

Computer Programming

Dr. Deepak B Phatak

Dr. Supratik Chakraborty

Department of Computer Science and Engineering
IIT Bombay

Session: Simple operations on structures

Quick Recap of Relevant Topics

- Brief introduction to object-oriented programming
- Defining structures in C++

Overview of This Lecture

- Accessing members of structures
- Initializing and copying structures

Acknowledgment

- Some examples in this lecture are from
An Introduction to Programming Through C++
by Abhiram G. Ranade
McGraw Hill Education 2014
- All such examples indicated in slides with the citation
AGRBook

Recall: Library Information Management System

[Ref. AGRBook]



- We want to design a book check out/return/claim management system of a small library
 - How does the system work?
 - Every patron has a numerical id
 - Every book has an accession number
 - **Check out:** A patron can check out upto 3 books at any time
 - **Claim:** If X has not already checked out 3 books, she can claim a book checked out by Y
 - When Y returns the book, it is held for X and cannot be lent to others
 - **Return:** A patron can return a book checked out by her at any time
- No late charges!

Recall: Structures in C++

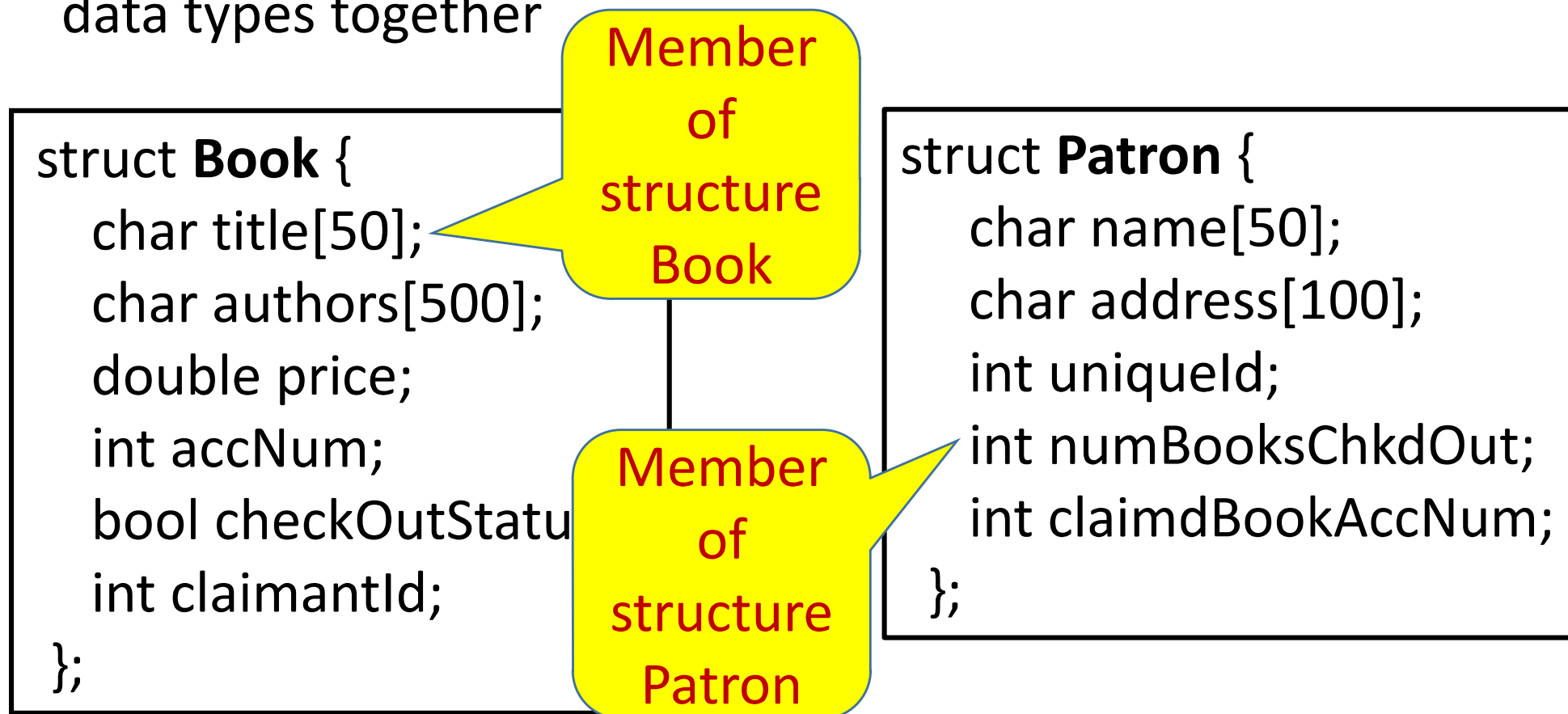
- Structures group a set of variables/arrays of possibly different data types together

```
struct Book {  
    char title[50];  
    char authors[500];  
    double price;  
    int accNum;  
    bool checkOutStatus;  
    int claimantId;  
};
```

```
struct Patron {  
    char name[50];  
    char address[100];  
    int uniqueId;  
    int numBooksChkOut;  
    int claimdBookAccNum;  
};
```

Recall: Structures in C++

- Structures group a set of variables/arrays of possibly different data types together



Recall: Structures in C++

- Variables and arrays of structure types can be declared

```
Book libraryShelf[1000];  
Book myChoice, yourChoice;
```

```
Patron libraryPatrons[200];  
Patron currentPatron, prevPatron;
```


Accessing Members of Structures

- How do we access the member named **price** of the object **myChoice** of (structure) type **Book** ?
- C++ provides the “.” operator for this:

myChoice.price

accesses the member named **price** of the object **myChoice**

```
struct Book {  
    char title[50];  
    char authors[500];  
    double price;  
    int accNum;  
    bool checkOutStatus;  
    int claimantId;  
};
```

```
Book myChoice;
```

Accessing Members of Structures

- `myChoice.price`

can be used in a program like any other double variable

Example program statements using `myChoice.price`

```
cin >> myChoice.price;
```

```
myChoice.price += 20;
```

```
cout << "Rs. " << myChoice.price;
```

```
struct Book {  
    char title[50];  
    char authors[500];  
    double price;  
    int accNum;  
    bool checkOutStatus;  
    int claimantId;  
};
```

```
Book myChoice;
```

Accessing Members of Structures

- `currPatron.name`

can be used in a program like any other character array

Example program statements using `currPatron.name`

```
if (currPatron.name[0] == 'S') {  
    cout << "Patron name: ";  
    cout << currPatron.name << endl;  
}
```

```
struct Patron {  
    char name[50];  
    char address[100];  
    int uniqueId;  
    int numBooksChkdOut;  
    int claimdBookAccNum;  
};
```

```
Patron currPatron;
```

Initializing Structures

- Recall declaring and initializing variables of simple data types

```
int index = 0;
```

```
char command = 'x';
```

Can we do similar initialization for structures?

Initializing Structures

```
struct Patron {  
    char name[50];  
    char address[100];  
    int uniqueId;  
    int numBooksChkdOut;  
    int claimdBookAccNum;  
};
```

Patron currPatron =
 {"Shashi Dev", "IIT Bombay, India", 2345, 0, -1};

Initializing Structures

```
struct Patron {  
    char name[50];  
    char address[100];  
    int uniqueId;  
    int numBooksChkdOut;  
    int claimdBookAccNum;  
};
```

currPatron object's members:
name: "Shashi Dev"
address: "IIT Bombay, India"
uniqueId: 2345
numBooksChkdOut: 0
claimdBookAccNum: -1

```
Patron currPatron =  
    {"Shashi Dev", "IIT Bombay, India", 2345, 0, -1};
```

Copying Structures

- Recall copying one variable to another for simple data types

```
int i, j;  
i = 27;  
j = i;
```

Can we similarly copy one object of a structure type to another object of the same structure type?

Copying structures

```
Patron currPatron, prevPatron;  
currPatron = {"Shashi Dev", "IIT Bombay, India", 2345, 0, -1};  
prevPatron = currPatron;
```

Each member of the object **currPatron** is copied to the corresponding member of the object **prevPatron** after executing **prevPatron = currPatron;**

Copying structures

```
Patron currPatron, prevPatron;  
currPatron = {"Shashi Dev", "IIT Bombay, India", 2345, 0, -1}  
prevPatron = currPatron;
```

currPatron before copying

name: "Shashi Dev"
address: "IIT Bombay, India"
uniqueId: 2345
numBooksChkdOut: 0
claimdBookAccNum: -1

prevPatron after copying

name: "Shashi Dev"
address: "IIT Bombay, India"
uniqueId: 2345
numBooksChkdOut: 0
claimdBookAccNum: -1

Summary

- The “.” operator to access members of structures
- Initializing structures
- Copying structures