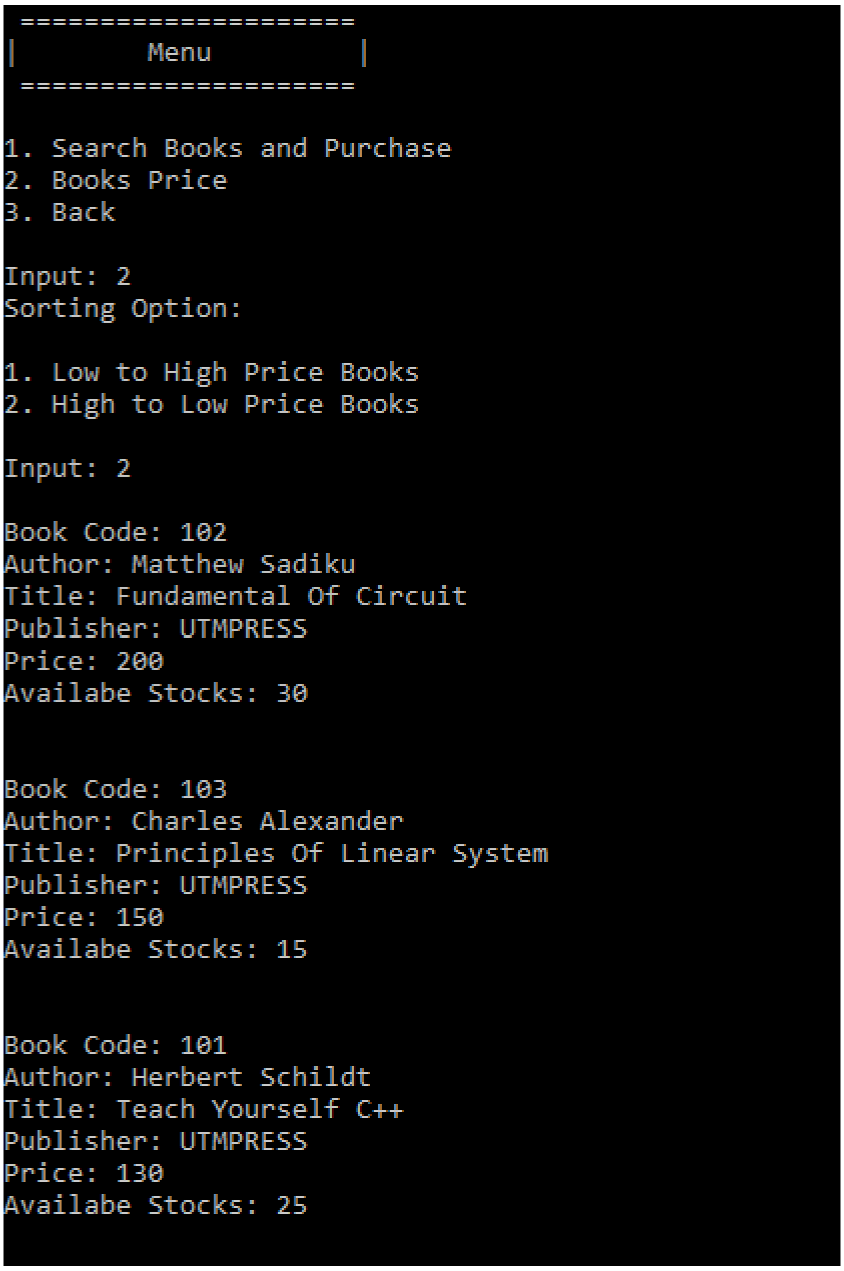
Output :

1. **Books price from Low to High :**



```
 =====================
|        Menu         |
 =====================

1. Search Books and Purchase
2. Books Price
3. Back

Input: 2
Sorting Option:

1. Low to High Price Books
2. High to Low Price Books

Input: 1

Book Code: 108
Author: Ismail Fauzi
Title: Digital Logic
Publisher: UTMPRESS
Price: 30
Availabe Stocks: 27


Book Code: 106
Author: Hamisan Rahmat
Title: Engineering Mathematics
Publisher: UTMPRESS
Price: 35
Availabe Stocks: 35


Book Code: 107
Author: Ismail Kamis
Title: Solution Calculus
Publisher: UTMPRESS
Price: 40
Availabe Stocks: 25
```

2. **Books price from High to Low :**



```
 =====================
|        Menu         |
 =====================

1. Search Books and Purchase
2. Books Price
3. Back

Input: 2
Sorting Option:

1. Low to High Price Books
2. High to Low Price Books

Input: 2

Book Code: 102
Author: Matthew Sadiku
Title: Fundamental Of Circuit
Publisher: UTMPRESS
Price: 200
Availabe Stocks: 30


Book Code: 103
Author: Charles Alexander
Title: Principles Of Linear System
Publisher: UTMPRESS
Price: 150
Availabe Stocks: 15


Book Code: 101
Author: Herbert Schildt
Title: Teach Yourself C++
Publisher: UTMPRESS
Price: 130
Availabe Stocks: 25
```

Objective for Assignment 2

- ⇨ Perform searching based on certain criteria. In our assignment we use Bubble sorting & Linear searching technique.
- ⇨ Implement linked list instead of array of data. Linked list must be containing the nodes for every instance created from class.
- ⇨ Implement linked list operations such as add, delete and search.

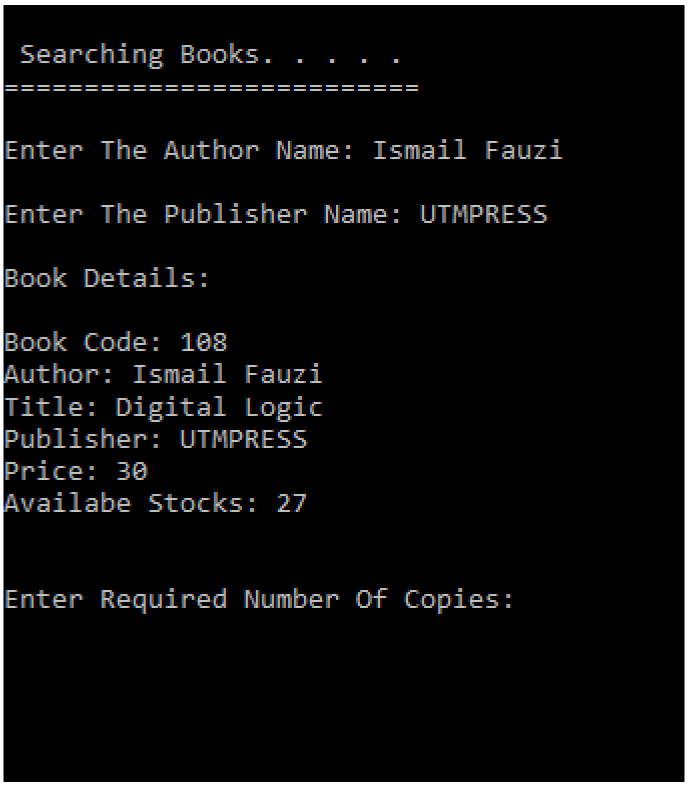
Synopsis for Assignment 2

1. **Searching**
2. **Using Linked list**
3. **Implementing Operations**

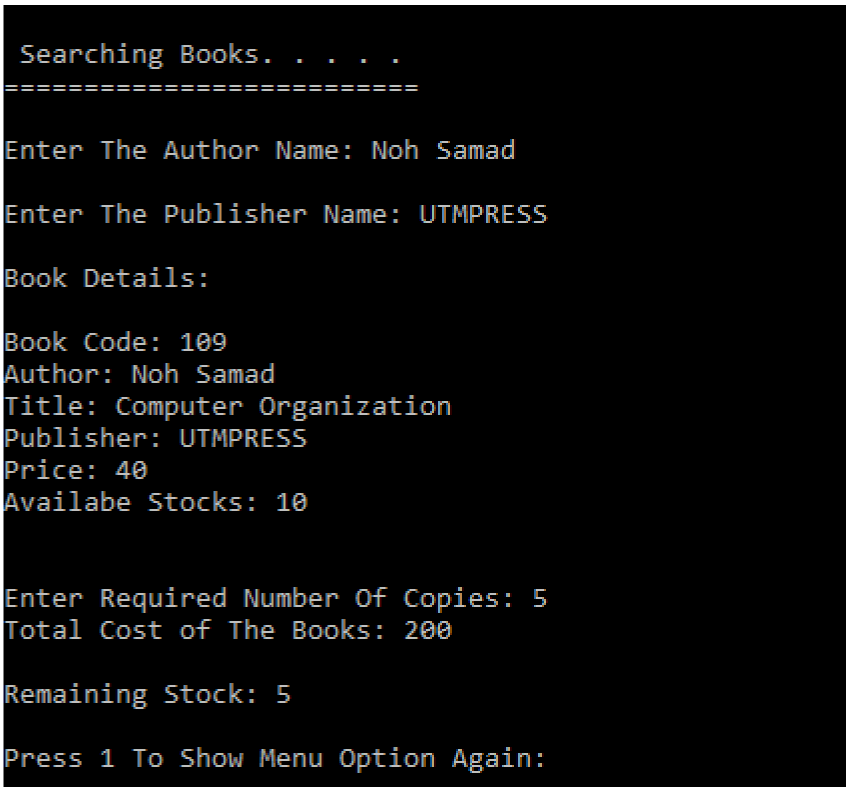
For searching books from an existing data file, we used the linear search technique, and the codes are given below.

```
//class constructor used for searching books
book(string nAuthor, string nPublisher) {
    author = nAuthor;
    publisher = nPublisher;
}

//class constructor used for searching bookcode
book(int nBookCode) {
    bookCode = nBookCode;
}

//Search Book Function
int search(book b)
{
    transform(author.begin(), author.end(), author.begin(), ::toupper);
    transform(b.author.begin(), b.author.end(), b.author.begin(), ::toupper);
    transform(publisher.begin(), publisher.end(), publisher.begin(), ::toupper);
    transform(b.publisher.begin(), b.publisher.end(), b.publisher.begin(), ::toupper);

    if (author == b.author && publisher == b.publisher)
        return 1;
    else
        return 0;
}
//To search bookcode
int searchCode(book b)
{
    if (bookCode == b.bookCode)
        return 1;
    else
        return 0;
}
```

Output :



```
 Searching Books. . . . .
==========================

Enter The Author Name: Ismail Fauzi

Enter The Publisher Name: UTMPRESS

Book Details:

Book Code: 108
Author: Ismail Fauzi
Title: Digital Logic
Publisher: UTMPRESS
Price: 30
Availabe Stocks: 27


Enter Required Number Of Copies:
```

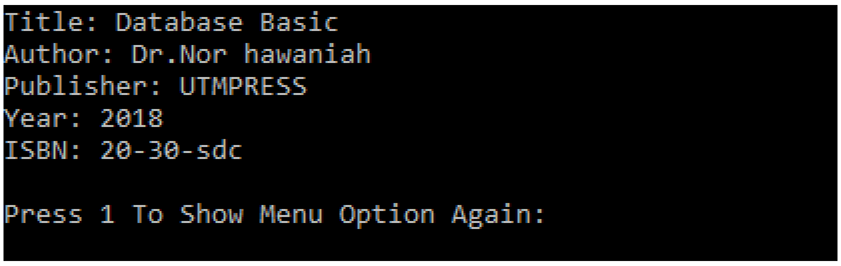


```
 Searching Books. . . . .
==========================

Enter The Author Name: Noh Samad

Enter The Publisher Name: UTMPRESS

Book Details:

Book Code: 109
Author: Noh Samad
Title: Computer Organization
Publisher: UTMPRESS
Price: 40
Availabe Stocks: 10


Enter Required Number Of Copies: 5
Total Cost of The Books: 200

Remaining Stock: 5

Press 1 To Show Menu Option Again:
```

⇨ Adding a new book.

```
//Function for adding a new book
void insertNewBook(bookPtr &head)
{
   string tit, autr, publs, isb;
   int yer;

   cout << "Title: ";
   cin.ignore(500, '\n');
   getline(cin, tit, '\n');
   cout << "Author: ";
   getline(cin, autr, '\n');
   cout << "Publisher: ";
   getline(cin, publs, '\n');
   cout << "Year: ";
   getline(cin, yer, '\n');
   cout << "ISBN: ";
   getline(cin, isb, '\n');

   head = new book(tit, autr, publs, yer, isb, head);
}
```

Output :

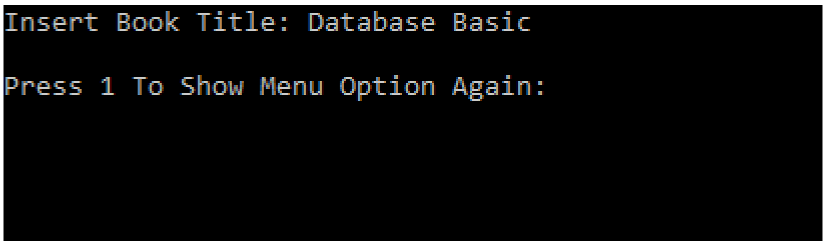


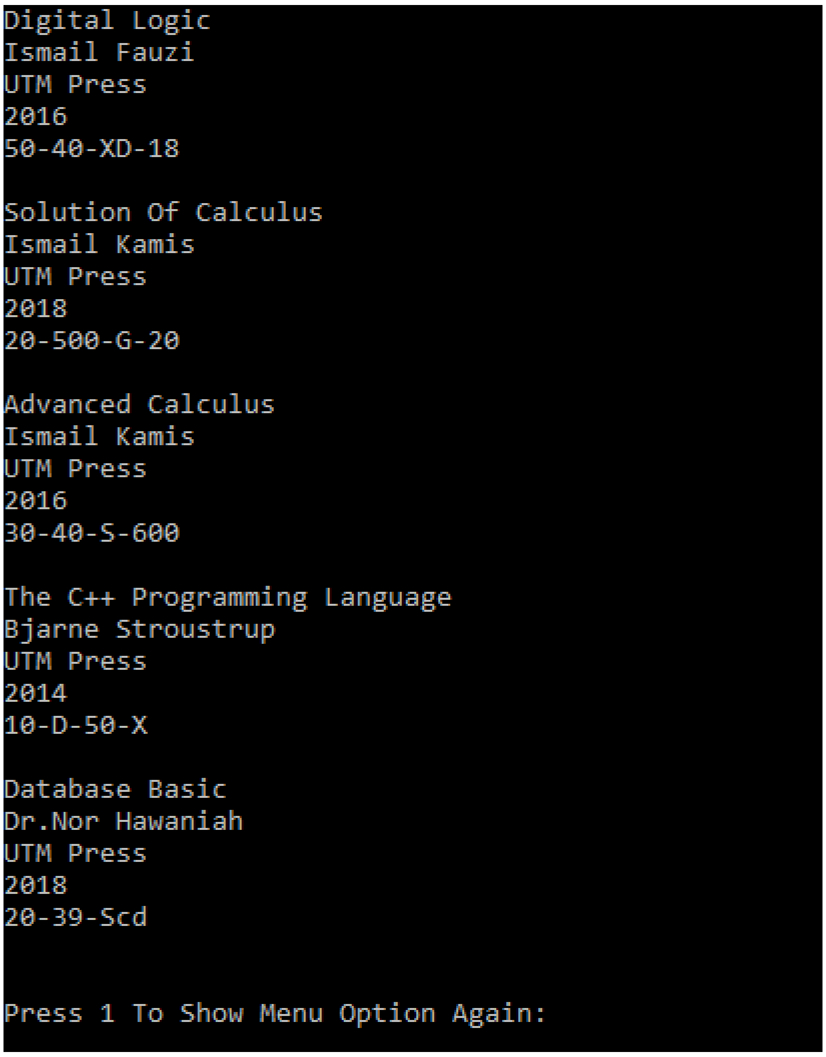
```
Title: Database Basic
Author: Dr.Nor hawaniah
Publisher: UTMPRESS
Year: 2018
ISBN: 20-30-sdc

Press 1 To Show Menu Option Again:
```

⇨ Deleting books from the list using Title.

```
//Function For deleting a Book Based on title
void deleteByTitle(bookPtr &head)
{
   string tit;

   cout << "Insert Book Title: ";
   cin.ignore(500, '\n');
   getline(cin, tit, '\n');
   bookPtr p = locateByNode(head, tit);
   if (p == NULL)
      cout << "\nDeletion cannot be done.\n\n";
   else if (head == p)
      head = p->next;
   else
   {
      bookPtr q = head;
      while (q->next != p)
         q = q->next;
      q->next = p->next;
   }
   delete p;
}
```

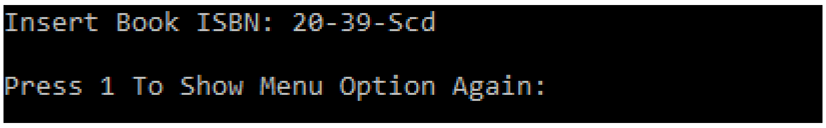
Output :


```
Insert Book Title: Database Basic

Press 1 To Show Menu Option Again:
```

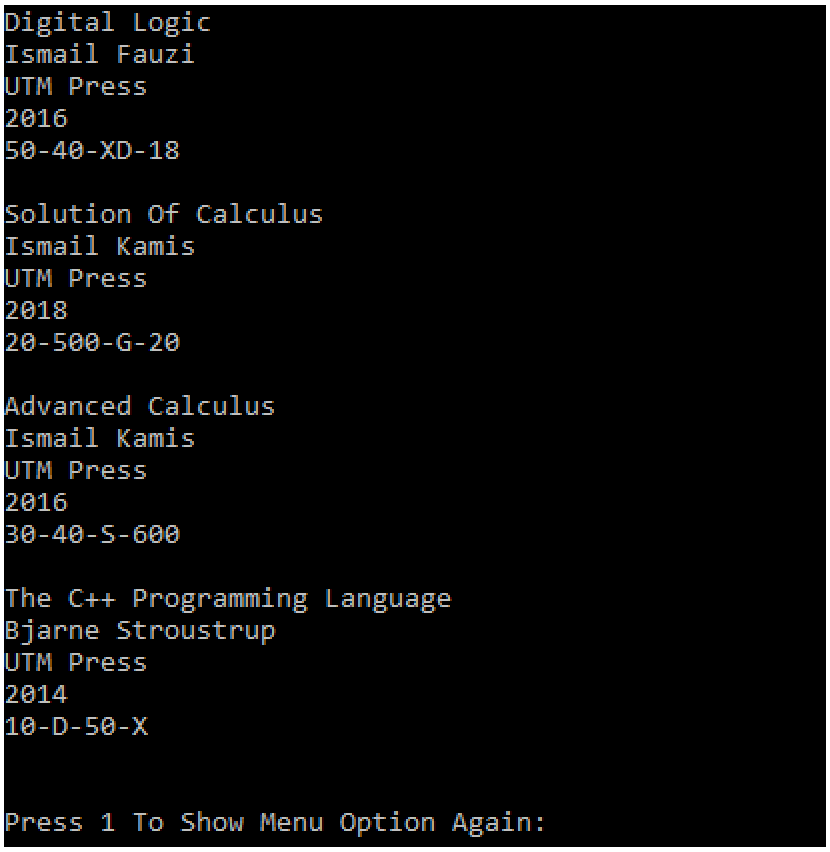

```
Digital Logic
Ismail Fauzi
UTM Press
2016
50-40-XD-18

Solution Of Calculus
Ismail Kamis
UTM Press
2018
20-500-G-20

Advanced Calculus
Ismail Kamis
UTM Press
2016
30-40-S-600

The C++ Programming Language
Bjarne Stroustrup
UTM Press
2014
10-D-50-X

Database Basic
Dr.Nor Hawaniah
UTM Press
2018
20-39-Scd


Press 1 To Show Menu Option Again:
```

⇨ Deleting books from the list using ISBN

```
//Function for deleting a book by ISBN
void deleteByIsbn(bookPtr &head)
{
    string isb;

    cout << "Insert Book ISBN: ";
    cin.ignore(500, '\n');
    getline(cin, isb, '\n');

    bookPtr p = locateNodeByIsbn(head, isb);

    if (p == NULL)
        cout << "\nDeletion cannot be done.\n\n";
    else if (head == p)
        head = p->next;
    else
    {
        bookPtr q = head;
        while (q->next != p)
            q = q->next;
        q->next = p->next;
    }
    delete p;
}
```



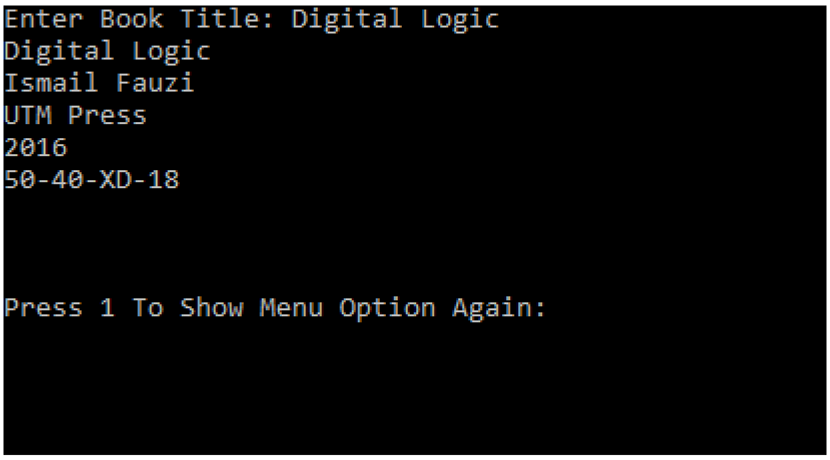
```
Insert Book ISBN: 20-39-Scd

Press 1 To Show Menu Option Again:
```



```
Digital Logic
Ismail Fauzi
UTM Press
2016
50-40-XD-18

Solution Of Calculus
Ismail Kamis
UTM Press
2018
20-500-G-20

Advanced Calculus
Ismail Kamis
UTM Press
2016
30-40-S-600

The C++ Programming Language
Bjarne Stroustrup
UTM Press
2014
10-D-50-X


Press 1 To Show Menu Option Again:
```

⇨ Searching books by title.

```
//Function for searching a book by title
void searchByTitle(bookPtr temp)
{
    string tit;

    cout << "Enter Book Title: ";
    cin.ignore(500, '\n');
    getline(cin, tit, '\n');


    while (temp != NULL)
    {
        if (tit == temp->title)
        {
            cout << temp->title << "\n";
            cout << temp->author << "\n";
            cout << temp->publisher << "\n";
            cout << temp->pubYear << "\n";
            cout << temp->isbn << "\n\n";
        }
        temp = temp->next;
    }
    cout << "\n";
}
```



```
Enter Book Title: Digital Logic
Digital Logic
Ismail Fauzi
UTM Press
2016
50-40-XD-18



Press 1 To Show Menu Option Again:
```

Objective for Mini Project

⇨ Implement Queue data structure using Stacks. Queue uses the FIFO(First-In-First-Out) concept.

Synopsis of Mini Project

Source codes for implementing Queue data structure.

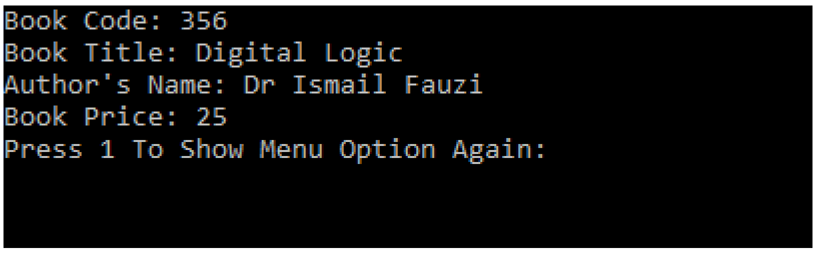
```
class Node {
public:
    int ID;
    string name;
    string publisherName;
    double price;      // data
    Node* next;

    Node(int _ID, string _name, string _publisherName, double _price) {

        _ID = ID;
        _name = name;
        _publisherName = publisherName;
        _price = price;
    }         // pointer to next node
};

class queue {
public:
    Node *backPtr = NULL, *frontPtr = NULL;

    void enQueue(int _ID, string _name, string _publisherName, double _price);
    void deQueue();
    void DisplayList();
    void searchByTitle();
    void printListByAuthor();
    void printBookList();
    void bookManagement(int input);
};

void queue::enQueue(int _ID, string _name, string _publisherName, double _price) {

    Node *newPtr = new         Node(_ID, _name, _publisherName, _price);
    newPtr->ID = _ID;
    newPtr->name = _name;
    newPtr->price = _price;
    newPtr->publisherName = _publisherName;

    if (frontPtr == NULL) {

        newPtr->next = NULL;
        frontPtr = backPtr = newPtr;
    }

    else {
        newPtr->next = NULL;
        backPtr->next = newPtr;
        backPtr = newPtr;
    }
}

void queue::deQueue() {
    if (frontPtr == NULL) cout << "Queue is empty";
    else {
        Node *temp = frontPtr;
        frontPtr = frontPtr->next;
        temp->next = NULL;
        delete temp;
    }

}
void queue::DisplayList() {

    if (frontPtr == NULL)
        cout << "Queue is Empty" << endl;
    else
    {
        //cout<<"Book ID\t\t"<<"Book Name \t\t\t"<<"Author Name\t\t"<<"Price"<<endl;
        Node *temp = frontPtr;
        while (temp->next != NULL) {

            cout <<"Book ID: "<< temp->ID<<endl<<"Book Name: "<<temp->name<<endl<<"Author Name: "<< temp-
>publisherName<<endl<<"Price: "<<temp->price<<endl<<endl;

            temp = temp->next;
        }

        cout <<"Book ID: "<< temp->ID<<endl<<"Book Name: "<<temp->name<<endl<<"Author Name: "<< temp-
>publisherName<<endl<<"Price: "<<temp->price<<endl<<endl;
    }
}
void queue::searchByTitle() {
    string _title;

    cout << "Enter the title: ";
    cin >> _title;
    //_title = " " + _title;
    //cout<<"Book ID\t\t"<<"Book Name \t\t"<<"Author Name\t\t"<<"Price"<<endl;
    Node *currNode = frontPtr;
    while (currNode) {


        if (_title == currNode->name) {

            cout <<"Book ID: "<< currNode->ID<<endl<<"Book Name: "<<currNode->name<<endl<<"Author Name: "<<
currNode->publisherName<<endl<<"Price: "<<currNode->price<<endl<<endl;

        }
        currNode = currNode->next;

    }

}
```

```
void queue::bookManagement(int input)
{
    int choice, x;
    int _id;
    string _name;
    string _publisherName;
    double _price;
    int _id2;
    string _name2;
    string _publisherName2;
    double _price2;



    while (1) {

        if (input == 1) {
            //Options for admins
            cout << "Select your option\n\n";
            cout << "1. Add a new book to the list\n";
            cout << "2. De-Queue a book from the list\n";
            cout << "3. Search for a book by Title\n";
            cout << "4. List all books\n";

            cout << "5. Back\n\n";

            do {
                cout << "Enter your choice => ";

                cin >> choice;

            } while ((choice < 1 || choice > 5) && cout << "Invalid Input!\n");

            switch (choice)
            {
                case 1:
                    system("cls");
                    cout<<"ID: ";
                    cin>>_id;
                    cout<<"name: ";
                    cin>>_name;
                    cout<<"Author Name: ";
                    cin>>_publisherName;
                    cout<<"Price: ";
                    cin>>_price;

                    enQueue(_id,_name,_publisherName,_price);
                    break;
                case 2:
                    system("cls");
                    deQueue();
                    break;
                case 3:
                    system("cls");
                    searchByTitle();
                    break;

                case 4:
                    system("cls");
                    DisplayList();
                    break;
                case 5:
                    return;
            }

        }

        else if (input == 2) {
            //Options for staffs
            cout << "Select your option\n\n";
            cout << "1. Add a new book to the list\n";
            cout << "2. DeQueue a book \n";
            cout << "3. Back\n\n";
            do {
                cout << "Enter your choice => ";

                cin >> choice;

            } while ((choice < 1 || choice > 3) && cout << "Invalid Input!\n");


            switch (choice)
            {
                case 1:
                    system("cls");
                    system("cls");
                    cout<<"ID: ";
                    cin>>_id2;
                    cout<<"name: ";
                    cin>>_name2;
                    cout<<"Author Name: ";
                    cin>>_publisherName2;
                    cout<<"Price: ";
                    cin>>_price2;

                    enQueue(_id2,_name2,_publisherName2,_price2);
                    break;
                case 2:
                    system("cls");
                    deQueue();
                    break;

                case 3:
                    return;
            }
        }

        else if (input == 3) {
            //Options for customers
            cout << "Select your option\n\n";
            cout << "1. Search for a book by Title\n";
            cout << "2. List all books\n";
            cout << "3. Back\n\n";
            do {
                cout << "Enter your choice => ";

                cin >> choice;

            } while ((choice < 1 || choice > 3) && cout << "Invalid Input!\n");
```

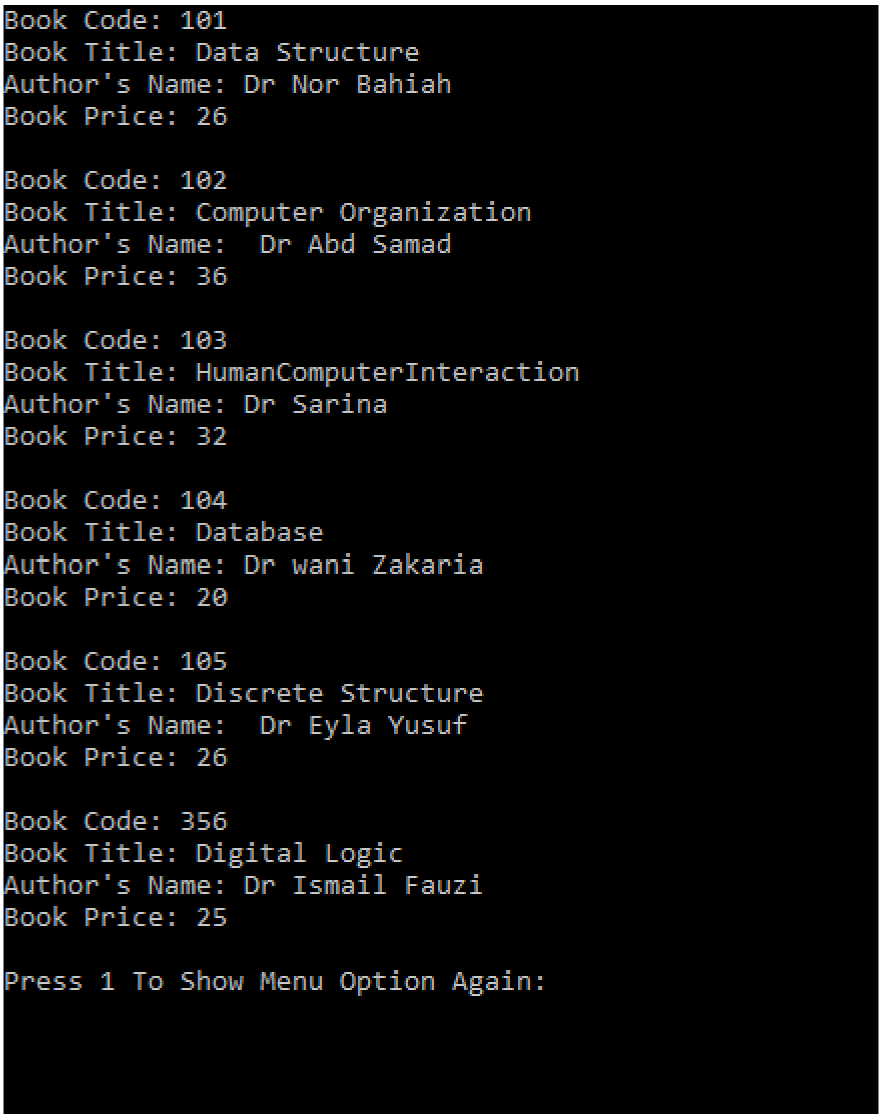
```
            switch (choice)
            {
                case 1:
                    system("cls");
                    searchByTitle();
                    break;
                case 2:
                    system("cls");
                    DisplayList();
                    break;
                case 3:
                    return;
            }
        }


        do {
            cout << "Press 1 To Show Menu Option Again: ";//For going back again
            cin >> x;

            if (x == 1)
                system("cls");

            else
                cout << "Invalid Input!\n" << endl;

        } while (x != 1);
    }
}
```

Program Outputs :

1. EnQueue a Book :

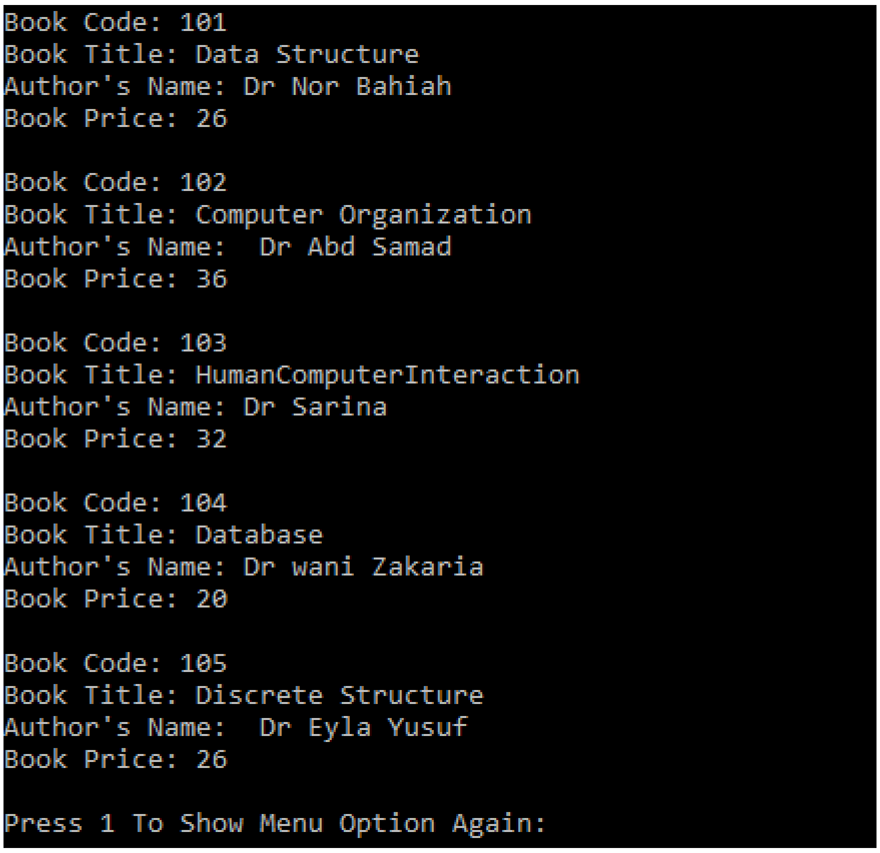


```
Book Code: 356
Book Title: Digital Logic
Author's Name: Dr Ismail Fauzi
Book Price: 25
Press 1 To Show Menu Option Again:
```

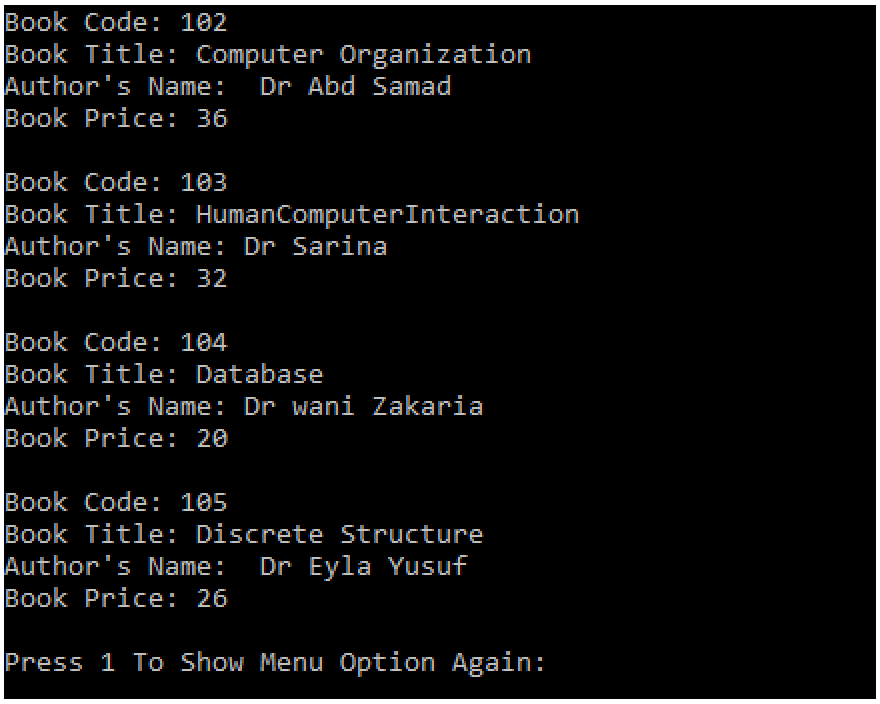


```
Book Code: 101
Book Title: Data Structure
Author's Name: Dr Nor Bahiah
Book Price: 26

Book Code: 102
Book Title: Computer Organization
Author's Name:  Dr Abd Samad
Book Price: 36

Book Code: 103
Book Title: HumanComputerInteraction
Author's Name: Dr Sarina
Book Price: 32

Book Code: 104
Book Title: Database
Author's Name: Dr wani Zakaria
Book Price: 20

Book Code: 105
Book Title: Discrete Structure
Author's Name:  Dr Eyla Yusuf
Book Price: 26

Book Code: 356
Book Title: Digital Logic
Author's Name: Dr Ismail Fauzi
Book Price: 25

Press 1 To Show Menu Option Again:
```

2. DeQueue a Book :

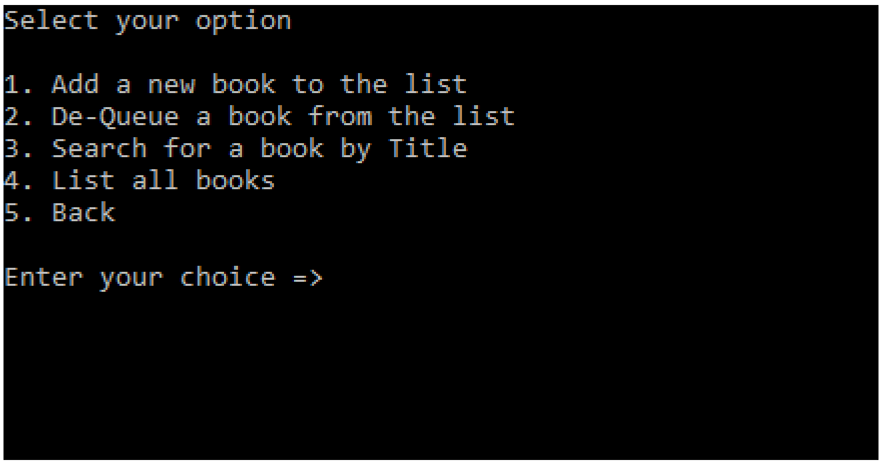


```
Book Code: 101
Book Title: Data Structure
Author's Name: Dr Nor Bahiah
Book Price: 26

Book Code: 102
Book Title: Computer Organization
Author's Name:  Dr Abd Samad
Book Price: 36

Book Code: 103
Book Title: HumanComputerInteraction
Author's Name: Dr Sarina
Book Price: 32

Book Code: 104
Book Title: Database
Author's Name: Dr wani Zakaria
Book Price: 20

Book Code: 105
Book Title: Discrete Structure
Author's Name:  Dr Eyla Yusuf
Book Price: 26

Press 1 To Show Menu Option Again:
```

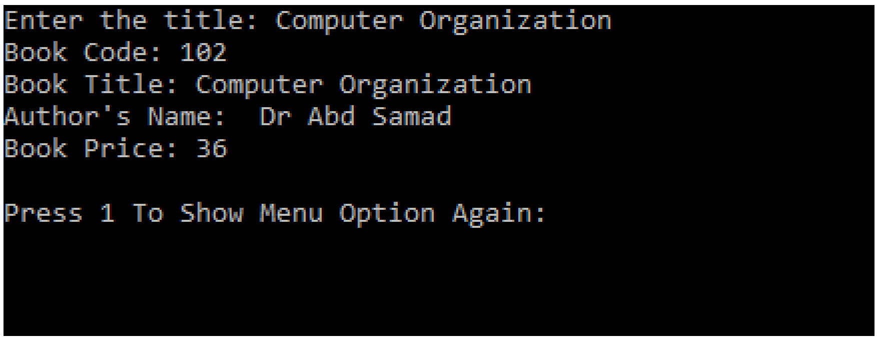


```
Book Code: 102
Book Title: Computer Organization
Author's Name:  Dr Abd Samad
Book Price: 36

Book Code: 103
Book Title: HumanComputerInteraction
Author's Name: Dr Sarina
Book Price: 32

Book Code: 104
Book Title: Database
Author's Name: Dr wani Zakaria
Book Price: 20

Book Code: 105
Book Title: Discrete Structure
Author's Name:  Dr Eyla Yusuf
Book Price: 26

Press 1 To Show Menu Option Again:
```

3. Search a book by Title :



```
Select your option

1. Add a new book to the list
2. De-Queue a book from the list
3. Search for a book by Title
4. List all books
5. Back

Enter your choice =>
```



```
Enter the title: Computer Organization
Book Code: 102
Book Title: Computer Organization
Author's Name:  Dr Abd Samad
Book Price: 36

Press 1 To Show Menu Option Again:
```

Conclusion : We have learnt so many things from the Data Structure & Algorithm course. Previously we only used the structured programming concept. But now we can also apply some concepts like sorting, searching, linked list, stack, Queue which can make our program more efficient. While doing the assignment 1, assignment 2 & mini project we have tried our best to implement those concepts in our codes.

Thanks to our lecturer for teaching us so well. We are grateful!