Topic 2

- 1. Variables
- 2. Arithmetic
- 3. Input and output
- 4. Problem solving: first do it by hand
- 5. Strings
- 6. Chapter summary

Arithmetic Operators



C++ has the same arithmetic operators as a calculator:

- * for multiplication: a * b
 (not a b or ab as in math)
 - for division: **a / b** (not ÷ or a fraction bar as in math)
- + for addition: **a + b**
- for subtraction: **a b**

Just as in regular algebraic notation, * and / have higher precedence than + and –.

In a + b / 2, the b / 2 happens first.

Increment and Decrement

• Changing a variable by adding or subtracting 1 is so common that there is a special shorthand for these:

The increment and decrement operators.

count++; // add 1 to count
count--; // subtract 1 from count

Example:

What is the value of variable count after the code below?

```
int count = 3;
count--;
count = count + 2;
count++;
```

C++ was based on C and so it's one better than C, right?

Guess how C++ got its name!

The % operator computes the remainder of an integer division.

It is called the *modulus operator* (also modulo and mod)

It has nothing to do with the % key on a calculator

Integer Division and Remainder Example

- You want to determine the value in dollars and cents stored in the piggy bank.
- You obtain the dollars through an integer division by 100.
 The integer division discards the remainder.
- To obtain the remainder (the cents), use the % operator:

```
int pennies = 1729;
int dollars = pennies / 100; // Sets dollars to 17
int cents = pennies % 100; // Sets cents to 29
```

(yes, 100 is a magic number)

More Integer Division and Remainder Examples

• What is the result from each of the following?



When both dividend (3) and divisor (2) are integers

• 3 / 2 = 1 (Why???)

When at least one of dividend and divisor is a real number (number with decimals)

- 3. / 2 = 1.5
- 3 / 2.0 = 1.5
- 3.0 / 2.0 = 1.5

Why do we need integer division?

- 6 kids with 10 apples
- Apples cannot be cut into piece decimal
- How many apple(s) can a kid get?
- How many apples are left?









- 6 kids with 10 apples
- Apples cannot be cut into pieces
- How many apple(s) can a kid get? 1
- How many apples are left?







- 6 kids with 10 apples
- Apples cannot be cut into pieces
- How many apple(s) can a kid get?
 10 /
 6 = 1



10 % 6 = 4



- Integer division: When both dividend (say, 10) and divisor (say, 6) are integers, discard fraction part of division result. So 10 / 6 = 1.
- How many apple(s) can a kid get? 10 /







123 minutes = ? Hours ? minutes

123 minutes = ? Hours ? Minutes Solution:

- (1) How many whole hours does 123 minutes have?
 - One hour = 60 minutes.

123 / 60 = <u>2 hours</u>

(2) After the above 2 hours, how many more minutes does 123 minutes have?

123 – 60 * 2 = 3. (* means multiplication in Java) or we use remainder operator %:

123 % 60 = <u>3 minutes</u>

Similarly, computer keeps track of system time...

Starting from 1 January 1970 00:00:00, there were 10000000 seconds elapsed, what was that time? 00:00:00 +100,000,000

secs = ?

Years ? Days



(1) 10000000 secs = ? minutes ? seconds 10000000 / 60 = 1666666 minutes 10000000 % 60 = 40 seconds. (2) 1666666 minutes = ? Hours ? minutes 16666666 / 60 = 27777 hours 1666666 % 60 = 1666666 - 60 * 27777

= <u>46 minutes</u>

(1) 10000000 secs = 1666666 mins 40 secs(2) 1666666 mins = 27777 Hours 46 mins (3) 27777 hours = ? Days ? hours 27777 / 24 = 1157 days 27777 % 24 = 27777 - 1157 * 24 = 9 hours (4) 1157 days = ? Years ? days 1157 / 365 = 3 years 1157 % 365 = 1157 - 3 * 365 = 62days

Starting from 1 January 1970 00:00:00, there were 100,000,000 seconds elapsed, what was that time?

- (1) 10000000 secs = 1666666 mins <u>40</u> <u>secs</u>
- (2) 1666666 minutes = 27777 Hours <u>46</u> <u>minutes</u>
- (3) 27777 hours = 1157 Days <u>9 hours</u>
 (4) 1157 days = 3 Years <u>62 days</u>
 The time was 1973 March 3, 09: 46: 40.



• What time in the afternoon for 14:00 ?



• What time in the afternoon for 17:00?

 You are at top of a round roller coaster, which turns another 380 degree in the next few minutes. What will be your new position?



Remainder Operator % Example 2

Now is 9am. After 5 hours, it is 2pm.
 (9 + 5) % 12 = 2



Converting Floating-Point Numbers to Integers

• When a floating-point value is assigned to an integer variable, the fractional part is discarded:

 You probably want to round to the *nearest* integer. To round a positive floating-point value to the nearest integer, add 0.5 and then convert to an integer:

```
int dollars = price + 0.5;
    // Rounds to the nearest integer
```

What about this?

$$b + \left(1 + \frac{r}{100}\right)^n$$

Inside the parentheses is easy:

1 + (r / 100)

But that raised to the *n*?

Powers and Roots - #include <cmath>

- In C++, there are no symbols for powers and roots. To compute them, you must call *functions*.
- The C++ library defines many mathematical functions such as sqrt (square root) and pow (raising to a power).
- To use the functions in this library, called the **cmath** library, you must place the line:

#include <cmath>

at the top of your program file.

It is also necessary to include

using namespace std;

at the top of your program file.

The power function has the base followed by a comma followed by the power to raise the base to:

```
pow(base, exponent)
```

Using the **pow** function:

double balance = b * pow(1 + r / 100, n);

Mathematical Expression	C++ Expression	Comments			
$\frac{x+y}{2}$	(x + y) / 2	The parentheses are required; $x + y/2$ computes $x + (y/2)$.			
$\frac{xy}{2}$	x * y / 2	Parentheses are not required; operators with the same precedence are evaluated left to right. xy as a math expression is $x*y$ in C++			
$\left(1 + \frac{r}{100}\right)^n$	pow(1 + r / 100, n)	Remember to add #include <cmath> to the top of your program.</cmath>			
$\sqrt{a^2+b^2}$	sqrt(a * a + b * b)	a * a is simpler than pow(a, 2).			
$\frac{i+j+k}{3}$	(i + j + k) / 3.0	If <i>i</i> , <i>j</i> , and <i>k</i> are integers, using a denominator of 3.0 forces floating-point division.			

Other Mathematical Functions (from <cmath>): Table 6

Table 6 Other Mathematical Functions					
Function	Description				
sin(x)	sine of x (x in radians)				
cos(x)	cosine of x				
tan(x)	tangent of <i>x</i>				
log10(x)	$(\text{decimal log}) \log_{10}(x), x > 0$				
abs(x)	absolute value $ x $				

Example:

double population = 73693997551.0; double decimal log = log10(population);

Math Function Examples

• Compute the result of each:

	pow(10,	3)			
	sqrt(100)			
	abs(3 –	10)			
	log10(10	00)			
	max(3, -	10)			
	cos(3.14	1592653	5)		
	tan(M_PI	(/4)			
//M_PI cc	nstant is	defined	in	cmath	library

Common Error – Unintended Integer Division

- If both arguments of / are integers, the remainder is discarded:
 7 / 3 is 2, not 2.5
- but
 - 7.0 / 4.0 7 / 4.0 7.0 / 4
- all yield 1.75.

Common Error – Unintended Integer Division, cont.

- It is unfortunate that C++ uses the same symbol: / for both integer and floating-point division. These are really quite different operations.
- It is a common error to use integer division by accident. Consider this segment that computes the average of three integers:

```
cout << "Please enter your last three test scores: ";
int s1;
int s2;
int s3;
cin >> s1 >> s2 >> s3;
double average = (s1 + s2 + s3) / 3; //ERROR
cout << "Your average score is " << average << endl;</pre>
```

More on Unintended Integer Division

- What could be wrong with that?
- Of course, in math the exact average of s1, s2, and s3 is

(s1+ s2+ s3) / 3

- Here, however, the / denotes integer division because
- both (s1+s2+s3) and 3 are integers.
- For example, if the scores add up to 14, the average = 4.
- Yes, the result of the integer division of 14 by 3 is 4, and the fractional 0.66667 <u>is discarded</u>.
- That integer 4 is then moved into the double variable **average**.

The remedy is to make the numerator or denominator into a floating-point number:

double total = s1 + s2 + s3; double average = total / 3;

or

double average = (s1 + s2 + s3) / 3.0;

Common Error – Unbalanced Parentheses

Consider the expression

(-(b * b - 4 * a * c) / (2 * a))

What is wrong with it?

The parentheses are *unbalanced*. This is very common with complicated expressions.

The Muttering Method

Count starting with 1 at the 1st parenthesis add one for each left paren (and subtract one for each right paren)

If your count is not 0 when you finish, or if you ever drop to -1, then... STOP, something is wrong.

Common Error – Forgetting Header Files

- Every program that carries out input or output needs the <iostream> header.
- If you use mathematical functions such as sqrt, you need to include <cmath>.
- If you forget to include the appropriate header file, the compiler will not know symbols such as cout or sqrt.
- If the compiler complains about an undefined function or symbol, check your header files.

Including the Right Header Files

- Sometimes you may not know which header file to include.
- Suppose you want to compute the absolute value of an integer using the abs function.
- As it happens, this version of **abs** is not defined in the <cmath> header but in <cstdlib>.
- How can you find the correct header file?
- Why do you think Tim Berners-Lee invented going online?
- Use a reference site on the Internet such as: <u>http://www.cplusplus.com</u>, or just Google "C++ abs()"

It is easier to read

x1 = (-b + sqrt(b * b - 4 * a * c)) / (2 * a);

than

$$x1=(-b+sqrt(b*b-4*a*c))/(2*a);$$

Itreallyiseasiertoreadwithspaces!

So always use spaces around all operators: + - * / % =

Spaces in Expressions: Unary Minus, Parentheses

- However, don't put a space after a *unary* minus: that's a used to negate a single quantity like this: –b
- That way, it can be easily distinguished from a *binary* minus, as in a b
- It is customary *not* to put a space between a function name and the parentheses.

Write sqrt(x) not sqrt (x)

Casts

- Occasionally, you need to store a value into a variable of a different type, or print it in a different way.
- A *cast* is a conversion from one type (such as int) to another type (such as double).
- For example, how to print or capture an exact quotient from two int variables?

```
int x= 25;
int y = 10;
cout << "The quotient is " << x / y;
//gives int quotient of 2, not what we want
```

The *cast* conversion syntax:

```
static_cast<newtype>( data_to_convert)
```

 For example, to get an exact quotient, we cast one of the int variables to a double <u>before</u> dividing

```
int x= 25;
int y = 10;
cout << x / static_cast<double>(y);
//gives double quotient of 2.5
```

 An older version of the *cast* conversion syntax also works, but its use is discouraged:

```
(newtype)data_to_convert
```

```
cout << x / (double)y;
```

```
//gives double quotient of 2.5
```

Combining Assignment and Arithmetic

- In C++, you can combine arithmetic and assignments.
- For example, the statement

total += cans * CAN VOLUME;

is a shortcut for

```
total = total + cans * CAN_VOLUME;
```

• Similarly,

total *= 2;

is another way of writing

total = total * 2;

• Many programmers *prefer* using this form of coding.