

**Illustrations List** [\(Main Page\)](#)

- Fig. 1.1** A typical C++ environment.  
**Fig. 1.2** Text printing program.  
**Fig. 1.3** Some common escape sequences.  
**Fig. 1.4** Printing on one line with separate statements using **cout**.  
**Fig. 1.5** Printing on multiple lines with a single statement using **cout**.  
**Fig. 1.6** An addition program.  
**Fig. 1.7** A memory location showing the name and value of a variable.  
**Fig. 1.8** Memory locations after values for two variables have been input.  
**Fig. 1.9** Memory locations after a calculation.  
**Fig. 1.10** Arithmetic operators.  
**Fig. 1.11** Precedence of arithmetic operators.  
**Fig. 1.12** Order in which a second-degree polynomial is evaluated.  
**Fig. 1.13** Equality and relational operators.  
**Fig. 1.14** Using equality and relational operators.  
**Fig. 1.15** Precedence and associativity of the operators discussed so far.  
**Fig. 1.16** Using new-style header files.

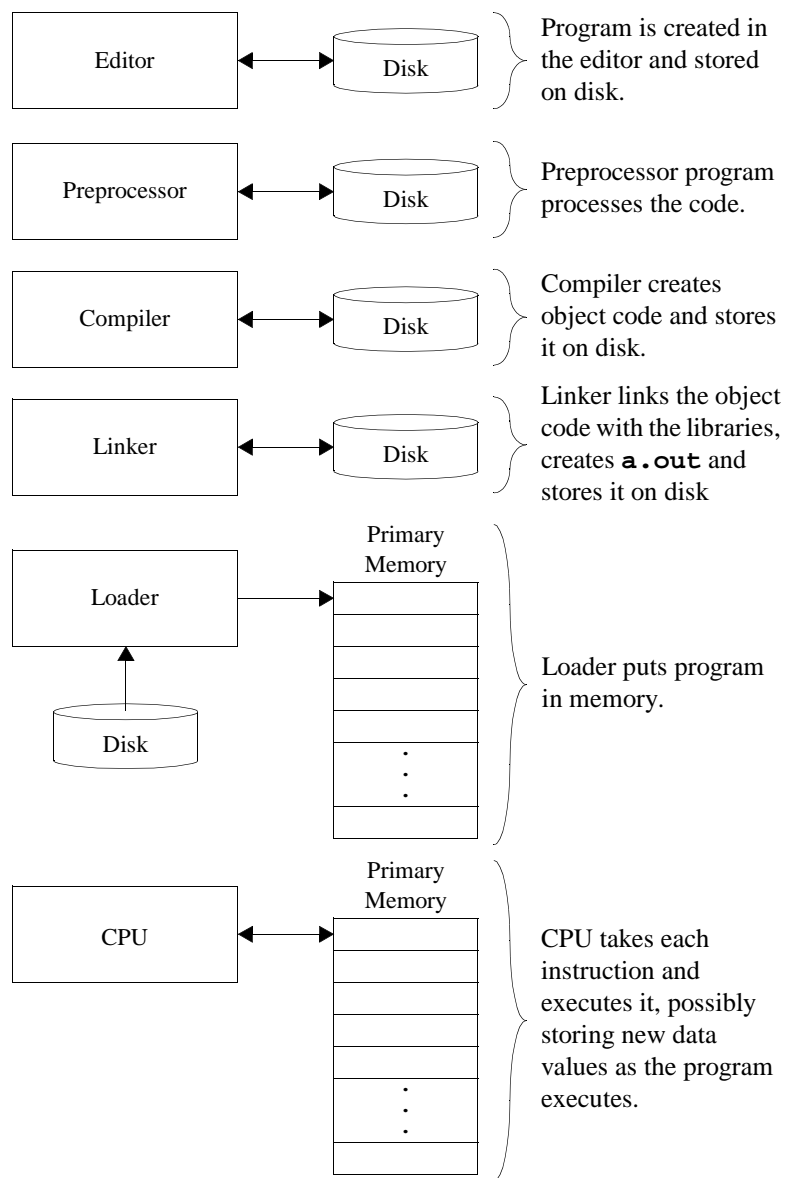


Fig. 1.1 A typical C++ environment.

---

```
1 // Fig. 1.2: fig01_02.cpp
2 // A first program in C++
3 #include <iostream.h>
4
5 int main()
6 {
7     cout << "Welcome to C++!\n";
8
9     return 0;    // indicate that program ended successfully
10 }
```



Welcome to C++!

---

Fig. 1.2 Text printing program.

Escape Sequence	Description
\n	Newline. Position the screen cursor to the beginning of the next line.
\t	Horizontal tab. Move the screen cursor to the next tab stop.
\r	Carriage return. Position the screen cursor to the beginning of the current line; do not advance to the next line.
\a	Alert. Sound the system bell.
\\	Backslash. Used to print a backslash character.
\"	Double quote. Used to print a double quote character.

---

Fig. 1.3 Some common escape sequences.

---

```
1 // Fig. 1.4: fig01_04.cpp
2 // Printing a line with multiple statements
3 #include <iostream.h>
4
5 int main()
6 {
7     cout << "Welcome ";
8     cout << "to C++!\n";
9
10    return 0;    // indicate that program ended successfully
11 }
```




Welcome to C++!

---

Fig. 1.4 Printing on one line with separate statements using **cout**.

---

```
1 // Fig. 1.5: fig01_05.cpp
2 // Printing multiple lines with a single statement
3 #include <iostream.h>
4
5 int main()
6 {
7     cout << "Welcome\nto\nnC++!\n";
8
9     return 0;    // indicate that program ended successfully
10 }
```




```
Welcome
to
C++!
```

---

**Fig. 1.5** Printing on multiple lines with a single statement using **cout**.

---

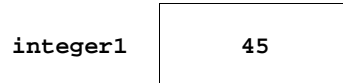
```
1 // Fig. 1.6: fig01_06.cpp
2 // Addition program
3 #include <iostream.h>
4
5 int main()
6 {
7     int integer1, integer2, sum;    // declaration
8
9     cout << "Enter first integer\n"; // prompt
10    cin >> integer1;                // read an integer
11    cout << "Enter second integer\n"; // prompt
12    cin >> integer2;                // read an integer
13    sum = integer1 + integer2;       // assignment of sum
14    cout << "Sum is " << sum << endl; // print sum
15
16    return 0;    // indicate that program ended successfully
17 }
```



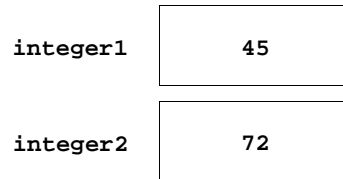
```
Enter first integer
45
Enter second integer
72
Sum is 117
```

---

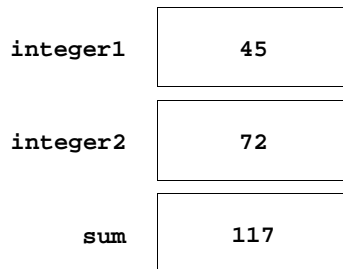
**Fig. 1.6** An addition program (part 2 of 2).



**Fig. 1.7** A memory location showing the name and value of a variable.



**Fig. 1.8** Memory locations after values for two variables have been input.



**Fig. 1.9** Memory locations after a calculation.

C++ operation	Arithmetic operator	Algebraic expression	C++ expression
Addition	+	$f + 7$	<code>f + 7</code>
Subtraction	-	$p - c$	<code>p - c</code>
Multiplication	*	$bm$	<code>b * m</code>
Division	/	$x / y$ or $\frac{x}{y}$ or $x \div y$	<code>x / y</code>
Modulus	%	$r \bmod s$	<code>r % s</code>

**Fig. 1.10** Arithmetic operators.

Operator(s)	Operation(s)	Order of evaluation (precedence)
( )	Parentheses	Evaluated first. If the parentheses are nested, the expression in the innermost pair is evaluated first. If there are several pairs of parentheses “on the same level” (i.e., not nested), they are evaluated left to right.
*, /, or %	Multiplication Division Modulus	Evaluated second. If there are several, they are evaluated left to right.
+ or -	Addition Subtraction	Evaluated last. If there are several, they are evaluated left to right.

Fig. 1.11 Precedence of arithmetic operators.

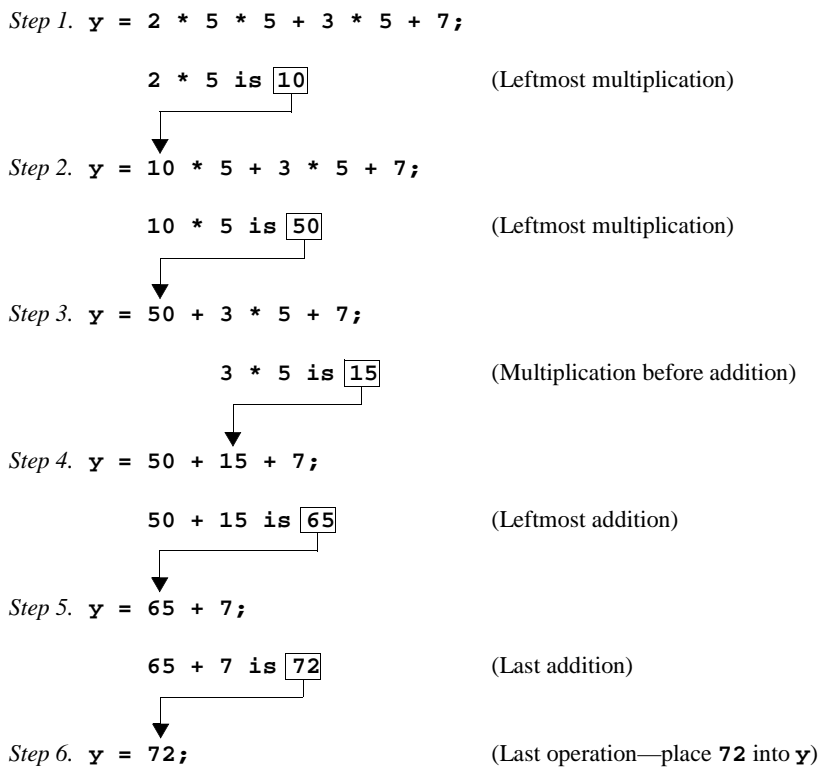


Fig. 1.12 Order in which a second-degree polynomial is evaluated.

Standard algebraic equality operator or relational operator	C++ equality or relational operator	Example of C++ condition	Meaning of C++ condition
<i>Equality operators</i>			
=	==	<b>x == y</b>	<b>x</b> is equal to <b>y</b>
≠	!=	<b>x != y</b>	<b>x</b> is not equal to <b>y</b>
<i>Relational operators</i>			
>	>	<b>x &gt; y</b>	<b>x</b> is greater than <b>y</b>
<	<	<b>x &lt; y</b>	<b>x</b> is less than <b>y</b>
≥	>=	<b>x &gt;= y</b>	<b>x</b> is greater than or equal to <b>y</b>
≤	<=	<b>x &lt;= y</b>	<b>x</b> is less than or equal to <b>y</b>

**Fig. 1.13** Equality and relational operators.

```

1 // Fig. 1.14: fig01_14.cpp
2 // Using if statements, relational
3 // operators, and equality operators
4 #include <iostream.h>
5
6 int main()
7 {
8     int num1, num2;
9
10    cout << "Enter two integers, and I will tell you\n"
11          << "the relationships they satisfy: ";
12    cin >> num1 >> num2;    // read two integers
13
14    if ( num1 == num2 )
15        cout << num1 << " is equal to " << num2 << endl;
16
17    if ( num1 != num2 )
18        cout << num1 << " is not equal to " << num2 << endl;
19
20    if ( num1 < num2 )
21        cout << num1 << " is less than " << num2 << endl;
22
23    if ( num1 > num2 )
24        cout << num1 << " is greater than " << num2 << endl;
25
26    if ( num1 <= num2 )
27        cout << num1 << " is less than or equal to "
28          << num2 << endl;
29
30    if ( num1 >= num2 )
31        cout << num1 << " is greater than or equal to "
32          << num2 << endl;
33
34    return 0;    // indicate that program ended successfully
35 }

```

```

Enter two integers, and I will tell you
the relationships they satisfy: 3 7
3 is not equal to 7
3 is less than 7
3 is less than or equal to 7

```

```

Enter two integers, and I will tell you
the relationships they satisfy: 22 12
22 is not equal to 12
22 is greater than 12
22 is greater than or equal to 12

```

Fig. 1.14 Using equality and relational operators (part 1 of 2).

```

Enter two integers, and I will tell you
the relationships they satisfy: 7 7
7 is equal to 7
7 is less than or equal to 7
7 is greater than or equal to 7

```

Fig. 1.14 Using equality and relational operators (part 2 of 2).

Operators	Associativity	Type
( )	left to right	parentheses
*   /   %	left to right	multiplicative
+   -	left to right	additive
<<   >>	left to right	stream insertion/extraction
<   <=   >   >=	left to right	relational
==   !=	left to right	equality
=	right to left	assignment

Fig. 1.15 Precedence and associativity of the operators discussed so far.




---

1

---

```
1 // Fig. 1.16: fig01_16.cpp
2 // Using new-style header files
3 #include <iostream>
4
5 using namespace std;
6
7 int main()
8 {
9     cout << "Welcome to C++!\n";
10    std::cout << "Welcome to C++!\n";
11
12    return 0;    // indicate that program ended successfully
13 }
```



```
Welcome to C++!
Welcome to C++!
```

---

**Fig. 1.16** Using new-style header files.