

Computer Science I

Chapter III - Expressions and Interactivity

Hanoi University of Science and Technology

October 22, 2025



Slide contents are mainly based on the following documents:

- Tony Gaddis, Starting out with C++, Six Edition, Pearson, 2009
- Tony Gaddis, Power Points Slides:
<https://www.ums1.edu/~antognolij/gaddis.html>



The `cin` Object

- Standard input object
- Like `cout`, requires `iostream` file
- Used to read input from keyboard
- Information retrieved from `cin` with `>>`
- Input is stored in one or more variables



The cin Object

Program 3-1

```
1 // This program asks the user to enter the length and width of
2 // a rectangle. It calculates the rectangle's area and displays
3 // the value on the screen.
4 #include <iostream>
5 using namespace std;
6
7 int main()
8 {
9     int length, width, area;
10
11     cout << "This program calculates the area of a ";
12     cout << "rectangle.\n";
13     cout << "What is the length of the rectangle? ";
14     cin >> length;
15     cout << "What is the width of the rectangle? ";
16     cin >> width;
17     area = length * width;
18     cout << "The area of the rectangle is " << area << ".\n";
19     return 0;
20 }
```

Program Output with Example Input Shown in Bold

This program calculates the area of a rectangle.
What is the length of the rectangle? **10 [Enter]**
What is the width of the rectangle? **20 [Enter]**
The area of the rectangle is 200.



The cin Object

- cin converts data to the type that matches the variable:

```
int height;  
cout << "How tall is the room?  ";  
cin >> height;
```



Displaying a Prompt

- A prompt is a message that instructs the user to enter data.
- You should always use `cout` to display a prompt before each `cin` statement.

```
cout << "How tall is the room? ";  
cin >> height;
```



The cin Object

- Can be used to input more than one value:
`cin >> height >> width;`
- Multiple values from keyboard must be separated by spaces
- Order is important: first value entered goes to first variable, etc.



The cin Object

Program 3-2

```
1 // This program asks the user to enter the length and width of
2 // a rectangle. It calculates the rectangle's area and displays
3 // the value on the screen.
4 #include <iostream>
5 using namespace std;
6
7 int main()
8 {
9     int length, width, area;
10
11     cout << "This program calculates the area of a ";
12     cout << "rectangle.\n";
13     cout << "Enter the length and width of the rectangle ";
14     cout << "separated by a space.\n";
15     cin >> length >> width;
16     area = length * width;
17     cout << "The area of the rectangle is " << area << endl;
18     return 0;
19 }
```

Program Output with Example Input Shown in Bold

This program calculates the area of a rectangle.

Enter the length and width of the rectangle separated by a space.

10 20 [Enter]

The area of the rectangle is 200



The cin Object

Program 3-3

```
1  // This program demonstrates how cin can read multiple val
2  // of different data types.
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      int whole;
9      double fractional;
10     char letter;
11
12     cout << "Enter an integer, a double, and a character: "
13     cin >> whole >> fractional >> letter;
14     cout << "Whole: " << whole << endl;
15     cout << "Fractional: " << fractional << endl;
16     cout << "Letter: " << letter << endl;
17     return 0;
18 }
```

Program Output with Example Input Shown in Bold

```
Enter an integer, a double, and a character: 4 5.7 b [Enter]
Whole: 4
Fractional: 5.7
Letter: b
```



Mathematical Expressions

- Can create complex expressions using multiple mathematical operators
- An expression can be a literal, a variable, or a mathematical combination of constants and variables
- Can be used in assignment, cout, other statements:

```
area = 2 * PI * radius;  
cout << "border is:  " << 2*(1+w);
```



Order of Operations

- In an expression with more than one operator, evaluate in this order:
 - (unary negation)
 - * / %, in order, left to right
 - + -, in order, left to right
- parentheses () can be used to override the order of
- Example: $2 + 2 * - 2 - (2 + 2)$



Order of Operations

Table 3-2 Some Simple Expressions and Their Values

Expression	Value
$5 + 2 * 4$	13
$10 / 2 - 3$	2
$8 + 12 * 2 - 4$	28
$4 + 17 \% 2 - 1$	4
$6 - 3 * 2 + 7 - 1$	6



Table 3-4 More Simple Expressions and Their Values

Expression	Value
$(5 + 2) * 4$	28
$10 / (5 - 3)$	5
$8 + 12 * (6 - 2)$	56
$(4 + 17) \% 2 - 1$	0
$(6 - 3) * (2 + 7) / 3$	9



Algebraic Expressions

- Multiplication requires an operator:

$\text{Area} = lw$ is written as $\text{Area} = l * w$;

- There is no exponentiation operator:

$\text{Area} = s^2$ is written as $\text{Area} = \text{pow}(s, 2)$;

- Parentheses may be needed to maintain order of operations:

$m = \frac{y_2 - y_1}{x_2 - x_1}$ is written as $m = (y_2 - y_1) / (x_2 - x_1)$;



Algebraic Expressions

Table 3-5 Algebraic and C++ Multiplication Expressions

Algebraic Expression	Operation	C++ Equivalent
$6B$	6 times B	<code>6 * B</code>
$(3)(12)$	3 times 12	<code>3 * 12</code>
$4xy$	4 times x times y	<code>4 * x * y</code>



Type Casting

- Used for manual data type conversion
- Useful for floating point division using ints:

```
double m;  
m = static_cast<double> (y2-y1) / (x2-x1);
```

- Useful to see int value of a char variable:

```
char ch = 'C';  
cout << ch << " is " << static_cast<int>(ch);
```



Type Casting

Program 3-9

```
1 // This program uses a type cast to avoid integer division.
2 #include <iostream>
3 using namespace std;
4
5 int main()
6 {
7     int books;           // Number of books to read
8     int months;          // Number of months spent reading
9     double perMonth;     // Average number of books per month
10
11     cout << "How many books do you plan to read? ";
12     cin >> books;
13     cout << "How many months will it take you to read them? ";
14     cin >> months;
15     perMonth = static_cast<double>(books) / months;
16     cout << "That is " << perMonth << " books per month.\n";
17     return 0;
18 }
```

Program Output with Example Input Shown in Bold

How many books do you plan to read? **30** [Enter]
How many months will it take you to read them? **7** [Enter]
That is 4.28571 books per month.

Does it work? `static_cast<double> (30 / 7);`



C-Style and Prestandard Type Cast Expressions

- C-Style cast: data type name in ()
`cout << ch << " is " << (int)ch;`
- Prestandard C++ cast: value in ()
`cout << ch << " is " << int(ch);`
- Both are still supported in C++, although `static_cast` is preferred



Multiple Assignment and Combined Assignment

- The `=` can be used to assign a value to multiple variables:

`x = y = z = 5;`

- Value of `=` is the value that is assigned

- Associates right to left:

`x = (y = (z = 5));`



Multiple Assignment and Combined Assignment

Table 3-8 (Assume $x = 6$)

Statement	What It Does	Value of x After the Statement
$x = x + 4;$	Adds 4 to x	10
$x = x - 3;$	Subtracts 3 from x	3
$x = x * 10;$	Multiplies x by 10	60
$x = x / 2;$	Divides x by 2	3
$x = x \% 4$	Makes x the remainder of $x / 4$	2



Combined Assignment

- Look at the following statement:
`sum = sum + 1;` This adds 1 to the variable `sum`.
- The combined assignment operators provide a shorthand for these types of statements.
- The statement
`sum = sum + 1;`
is equivalent to
`sum += 1;`



Combined Assignment Operators

Table 3-8 (Assume $x = 6$)

Statement	What It Does	Value of x After the Statement
$x = x + 4;$	Adds 4 to x	10
$x = x - 3;$	Subtracts 3 from x	3
$x = x * 10;$	Multiplies x by 10	60
$x = x / 2;$	Divides x by 2	3
$x = x \% 4$	Makes x the remainder of $x / 4$	2



Formatting Output

- Can control how output displays for numeric, string data
 - size
 - position
 - number of digits
- Requires `iomanip` header file



Formatting Output

- Some affect just the next value displayed:
 - `setw(x)`: print in a field at least `x` spaces wide. Use more spaces if field is not wide enough



Formatting Output

Program 3-13

```
1 // This program displays three rows of numbers.
2 #include <iostream>
3 #include <iomanip>      // Required for setw
4 using namespace std;
5
6 int main()
7 {
8     int num1 = 2897, num2 = 5,    num3 = 837,
9         num4 = 34,   num5 = 7,    num6 = 1623,
10        num7 = 390,  num8 = 3456, num9 = 12;
11
12    // Display the first row of numbers
13    cout << setw(6) << num1 << setw(6)
14         << num2 << setw(6) << num3 << endl;
15
16    // Display the second row of numbers
17    cout << setw(6) << num4 << setw(6)
18         << num5 << setw(6) << num6 << endl;
19
20    // Display the third row of numbers
21    cout << setw(6) << num7 << setw(6)
22         << num8 << setw(6) << num9 << endl;
23    return 0;
24 }
```

Program Output

```
2897    5    837
  34    7   1623
 390 3456    12
```



Formatting Output

- Some affect values until changed again:
 - `fixed`: use decimal notation for floating-point values
 - `setprecision(x)`: when used with `fixed`, print floating-point value using `x` digits after the decimal. Without `fixed`, print floating-point value using `x` significant digits
 - `showpoint`: always print decimal for floating-point values



Formatting Output

Program 3-17

```
1 // This program asks for sales figures for 3 days. The total
2 // sales are calculated and displayed in a table.
3 #include <iostream>
4 #include <iomanip>
5 using namespace std;
6
7 int main()
8 {
9     double day1, day2, day3, total;
10
11     // Get the sales for each day.
12     cout << "Enter the sales for day 1: ";
13     cin >> day1;
14     cout << "Enter the sales for day 2: ";
15     cin >> day2;
16     cout << "Enter the sales for day 3: ";
17     cin >> day3;
18
19     // Calculate the total sales.
20     total = day1 + day2 + day3;
21
22     // Display the sales figures.
23     cout << "\nSales Figures\n";
24     cout << "-----\n";
25     cout << setprecision(2) << fixed;
26     cout << "Day 1: " << setw(8) << day1 << endl;
27     cout << "Day 2: " << setw(8) << day2 << endl;
28     cout << "Day 3: " << setw(8) << day3 << endl;
29     cout << "Total: " << setw(8) << total << endl;
30     return 0;
31 }
```

Program Output with Example Input Shown in Bold

```
Enter the sales for day 1: 1321.87 [Enter]
Enter the sales for day 2: 1869.26 [Enter]
Enter the sales for day 3: 1403.77 [Enter]
```

```
Sales Figures
-----
Day 1:   1321.87
Day 2:   1869.26
Day 3:   1403.77
Total:   4594.90
```



Formatting Output

Table 3-12

Stream Manipulator	Description
<code>setw(<i>n</i>)</code>	Establishes a print field of <i>n</i> spaces.
<code>fixed</code>	Displays floating-point numbers in fixed point notation.
<code>showpoint</code>	Causes a decimal point and trailing zeroes to be displayed, even if there is no fractional part.
<code>setprecision(<i>n</i>)</code>	Sets the precision of floating-point numbers.
<code>left</code>	Causes subsequent output to be left justified.
<code>right</code>	Causes subsequent output to be right justified.



Working with Characters and string Objects

- Using `cin` with the `>>` operator to input strings can cause problems
- It passes over and ignores any leading whitespace characters (spaces, tabs, or line breaks)
- To work around this problem, you can use a C++ function named `getline`.



Working with Characters and string Objects

Program 3-19

```
1  // This program demonstrates using the getline function
2  // to read character data into a string object.
3  #include <iostream>
4  #include <string>
5  using namespace std;
6
7  int main()
8  {
9      string name;
10     string city;
11
12     cout << "Please enter your name: ";
13     getline(cin, name);
14     cout << "Enter the city you live in: ";
15     getline(cin, city);
16
17     cout << "Hello, " << name << endl;
18     cout << "You live in " << city << endl;
19     return 0;
20 }
```

Program Output with Example Input Shown in Bold

Please enter your name: **Kate Smith [Enter]**
Enter the city you live in: **Raleigh [Enter]**
Hello, Kate Smith
You live in Raleigh



Working with Characters and string Objects

- To read a single character:

- Use `cin`:

```
char ch;  
cout << "Strike any key to continue";  
cin >> ch;
```

Problem: will skip over blanks, tabs, ...

- Use `cin.get()`:

```
cin.get(ch);
```

Will read the next character entered, even whitespace



Working with Characters and string Objects

Program 3-21

```
1 // This program demonstrates three ways
2 // to use cin.get() to pause a program.
3 #include <iostream>
4 using namespace std;
5
6 int main()
7 {
8     char ch;
9
10    cout << "This program has paused. Press Enter to continue.";
11    cin.get(ch);
12    cout << "It has paused a second time. Please press Enter again.";
13    ch = cin.get();
14    cout << "It has paused a third time. Please press Enter again.";
15    cin.get();
16    cout << "Thank you!";
17    return 0;
18 }
```

Program Output with Example Input Shown in Bold

This program has paused. Press Enter to continue. **[Enter]**
It has paused a second time. Please press Enter again. **[Enter]**
It has paused a third time. Please press Enter again. **[Enter]**
Thank you!



string Member Functions and Operators

- To find the length of a string:

```
string state = "Texas";  
int size = state.length();
```

- To concatenate (join) multiple strings:

```
greeting2 = greeting1 + name1;  
greeting1 = greeting1 + name2;
```

- Or using the += combined assignment operator:

```
greeting1 += name2;
```



More Mathematical Library Functions

- Require `cmath` header file
- Take `double` as input, return a `double`
- Commonly used functions:

`sin` Sine

`cos` Cosine

`tan` Tangent

`sqrt` Square root

`log` Natural (e) log

`abs` Absolute value



More Mathematical Library Functions

- These require `cstdlib` header file
- `rand()`: returns a random number (`int`) between 0 and the largest `int` the compute holds. Yields same sequence of numbers each time program is run.
- `srand(x)`: initializes random number generator with `unsigned int x`



Hand Tracing a Program

- Hand trace a program: act as if you are the computer, executing a program:
 - step through and 'execute' each statement, one-by-one
 - record the contents of variables after statement execution, using a hand trace chart (table)
- Useful to locate logic or mathematical errors



Hand Tracing a Program

Program 3-26 (with hand trace chart filled)

```
1 // This program asks for three numbers, then
2 // displays the average of the numbers.
3 #include <iostream>
4 using namespace std;

5 int main()

6 {
7     double num1, num2, num3, avg;
8     cout << "Enter the first number: ";
9     cin >> num1;
10    cout << "Enter the second number: ";
11    cin >> num2;
12    cout << "Enter the third number: ";
13    cin >> num3;
14    avg = num1 + num2 + num3 / 3;
15    cout << "The average is " << avg << endl;
16    return 0;
17 }
```

num1	num2	num3	avg
?	?	?	?
?	?	?	?
10	?	?	?
10	?	?	?
10	20	?	?
10	20	?	?
10	20	30	?
10	20	30	40
10	20	30	40



A Case Study

Problem: General Crates, Inc. builds custom-designed wooden crates. With materials and labor, it costs GCI \$0.23 per cubic foot to build a crate. In turn, they charge their customers \$0.50 per cubic foot for the crate. You have been asked to write a program that calculates the volume (in cubic feet), cost, customer price, and profit of any crate GCI builds.



Table 3-14

Constant or Variable	Description
<code>COST_PER_CUBIC_FOOT</code>	A named constant, declared as a <code>double</code> and initialized with the value 0.23. This represents the cost to build a crate, per cubic foot.
<code>CHARGE_PER_CUBIC_FOOT</code>	A named constant, declared as a <code>double</code> and initialized with the value 0.5. This represents the amount charged for a crate, per cubic foot.
<code>length</code>	A <code>double</code> variable to hold the length of the crate, which is input by the user.
<code>width</code>	A <code>double</code> variable to hold the width of the crate, which is input by the user.
<code>height</code>	A <code>double</code> variable to hold the height of the crate, which is input by the user.
<code>volume</code>	A <code>double</code> variable to hold the volume of the crate. The value stored in this variable is calculated.
<code>cost</code>	A <code>double</code> variable to hold the cost of building the crate. The value stored in this variable is calculated.
<code>charge</code>	A <code>double</code> variable to hold the amount charged to the customer for the crate. The value stored in this variable is calculated.
<code>profit</code>	A <code>double</code> variable to hold the profit GCI makes from the crate. The value stored in this variable is calculated.



The program must perform the following general steps:

- Step 1: Ask the user to enter the dimensions of the crate
- Step 2: Calculate:
 - the crate's volume
 - the cost of building the crate
 - the customer's charge
 - the profit made
- Step 3: Display the data calculated in Step 2.



Calculations

- $\text{volume} = \text{length} \times \text{width} \times \text{height}$
- $\text{cost} = \text{volume} \times 0.23$
- $\text{charge} = \text{volume} \times 0.5$
- $\text{profit} = \text{charge} - \text{cost}$



Second week

- Programming Challenges: 1 - 7
- Homework: 8 - 24

