

Chapter 1: Getting Started

1. Hello World!

Every C++ program must have exactly one global function named `main()`. The program starts by executing that function. The `int` value returned by `main()`, if any, is the program's return value to "the system." If no value is returned, the system will receive a value indicating successful completion. A nonzero value from `main()` indicates failure.

Not every operating system and execution environment make use of that return value: Linux/Unix-based and Mac-based environments often do, but Windows-based environments rarely do. For example, if you run Xcode in Mac and return zero from `main()`, you will see a line at the end indicated

```
| Program ended with exit code: 0
```

Typically, a program produces some output. Here is a program that writes Hello, World!

Hello.cpp

```
| #include <iostream>
|
| int main(){
|     std::cout << "Hello World!\n";
|     return 0;
| }
```

The line `#include <iostream>` instructs the compiler to include the declarations of the standard stream I/O facilities as found in `iostream`. Without these declarations, the expression

```
| std::cout << "Hello World!\n";
```

would make no sense. The operator `<<` ("put to") writes its second argument onto its first. In this case, the string literal `"Hello, World!\n"` is written onto the standard output stream `std::cout`. A string literal is a sequence of characters surrounded by double quotes. In a string literal, the backslash character `\` followed by another character denotes a single "special character." In this case, `\n` is the newline character. The `std::` specifies that the name `cout` is to be found in the standard-library namespace.

2. A More Serious Program

Problem statement: a bookstore keeps a file of transactions. Each transaction records the sale of a given book and contains an ISBN (International Standard Book Number, a unique identifier assigned to most books published throughout the world), the number of copies sold, and the price at which each copy was sold. Each transaction looks like

0-201-70353-X 4 24.99

where the first element is the ISBN, the second is the number of books sold, and the last is the sales price.

(PPT)

Reading an Unknown Number of Inputs

C++ 101: Write a program that sums from 0 to 9 using for loop control and print the sum.

Sum.cpp

A:

A logical extension of this program would be to ask the user to input a set of numbers to sum.

In this case, we won't know how many numbers to add. Instead, we'll keep reading numbers until there are no more numbers to read. Write a program and use while loop for the task.

```
Enter the number to be summed: 2
Enter the number to be summed (non-integer to quit): 3
Enter the number to be summed (non-integer to quit): 6
Enter the number to be summed (non-integer to quit): !
Sum is: 11
```

AddSum.cpp

A:

(PPT)

Introducing Classes

(Reading and writing Sales_item object)

ItemIO.cpp

```
#include <iostream>
#include "Sales_item.h"
int main()
{
    Sales_item book;
    // read ISBN, number of copies sold, and sales price
    std::cin >> book;
    // write ISBN, number of copies sold, total revenue, and average
    // price
    std::cout << book << std::endl;
    return 0;
}
```

Q: If the input to the program is 0-2-1-70353-X 4 24.99, what is the output?

A:

(Adding two Sales_item objects)

AddItem.cpp

```
#include <iostream>
#include "Sales_item.h"
int main()
{
    Sales_item item1, item2;
    std::cin >> item1 >> item2; // read a pair of transactions
    std::cout << item1 + item2 << std::endl; // print their sum
    return 0;
}
```

Q: If the inputs to the program are 0-2-1-70353-X 4 20.0 and 0-2-1-70353-X 3 20.0, what will be the output?

A:

(Class Member Function) A member function is a function that is defined by a class. Member functions are defined once for the class but are treated as members of each object. We refer to these operations as member functions because they usually operate on a specific object. A dot operator is used to call a member function.

AddItemCheck.cpp

```

#include <iostream>
#include "Sales_item.h"

int main()
{
    Sales_item item1, item2;

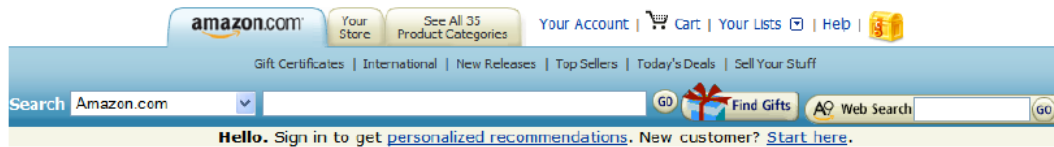
    std::cin >> item1 >> item2;
    // first check that item1 and item2 represent the same book
    if (item1.isbn() == item2.isbn()) {
        std::cout << item1 + item2 << std::endl;
        return 0;    // indicate success
    } else {
        std::cerr << "Data must refer to same ISBN"
                  << std::endl;
        return -1;   // indicate failure
    }
}

```

Q: where is the member function? What is its behavior?

A:

(Put things together and solve the problem)



(Our Task) write a program so we can

- read from the file with transactions
- computes the number of copies of each title sold
- the total revenue (總收入) from that book
- the average sales price.

To ease our life for now, we'll **assume** that all of the transactions for a given ISBN appear together.

(The Input File)

book_sales

0-201-70353-X 4 24.99
0-201-82470-1 4 45.39
0-201-88954-4 2 15.00
0-201-88954-4 5 12.00
0-201-88954-4 7 12.00
0-201-88954-4 2 12.00
0-399-82477-1 2 45.39
0-399-82477-1 3 45.39
0-201-78345-X 3 20.00
0-201-78345-X 2 25.00

(The Logics)

- Our program will combine the data for each ISBN in a Sales_item object named total.
- Each transaction we read from the standard input will be stored in a second
- Sales_item object named trans.
- Each time we read a new transaction we'll compare it to the Sales_item object total.
- If the objects refer to the same ISBN, we'll update total. Otherwise we'll print the value in total and reset it using the transaction we just read.

(The Code)

AvgPrice.cpp

```
#include <iostream>
#include "Sales_item.h"

int main()
{
    Sales_item total; // variable to hold data for the next transaction

    // read the first transaction and ensure that there are data
    if (std::cin >> total) {
        Sales_item trans; // variable to hold the running sum
        // read and process the remaining transactions
        while (std::cin >> trans) {
            // if we're still processing the same book
            if (total.isbn() == trans.isbn())
                total += trans; // update the running total
            else {
                // print results for the previous book
                std::cout << total << std::endl;
                total = trans; // total now refers to the next book
            }
        }
        std::cout << total << std::endl; // print the last transaction
    } else {
        // no input! warn the user
        std::cerr << "No data?!" << std::endl;
        return -1; // indicate failure
    }

    return 0;
}
```

Q: what are the outputs?

(Windows)

```
C:\Users\dchen_000\Desktop\Code20140916\Debug>Code20140916.exe < ../book_sales
```

(file redirection)

A: