

Ch 3: Expressions and Interactivity

Part 3

CS 1428
Fall 2011

Jill Seaman

Lecture 7

1

File Input/Output

- Variables are stored in Main Memory/RAM
 - values are lost when program is finished executing
- To preserve the values computed by the program: save them to a file
- Files are stored in Secondary Storage
- To have your program manipulate values stored in a file, they must be input into variables first.

2

File Stream Objects

- File stream data types:
 - ifstream
 - ofstream
- use `#include <fstream>` for these
- objects of type ofstream can output (write) values to a file. (like cout)
- objects of type ifstream can input (read) values from a file. (like cin)

3

Opening Files

- To input from a file, declare an ifstream variable and open a file by its name.

```
ifstream someFile;  
someFile.open("mydatafile.txt");
```
- To output to a file, declare an ofstream variable, and open a file by its name.

```
ofstream anotherFile;  
anotherFile.open("myoutputfile.txt");
```
- If the file "myoutputfile.txt" does not exist, it will be created.
- Stream variable is associated with the file.

4

Closing Files

- To close a file stream when you are done reading/writing:

```
someFile.close();  
anotherFile.close();
```

- Not required, but good practice.

5

Writing to Files

- Use the stream insertion operator: <<

```
#include <iostream>  
#include <fstream>  
using namespace std;  
  
int main() {  
    ofstream outFile;  
    outFile.open("demofile.txt");  
  
    int age;  
    cout << "Enter your age: ";  
    cin >> age;  
  
    outFile << "Age is: " << age << endl;  
    outFile.close();  
  
    return 0;  
}
```

6

Reading from Files

- Stream extraction operator: >>

Names.txt:
Tom
Dick
Harry

```
char name[25];  
  
ifstream inFile;  
inFile.open("Names.txt");  
inFile >> name;  
  
cout << name;  
inFile.close();
```

7

Reading from Files

- When opened, file stream's read position points to first character in file.
- extraction operator (>>) starts at read position and skips whitespace to read data into the variable.
- The read position then points to whitespace after the value it just read.

8

Reading Example

data.txt: 24 13
 34 100

```
ifstream inFile;  
inFile.open("data.txt");  
  
int a, b;  
  
inFile >> a;  
cout << a << " ";  
  
inFile >> a >> b;  
cout << a << " " << b << endl;  
  
inFile.close();
```

What is output by this code segment?

9

2.12 Scope

- A variable's scope is the part of the program that has access to the variable.
- Rule 1: A variable cannot be used before it is defined.

```
#include <iostream>  
using namespace std;  
  
int main () {  
    value = 150;    //error, use of value before it is defined  
    int value;  
    cout << value;  
}
```

10

3.6 Named Constants

- Literals do not have “meaningful names”

```
cost = price + (price * .0825);
```

- what is the meaning of .0825?
- Same literal may be used throughout a program, but may want to change it later.
 - maybe .0825 occurs in dozens of places in the code.
 - search and replace problem.

11

3.6 Named Constants

- Literals may be given names to be used in their place.

```
const double SALES_TAX_RATE = .0825;  
cost = price + (price * SALES_TAX_RATE);
```

- const makes the variable read-only
- initialization required
- All-caps for the name of the constant is just a convention

12

3.7 Multiple Assignment

- You can assign the same value to several variables in one statement:

```
a = b = c = 12;
```

- is equivalent to:

```
a = 12;  
b = 12;  
c = 12;
```

13

3.7 Combined Assignment

- Assignment statements often have this form:

```
number = number + 1;    //add 1 to number  
total = total + x;      //add x to total  
y = y / 2;              //divide y by 2
```

```
int number = 10;  
number = number + 1;  
cout << number << endl;
```

- C/C++ offers shorthand for these:

```
number += 1;    // short for number = number+1;  
total -= x;     // short for total = total-x;  
y /= 2;         // short for y = y / 2;
```

14

3.11 More Mathematical Library Functions

<code>pow</code>	<code>y = pow(x,d);</code>	returns x raised to the power d
<code>abs</code>	<code>y = abs(x);</code>	returns absolute value of x
<code>sqrt</code>	<code>y = sqrt(x);</code>	returns square root of x
<code>sin</code>	<code>y = sin(x);</code>	returns the sine of x (in radians)
etc.		

15

3.12 Hand Tracing a Program

- You be the computer. Track the values of the variables as the program executes.

```
int main() {
    double num1, num2, num3, avg;
    cout << "Enter first number";
    cin >> num1;
    cout << "Enter second number";
    cin >> num2;
    cout << "Enter third number";
    cin >> num3;

    avg = num1 + num2 + num3 / 3;

    cout << "The average is " << avg
         << endl;
}
```

16