Chapter Three
Keyboard Input and Screen Output
第 3 章　键盘输入和屏幕输出

In C++, data stream objects are used to perform basic input and output of data to and from various devices such as the keyboard and the screen. A stream is a data communication object connected to an input or output device.

Just as standard output stream `cout` is automatically associated with the screen, the standard input stream `cin` (console input) is automatically associated with the keyboard. The insertion operator << is used to write data to `cout` and the extraction operator >> is used to read data from the keyboard.

Essentially, reading data from the keyboard is the opposite of writing data out to the screen. The opposite of `cout` is `cin` and the opposite of << is >>.

在C++中，数据流对象用于在各种不同的设备（如键盘和屏幕）上执行基本的数据输入和输出操作。所谓流，就是与输入/输出设备相关联的数据通信对象。

正如标准输出流cout自动与屏幕相关联一样，标准输入流cin自动与键盘相关联。流插入运算符<<用于将数据输出到cout，流提取运算符>>用于从键盘读取数据。

3.1　Simple keyboard input（简单的键盘输入）

The following program reads a number from the keyboard and stores it in the variable `num` (for number).

下面的程序从键盘读入一个数，并将其保存到变量num 中。

Program Example P3A

```
1  // Program Example P3A
2  // Program to demonstrate keyboard input.
3  #include <iostream>
4  using namespace std ;
5
6  main()
7  {
8    int num ;
9
10   cout << "Please type a number: " ;
11   cin >> num ;
12   cout << "The number you typed was " << num << endl ;
13 }
```

This program simply asks the user for a number and displays the entered number on the screen.

Line 10 displays the message:

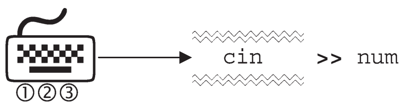
```
Please type a number:
```

Line 11 causes the computer to wait indefinitely until you type a number and press the Enter key.
When you type in a number (e.g. 123) followed by the Enter key, the program continues to line 12 and displays:

```
The number you typed was 123
```

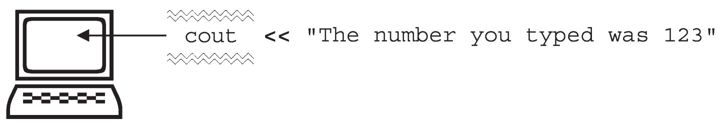
As a memory aid when using `cin`, think of the data as entering the input stream and flowing in the direction indicated by the arrows >> i.e. towards the memory variable.

使用cin 时，可以认为数据是先输入到输入流中，然后按照箭头>>所指的方向流入到内存中的变量里。



Similarly when using cout, think of the data as entering the output stream and flowing in the direction indicated by the arrows << i.e. towards the screen.

类似地，使用cout时，可以认为数据是先输入到输出流中，然后按照箭头<<所指的方向送到屏幕显示。



The next program inputs two floating-point numbers from the keyboard and displays the result of their addition.

Program Example P3B

```
// Program Example P3B
// Program to input two numbers and display their sum.
#include <iostream>
using namespace std ;

main()
{
  float num1, num2 ;

  cout << "Type in 2 numbers. Press Enter after each number." << endl ;
  cin >> num1 >> num2 ;

  float sum ; ←———————— A variable can be defined just before it is used.
  sum = num1 + num2 ;
  cout << num1 << " + " << num2 << " = " << sum << endl ;
 }
```

A sample run of this program is:

```
Type in 2 numbers. Press Enter after each number.
1.1
2.2
1.1 + 2.2 = 3.3
```

Line 11 reads the two numbers from the keyboard and line 14 assigns the result of their addition to the variable sum defined on line 13.

As shown in line 13, a variable can be defined at any point in a program, provided it is defined before it is used.

在程序的任何地方都可以定义变量，只要是在使用之前定义即可。

3.2 Manipulators（流操纵符）

Manipulators are used to modify input and output data streams. The manipulator `endl` was used in previous programs to skip to the start of a new line on the screen.

流操纵符用于对输入和输出数据流进行修改。例如前面程序中使用的流操纵符endl，它的作用是将光标移到一个新行的起始位置。

A list commonly used manipulators is given in appendix E, but this section will look at just five of them: `endl`, `setw`, `setfill`, `fixed` and `setprecision`.

A manipulator can appear anywhere in a series of insertion or extraction operations. For example,

流操纵符可以出现在一个连续的流插入和流提取运算符中的任何位置。

```
cout << endl << endl << "endl can be used anywhere" << endl ;
```

This example will skip to the start of a new line, skip another line, display the text in the double quotes and then skip to the start of the following line.

这个例子是先将光标移到一个新行的起始位置，然后再换到下一行后，输出双引号中的提示信息，最后再将光标移到一个新行的起始位置。

The manipulator `setw` is used to set the width of a data field. The width of a data field is the number of columns that the data item occupies on the screen.

流操纵符setw用于设置数据域的宽度。数据域的宽度指的是数据项在屏幕上所占的列数。

Program Example P3C

```
// Program Example P3C
// Demonstration of the setw manipulator.
#include <iostream>
#include <iomanip>
using namespace std ;

main()
{
  int num1 = 123, num2 = 4567 ;

  cout << "Without setw:" << endl ;
  cout << num1 << num2 << endl ;
  cout << "With setw:" << endl ;
  cout << setw( 4 ) << num1 << setw( 7 ) << num2 << endl ;
 }`
```

The output from this program is:

```
Without setw:
1234567
With setw:
 123   4567
```

Line 4 is required for any manipulator, like setw, that has a value in parentheses. Other manipulators, such as endl, do not require this line. Without using setw, the two numbers are displayed beside each other without any intervening space, making it difficult to see where one number ends and the next number starts.

使用像setw这样的在圆括号中有一个参数的流操纵符时，第4行是必需的。而使用其他流操纵符（如endl）则不需要这一行。

不使用setw时，这两个数就会紧挨在一起显示，中间没有任何空格，这样就很难区分第一个数是在哪里结束的，第二个数是在哪里开始的。

The field width (number of columns) for num1 is set to 4 on line 14 by inserting the manipulator setw(4). Since num1 is only a threedigit number and the field width is set to four, a space precedes the three digits of num1. Similarly the field width for num2 is set to 7, resulting in three spaces preceding its four-digit value.

If the field width is set too small to display a value, the width is automatically expanded so that all the digits in the value are displayed. The manipulator setfill is used to change the "padding" character from a space to any other character. This is demonstrated in the next program.

如果域宽设置得太小不足以显示一个数值，那么系统会自动调整域宽使其能够显示数据的所有位的数字。

流操纵符setfill用于把占位符从空格改变为其他字符。

Program Example P3D

```
1  // Program Example P3D
2  // Demonstration of the setfill manipulator.
3  #include <iostream>
4  #include <iomanip>
5  using namespace std;
6
7  main()
8  {
9    double num = 123.456;
10
11   cout << setw( 9 ) << setfill( '*' ) << num << endl;
12   cout << setw( 9 ) << setfill( '0' ) << num << endl;
13   cout << setw( 10 ) << num << endl;
14}
```

The output from this program is:

```
**123.456
00123.456
000123.456
```

Unlike setw, which applies only to the next data item in the output stream, the setfill manipulator remains in effect for all subsequent data items sent to the output stream. This is shown in line 13 where the fill character remains '0', as set in line 12.

流操纵符setw只对输出流中的下一个数据项起作用，而setfill将对送入输出流中的所有后继数据项都起作用。

The manipulator setprecision is used to specify the number of digits of a number to display. There are two ways of using this manipulator as shown in the next program.

流操纵符setprecision用于指定要显示的数据的位数。

Program Example P3E

```
1  // Program Example P3E
2  // Program to demonstrate the setprecision and fixed manipulators.
3  #include <iostream>
4  #include <iomanip>
5  using namespace std ;
6
7  main()
8  {
9    double num = 123.45678 ;
10
11   cout << num << endl ;
12   cout << setprecision( 7 ) << num << endl ;
13   cout << fixed << setprecision( 2 ) << num << endl ;
14 }
```

The output from this program is:

```
123.457
123.4568
123.46
```

By default, the maximum number of digits displayed for a number is six, which includes digits before and after the decimal point. On line 11, num is rounded up so that there is a total of six digits displayed.
In line 12, num is rounded up so that there is a total of seven digits displayed.
In line 13, the manipulator fixed precedes the setprecision specification. In this case setprecision refers to the number of digits after the decimal point. Line 13 therefore displays the value of num rounded up to two places of decimals.
Both manipulators fixed and setprecision remain in effect for subsequent insertions into the output stream.

默认情况下，数据显示的最大位数（包括小数点之前和小数点之后）是6。执行第11行时，系统对num 进行了舍入处理，以便只显示6位数字。
在第12行，对num进行了舍入处理，只显示7位数字。
在第13行，流操纵符fixed放置在setprecision说明之前。这时，setprecision指定的是小数点后面的显示位数。因此，第13行显示的num的值只保留了两位小数。
流操纵符fixed和setprecision将一直对输出流中的后继数据项起作用。

3.3 Single-character input and output （单个字符的输入和输出）

Unlike pressing keys such as A, B or C on the keyboard, pressing keys such as Tab, Enter and the space bar do not display anything on the screen. These keys generate an invisible blank or white space on the screen and are consequently called whitespace characters. (Of course it is only when the background is white that white spaces are displayed. If the background is black then black spaces are generated. Nevertheless, they are still called whitespace characters.)
The following code segment inputs a character from the keyboard to the variable ch, ignoring whitespace characters.

与在键盘上按下A、B、C这样的键不同的是，按下诸如Tab、Enter和空格键不会在屏幕上显示任何字符。因此，这些键在屏幕上产生的显示结果是不可见的空白或空格，称为空白字符（当然，仅当背景是白色时，才显示白色的空格。而当背景是黑色时，产生的是黑色的空格。尽管如此，它们仍然被称为空白字符）。

```
char ch ;
```

```
cin >> ch ; // Reads the next character,
            // whitespace characters are ignored.
```

Sometimes it is required to read a single character from the keyboard regardless of whether it is a whitespace character or not. This can be done by using the manipulator `noskipws` (<u>no</u> skip <u>w</u>hitespace).

有时，需要从键盘读入单个字符，不管它是否是空白字符。这时就要使用流操纵符noskipws（即no skip whitespace）来实现。

```
cin >> noskipws >> ch ; // Reads the next character,
                        // a whitespace character
                        // may be read.
```

Alternatively, the function `get()` associated with the input stream object `cin` can be used.

另一种方法是将函数get( )与输入流对象cin联合使用。

```
cin.get( ch ) ; // Also reads the next character,
                // a whitespace character may be read.
```

A function is a block of program code that carries out a specific task. Functions that are associated with an object are called *member functions* of the object.

函数是执行特定功能的程序模块。与一个对象关联的函数称为对象的成员函数。

In this case the function is pre-written and is available for use by a programmer. Chapters 7 and 8 show how to write functions and member functions.

In addition to displaying a character using >>, the output stream object `cout` has a member function `put()` that can be used to display a character.

除>>之外，还可以利用输出流对象cout的成员函数put( )来显示一个字符。

```
cout.put( ch ) ; // Display the character ch.
```

Programming pitfalls

1. Do not mix up the insertion operator << and the extraction operator >>. The insertion operator is used to insert data into the output stream; the extraction operator is used to read data from the input stream.

2. Some manipulators apply only to the next data field (e.g. `setw`); others (e.g. `setprecision`) stay in effect for all subsequent data fields.

3. The line

   ```
   #include <iomanip>
   ```

   is required for a manipulator that has a value in parentheses, e.g. `setw( 4 )`. Other manipulators, such as `endl`, do not require this line.

1. 不要混淆流插入运算符（<<）和流提取运算符（>>）。流插入运算符用于向输出流中插入数据，而流提取运算符用于从输入流中读取数据。
2. 某些流操纵符只作用于下一个数据域（例如setw）；而另外一些流操纵符（例如 setprecision）对所有的后继数据域都起作用。
3. 在使用圆括号中有一个参数的流操纵符时，例如setw(4)，一定要加上“include<iomanip>”这行代码。对于其他流操纵符（例如endl），则不需要加入这一行。

Quick syntax reference

	Syntax	Examples

Input data from the keyboard	`cin >> variable1`
	`    >> variable2`
	`... >> variablen;`
		`int num1, num2; float num3;`
		
		`cin >> num1 >> num2 >> num3;`

Input a single character from the keyboard	`cin.get( variable );`	`char char_in;`
		`cin.get( char_in );`

Output a single character to the screen	`cout.put( variable );`	`char char_out;`
		`cout.put( char_out );`

Set the width of a field	`setw( integer )`	`float num;`
		`cout << setw( 5 ) << num;`

Set the number of decimal places	`fixed << setprecision( integer )`	`cout << fixed`
		`<< setprecision( 2 )`
		`<< num ;`

Set a fill character	`setfill( character )`	`cout << setw( 5 )`
		`<< setfill('0')`
		`<< num ;`

Exercises

1. Write a program to input four numbers and display them in reverse order.
2. Write a program that inputs a number of hours and displays the equivalent number of weeks, days and hours. For example, an input of 553 should display 3 weeks, 2 days and 1 hour.
3. Assuming the human heart rate is seventy-five beats per minute, write a program to ask a user their age in years and to calculate the number of beats their heart has made so far in their life. Ignore leap years.（leap years：闰年）

4. Write a program to accept a temperature in degrees Fahrenheit and convert it to degrees Celsius. Your program should display the following prompt:

```
Enter a temperature in degrees Fahrenheit:
```

   You will then enter a decimal number followed by the Enter key.
   The program will then convert the temperature by using the formula

   Celsius = (Fahrenheit − 32.0) * (5.0 / 9.0)

   Your program should then display the temperature in degrees Celsius using an appropriate message.
5. Make changes to the program in exercise 4 to accept the temperature in degrees Celsius and convert it to degrees Fahrenheit.
   (Fahrenheit：华氏温度；Celsius：摄氏温度)
6. Write a program to accept a distance in kilometres and display the equivalent distance in miles.
   (1 mile = 1.609 344 kilometres.)
7. Write a program to input three floating-point numbers from the keyboard and to calculate
   (a) their sum and
   (b) their average.
   Display the results to three decimal places.
8. Write a program to input a year and display the century that year is in.
   For example, if the input is 2016 then the ouput should be 21.
9. Write a program to read in two numbers from the keyboard and to display the result of dividing the second number into the first.
   For example, if the input is 123 and 12, the result should be displayed in the following format:

```
123 divided by 12 = 10 Remainder = 3
```

   (Hint: use the modulus operator % to get the remainder 3, and use integer division to get the quotient 123.)(modulus operator：求余运算符)