

Principles of Programming Language (CS302)

Practical Exam

U19CS012

1.) A **book shop** maintains the inventory of books that are being sold at the shop. The list includes details such as author, title, price, publisher and stock position.

Whenever a customer wants a book, the sales person inputs **the title and author** and the system searches the list and displays whether it is available or not.

- ✓ If it is not, an appropriate message is displayed.
- ✓ If it is, then the system displays the **book details** and requests for the **number of copies** required.
- ✓ If the requested copies are **available**, the total cost of the requested copies is displayed; otherwise the message "Required copies not in stock" is displayed.

Design a system using a class called **books** with suitable member functions and constructors. Use **new** operators in constructors to allocate memory space required. Implement **C++ program** for the system.

Improve the system design to incorporate the following features:

- The **price of the books should be updated** as and when required. Use a member function to implement this.
- The **stock value of each book should be automatically** updated as soon as a transaction is completed.
- The number of **successful and unsuccessful transactions** should be recorded for the purpose of statistical analysis. Use **static data members** to keep count of transactions.
- Also demonstrate the use of pointers to access the members.

Code

```
#include <bits/stdc++.h>
using namespace std;

// For Statistical Analysis of Transactions
static int success = 0;
static int failure = 0;

// Book Class
class book
```

```

{
public:
    string author, title, publisher;
    int price, stock;

    // Book Constructor
    book() {}
    book(string author, string title, string publisher, int price, int stock)
    {
        this->author = author;
        this->title = title;
        this->publisher = publisher;
        this->price = price;
        this->stock = stock;
    }

    // To Check if Stock is Available or Not
    bool is_available()
    {
        return stock > 0;
    }

    // Function to Check if Match is Found or Not
    bool match(string title, string author)
    {
        return this->title == title and this->author == author;
    }

    // Returns the Price if Stock is Available
    float available(int copies)
    {
        if (stock >= copies)
        {
            stock -= copies;
            return (copies * price);
        }
        else
        {
            return -1;
        }
    }

    // F(x) to Update the Price
    void update_price(int price)
    {
        this->price = price;
    }

    // F(x) to Update the Stock
    void update_stock(int stock)

```

```

{
    this->stock += stock;
}
};

// Inventory Class
class inventory
{
    vector<book *> books;

public:
    // F(x) to add Book
    void add_book(book *b)
    {
        books.push_back(b);
        success++;
    }

    // F(x) to Search for Particular Book [Linear Search]
    bool search_book(string title, string author)
    {
        for (int i = 0; i < books.size(); i++)
        {
            if (books[i]->match(title, author))
            {
                return true;
            }
        }
        return false;
    }

    // F(x) to issue Book
    bool issue_book(string title, string author, int copies)
    {
        for (int i = 0; i < books.size(); i++)
        {
            if (books[i]->match(title, author))
            {
                if (books[i]->is_available())
                {
                    float cost = books[i]->available(copies);
                    if (cost != -1)
                    {
                        cout << "Book issued successfully. Cost : " << cost << endl;
                        success++;
                        return true;
                    }
                }
                else
                {
                    cout << "Not Enough Copies Available. No Book Issued!" << endl;

```

```

        failure++;
        return false;
    }
}
else
{
    cout << "Book Not Available." << endl;
    failure++;
    return false;
}
}
failure++;
cout << "Book Not Found." << endl;
return false;
}

// F(x) to Update the Price
void update_price(string title, string author, int price)
{
    for (int i = 0; i < books.size(); i++)
    {
        if (books[i]->match(title, author))
        {
            books[i]->update_price(price);
            cout << "Price Updated." << endl;
            success++;
            return;
        }
    }
    failure++;
    cout << "Book Not Found." << endl;
}

// F(x) to Update the Stock of Book
void update_stock(string title, string author, int stock)
{
    for (int i = 0; i < books.size(); i++)
    {
        if (books[i]->match(title, author))
        {
            books[i]->update_stock(stock);
            cout << "Stock Updated." << endl;
            success++;
            return;
        }
    }
    failure++;
    cout << "Book Not Found." << endl;
}

```

```

};

// F(x) to Print the Menu Options
void menu()
{
    cout << "~~~~~\n";
    cout << "1 -> Add Book" << endl;
    cout << "2 -> Search Book" << endl;
    cout << "3 -> Issue Book" << endl;
    cout << "4 -> Update Book Price" << endl;
    cout << "5 -> Update Book Stock" << endl;
    cout << "6 -> Statistical Analysis" << endl;
    cout << "7 -> Exit" << endl;
    cout << "~~~~~\n";
}

int main()
{
    inventory store;
    book tmp_book;
    string author, title, publisher;
    int price, stock, copies;
    int cost, choice = 0;

    while (choice != 7)
    {
        menu();
        cout << "Enter your Choice : ";
        cin >> choice;

        switch (choice)
        {
            case 1:
            {
                cout << "Enter Author : ";
                cin >> author;
                cout << "Enter Title : ";
                cin >> title;
                cout << "Enter Publisher : ";
                cin >> publisher;
                cout << "Enter Price : ";
                cin >> price;
                cout << "Enter Stock : ";
                cin >> stock;

                tmp_book = book(author, title, publisher, price, stock);
                store.add_book(new book(tmp_book));

                cout << "Book Added Successfully." << endl;
                break;
            }
        }
    }
}

```

```

}
case 2:
{
    cout << "Enter Author : ";
    cin >> author;
    cout << "Enter Title : ";
    cin >> title;
    if (store.search_book(title, author))
    {
        cout << "Book Found." << endl;
    }
    else
    {
        cout << "Book Not Found." << endl;
    }
    break;
}
case 3:
{
    cout << "Enter Author : ";
    cin >> author;
    cout << "Enter Title : ";
    cin >> title;
    cout << "Enter No. of Copies: ";
    cin >> copies;
    store.issue_book(title, author, copies);
    break;
}
case 4:
{
    cout << "Enter Author : ";
    cin >> author;
    cout << "Enter Title : ";
    cin >> title;
    cout << "Enter New Price : ";
    cin >> price;
    store.update_price(title, author, price);
    break;
}
case 5:
{
    cout << "Enter Author : ";
    cin >> author;
    cout << "Enter Title : ";
    cin >> title;
    cout << "Enter Stock to be Added : ";
    cin >> stock;
    store.update_stock(title, author, stock);
    break;
}

```

```

    case 6:
    {
        cout << "Successful Transactions : " << success << endl;
        cout << "Failure Transactions      : " << failure << endl;
        break;
    }
    case 7:
    {
        cout << "Thank You for Visiting Our Book Shop!" << endl;
        break;
    }
    default:
    {
        cout << "Invalid Choice Entered." << endl;
        break;
    }
}
cout << "~~~~~\n";
return 0;
}

```

Output

Task 1: Adding a Book

```

~~~~~
1. Add Book
2. Search Book
3. Issue Book
4. Update Book Price
5. Update Book Stock
6. Statistical Analysis
7. Exit
~~~~~
Enter your Choice : 1
Enter Author : Morris
Enter Title : Electronics
Enter Publisher : Pearson
Enter Price : 700
Enter Stock : 10
Book Added Successfully.
~~~~~
1. Add Book
2. Search Book
3. Issue Book
4. Update Book Price
5. Update Book Stock
6. Statistical Analysis
7. Exit
~~~~~
Enter your Choice : 1
Enter Author : Balaguruswamy
Enter Title : OOP
Enter Publisher : McGrawHill
Enter Price : 500
Enter Stock : 5
Book Added Successfully.
~~~~~

```

Task 2: Searching a Book

```
~~~~~
1. Add Book
2. Search Book
3. Issue Book
4. Update Book Price
5. Update Book Stock
6. Statistical Analysis
7. Exit
~~~~~
Enter your Choice : 2
Enter Author : Morris
Enter Title : Electronics
Book Found.
~~~~~
1. Add Book
2. Search Book
3. Issue Book
4. Update Book Price
5. Update Book Stock
6. Statistical Analysis
7. Exit
~~~~~
Enter your Choice : 2
Enter Author : Bhagya
Enter Title : Cpp
Book Not Found.
~~~~~
```

Task 3: Issuing a Book

We will first Try to Issue a Book with Larger than Stock Available.

Secondly, We will Check with Quantity within Stock Available and get the Cost after issuing the Book.


~~~~~  
Enter your Choice : 2  
Enter Author : Bhagya  
Enter Title : Cpp  
Book Not Found.  
~~~~~

1. Add Book
 2. Search Book
 3. Issue Book
 4. Update Book Price
 5. Update Book Stock
 6. Statistical Analysis
 7. Exit
- ~~~~~

Enter your Choice : 3
Enter Author : Balaguruswamy
Enter Title : OOP
Enter No. of Copies: 8
Not Enough Copies Available. No Book Issued!
~~~~~

1. Add Book
  2. Search Book
  3. Issue Book
  4. Update Book Price
  5. Update Book Stock
  6. Statistical Analysis
  7. Exit
- ~~~~~

Enter your Choice : 3  
Enter Author : Balaguruswamy  
Enter Title : OOP  
Enter No. of Copies: 2  
**Book issued successfully. Cost : 1000**  
~~~~~

Task 4: Updating Book Price

- ```
~~~~~  
1. Add Book  
2. Search Book  
3. Issue Book  
4. Update Book Price  
5. Update Book Stock  
6. Statistical Analysis  
7. Exit  
~~~~~
```

```
Enter your Choice : 4
Enter Author : Morris
Enter Title : Electronics
Enter New Price : 1000
Price Updated.
~~~~~
```

#### Task 5: Updating Stocks

- ```
~~~~~  
1. Add Book
2. Search Book
3. Issue Book
4. Update Book Price
5. Update Book Stock
6. Statistical Analysis
7. Exit
~~~~~
```

```
Enter your Choice : 5  
Enter Author : Balaguruswamy  
Enter Title : OOP  
Enter Stock to be Added : 10  
Stock Updated.  
~~~~~
```

- ```
~~~~~  
1. Add Book  
2. Search Book  
3. Issue Book  
4. Update Book Price  
5. Update Book Stock  
6. Statistical Analysis  
7. Exit  
~~~~~
```

```
Enter your Choice : 3  
Enter Author : Balaguruswamy  
Enter Title : OOP  
Enter No. of Copies: 10
```

Now, 10 Books can be Issued!

```
Book issued successfully. Cost : 5000  
~~~~~
```

Task 6: Statistical Analysis

```
~~~~~  
1. Add Book  
2. Search Book  
3. Issue Book  
4. Update Book Price  
5. Update Book Stock  
6. Statistical Analysis  
7. Exit  
~~~~~
```

```
Enter your Choice : 6  
Successful Transactions : 6  
Failure Transactions : 2  
~~~~~
```

```
1. Add Book  
2. Search Book  
3. Issue Book  
4. Update Book Price  
5. Update Book Stock  
6. Statistical Analysis  
7. Exit  
~~~~~
```

```
Enter your Choice : 7  
Thank You for Visiting Our Book Shop!  
~~~~~
```

2.) Define a predicate `memCount (AList, BList, Count)` that is true if `AList` occurs `Count` times within `BList`. Define without using an accumulator.

Example: `memCount([a,b] , [a, [a,b,c], [a,b], [d,e,f], [a,b]] , 2).`

Code

```
% U19CS012 - BHAGYA VINOD RANA
```

```
% What I Understood from Question
```

```
% You are Given List Blist and You need to Check the Frequency of Alist in Blist
```

```
% and Check it is Equal to Count
```

```

% Prolog F(x) to Check if Two Lists are Equal
same([], []).
same([H1|R1], [H2|R2]):-
    H1 = H2,
    same(R1, R2).

% Code to Get the Frequency of Alist in Blist
freq(Alist,[],0).
freq(Alist,[H|T], C):-
    (same(Alist,H)) ->
    (
        freq(Alist,T,Cnt),
        C is Cnt+1
    )
    ;
    (
        freq(Alist,T, C)
    ).

% Main Function
memCount(Alist, Blist, Count):-
    % Function to Iterate the Blist and Count the Occurance of Alist in Ans
    freq(Alist,Blist,Ans),
    % Compare the Frequency Calculated with Count
    Ans = Count.

```

Output

```

6 ?- memCount([a,b],[a,[a,b,c],[a,b],[d,e,f],[a,b]],2).
true .

7 ?- memCount([a,b],[a,[a,b,c],[a,b],[d,e,f],[a,b]],X).
X = 2 .

8 ?- memCount([a,b],[a,[a,b,c],[a,b],[d,e,f],[a,b]],1).
false.

```

**You can observe
the frequency
of [a,b] in Blist
is 2.**

SUBMITTED BY: U19CS012

BHAGYA VINOD RANA