Week 1

Operators, Data Types & I/O

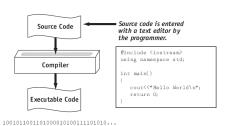
Gaddis: Chapters 1, 2, 3

CS 5301 Fall 2014

Jill Seaman

Programming

- A program is a set of instructions that the computer follows to perform a task
- It must be translated from a programming language (C++) to machine code in order to run on the machine.



Tony Gaddis, Starting out with C++: From Control Structures Through Objects 7th ed

2

Structure of a C++ Program

• Hello world:

```
//This program outputs a message to the screen
#include <iostream>
using namespace std;
int main() {
   cout << "Hello world!" << endl;
}</pre>
```

• In general:

```
//This is a comment
#include <includefile> ...
using namespace std;
int main() {
   statements ...
}
```

Variables, Data Types

- Variable: portion of memory that stores a value
- · Identifier: name of a program element
- Fundamental data types

short float bool int double char long long double

Variable Declaration statement

datatype identifier;

float hours;

• Variable Initialization statement:

datatype identifier = constant;

int count = 0;

Integer types

Integers are whole numbers such as 12, 7, and -99

Data Type	Range
short	-23,768 to 32,767
int	-2,147,483,648 to 2,147,483,647
long	-2,147,483,648 to 2,147,483,647

- char type stores characters such as 'A', '@', and '9'
- The ascii code value (an integer) of the character is stored in memory.

Floating-point types

(and bool)

- Floating point types store real numbers such as 12.45 and -3.8
- They are stored using scientific notation.

Data Type	Range
float	$\pm 3.4E-38$ to $\pm 3.4E38$
double	$\pm 1.7E-308$ to $\pm 1.7E308$
long double	$\pm 1.7E-308$ to $\pm 1.7E308$

- bool type stores values that are true or false
 - false is 0, true is 1.

6

Constants

• Literals (specific value of a given type)

1	
75	
-2	

п	
	12.45
	-3.8
	6.25e-5

true false

'A' '2'

 Named Constants: variable whose value cannot be changed

Assignment statement, expressions

To change the value of a variable:

variable = expression;

count = 10;

- * The lefthand side must be a variable
- The righthand side is an expression of the right type
- What is an expression?
 - * an expression has a type and evaluates to a value
 - literal
 - named constant
 - variable
 - arithmetic expression
 - + etc.

Arithmetic Operations

7 * 2

8 - 5 * 10

(3 * 10) / 2

- arithmetic operators:
 - + addition
 - subtraction
 - * multiplication
 - / division
 - % modulo (remainder)
- Integer division:

```
14 ÷ 3 = 4 r. 2 (because 4*3+2 = 14)

14/3 => 4 in C++

14%3 => 2 in C++

14.0/3.0 => 4.6666667 in C++
```

9

Operator precedence

- In an expression with multiple operators, which one happens first?
- Use this order for different operators:

```
+- (unary)

* / %

+ - (binary)

<><=>=
=!

We will study relational and logical operators next week.
```

- Use this order for multiple occurrences of the same operator
 - (unary negation) associates right to left
 - *, /, %, +, associate left to right

10

Basic Input/Output

- Output (cout and <<)
 - sends data to the screen (console)

```
cout << expression;
cout << expr1 << expr2;
cout << "hello";
cout << "Count is: " << count << endl;</pre>
```

- Input (cin and >>)
 - receives data typed in from the keyboard (stops at space)

```
cin >> variable;
cin >> var1 >> var2;

right hand side must be a variable!

cout << "Enter the height and width: ";
cin >> height >> width;
cout << "The height is " << height << endl;</pre>
```

Formatting output

- Goal: control how output displays for numeric data
- these require #include<iomanip>
- setw(x): print next value in a field at least x spaces wide (right justified, padded with spaces).

```
cout << setw(6) << 1234 << setw(6) << 5 << endl;
cout << setw(6) << 5 << setw(6) << 1234 << endl;
```

- fixed: always use decimal notation (not scientific)
- setprecision(x): when used with fixed, print floating point values using x digits after the decimal

```
cout << fixed << setprecision(2);
cout << 3.14159 << endl;
float x = 20;
cout << x << endl;</pre>
```

3.14 20.00

12

1234

The string class

- <u>string literals:</u> represent sequences of chars, inside of double quotes: | cout << "Hello"; |
- To define string variables:

```
string firstName, lastName;
```

Operations include:

```
string name;
name = "George";
cout << name.size() << " ";
cout << name[2] << endl;</pre>
```

- = for assignment

- .size() function for length
- [n] to access one character in the nth position.

13

Type conversions

- Implicit
 - assignment:

```
int x;
double d = 3.1415;
x = d;
cout << x << endl;</pre>
```

the type of expression on the right will be converted to type of variable on left, possibly losing information.

- binary operations:

```
int x = 10;
double d = 2.3;
cout << x + d << endl;</pre>
```

the operand with the lower ranking type is converted to the type of the other.

Explicit

```
int x, y;
...
float avg = static_cast<float>(x)/y;

Of

float avg = x/(float)y; //c-style notation
```

Order of types:

long double
double
float
long
int
char

Comments

Single-Line Comments

```
// this text is ignored, to end of line
```

Multi-Line Comments

```
/* Anything occurring between a slash star and
  a star slash is ignored. Even when spanning
  multiple lines. */
```

 Use comments to explain your code to a human reader who knows C++.

Programming Style

- The visual organization of the source code
- Purpose: improve the readability of the source code
- Includes the use of spaces, tabs, and blank lines
- Includes naming of variables, constants.
- Includes where to use comments.
- Common elements to improve readability:
 - Braces { } aligned vertically
 - · Indentation of statements within a set of braces
 - Lines shorter than 80 characters.