目 录

第 1 章　C++ 语言概述 / 1

第 2 章　数据类型、表达式和基本运算 / 12

第 3 章　流程控制语句 / 36

第 4 章　数组、指针与引用 / 70

第 5 章 函 数 / 109

第 6 章 类和对象 / 156

第 10 章　输入输出流 / 270

第 1 章 C++ 语言概述

1.1 认识 C++ 语言

程序设计是编写计算机程序的过程，是将人类的自然语言所描述的问题及解决问题的方法和步骤转化为用计算机语言来描述的过程。

1.1.1 程序设计方法

程序一般由算法和数据两方面构成。算法就是解决问题的方法和步骤。算法在数据上操作，以提供问题的解决方案。纵观计算机发展的历史，这两个方面（算法和数据）一直保持不变，发展演化的是它们之间的关系，就是所谓的程序设计方法。目前常用的程序设计方法主要包括结构化程序设计（structured programming）、面向对象程序设计（object-oriented programming，OOP）和泛型程序设计（generic programming）。

1. 结构化程序设计

结构化程序设计是以功能为中心，基于功能分解的程序设计方法。一般采用自顶向下、逐步求精的方法，将一个复杂的系统功能逐步分解成由许多简单的子功能构成，然后分别对子功能进行编程实现。一个程序由一些子程序构成，每个子程序对应一个子功能，实现了功能抽象。子程序描述一系列的操作，是操作的封装体。结构化程序的执行过程体现为一系列子程序的调用。在程序中，数据处于附属地位，它独立于子程序，在调用子程序时，数据作为参数传递给子程序使用。

可以用一个式子来描述结构化程序的本质特征：程序 = 算法 + 数据结构。式中的“算法”是对数据加工步骤的描述，而“数据结构”是对算法所加工的数据的描述。早期的程序大都采用了结构化程序设计方法，其不足之处如下：数据与操作分离，缺乏对数据的保护；功能会随着需求的改变而变化，而功能子程序的重新设计往往会导致整个程序结构的变动，使得程序难以维护；子程序往往是针对某个应用程序而设计的，它们很难用于其他的应用程序，导致难以重复使用（简称复用）。复用往往以一个一个的子程序为单位。

2. 面向对象程序设计

20 世纪 70 年代，由于软件危机的出现，结构化程序设计越来越不能满足大型程序设计的要求，程序设计的焦点从结构化程序设计方法转移到抽象数据类型的程序设计上，现在通常称为面向对象程序设计。一个面向对象的程序由一些对象构成，对象是由一些数据及可施于这些数据的操作所构成的封装体，对象的特征由相应的类来描述，一个类可以从其他的类继承。面向对象程序的执行过程体现为各个对象之间互相发送和处理消息。面向对象的程序描述如下：程序 = 对象 / 类 + 对象 / 类 +…。其中，对象 / 类 = 数据 + 操作。

在面向对象程序设计中，把数据和对数据的操作封装在一起，对数据的操作必须通过相应的对象来进行，从而加强了数据的保护。对象在问题的求解领域是相对稳定的实体，由对象构成的程序能够适应软件需求的变化。在面向对象程序设计中软件的复用以类为单位。

泛型程序设计本书不做详细介绍，有需要的读者可自行参阅相关资料。

1.1.2 程序设计语言

在日常生活中，人们用自然语言进行相互之间的交流。然而当人们使用计算机来解决实际问题时，就需要使用计算机能够“理解”的语言，这类语言统称为计算机程序设计语言。程序设计语言的发展经历了机器语言、汇编语言和高级语言三个阶段。机器语言是用二进制代码表示的语言；汇编语言是采用助记符表示的语言；高级语言是与自然语言较为接近的语言，提高了编程效率，改进程序的可读性、可维护性，因此得到普遍使用。高级语言的种类很多，如 Fortran、Basic、Pascal、Java、C 和 C++ 等。虽然各种高级语言都有自己的特点和应用范围，但是它们在表达和功能体现等方面所产生的共性多半是相通的。因此努力学好并掌握一门程序设计语言，往往可以触类旁通，不仅可以更好地驾驭计算机，也可以为以后充分挖掘自身的潜能和提高自身的能力打下基础。

1.1.3 C++ 语言的发展

C++ 是当今非常流行的一种既支持结构化程序设计又支持面向对象程序设计的高级程序设计语言。它适合作为系统描述语言，既可用来写系统软件，也可用来写应用软件。它是 20 世纪 80 年代初由美国贝尔实验室（Bell Laboratory）在 C 语言的基础上开发的。

C 语言最早的原型是 Algol 60。1963 年，剑桥大学将其发展成为 CPL（combined programming language）。1967 年，剑桥大学的马丁·理察德（Matin Richards）对 CPL 语言进行了简化，产生了 BCPL（basic combined programming language）。1970 年，美国贝尔实验室的肯尼斯·蓝·汤普森（Kenneth Lane Thompson）对 BCPL 进行了修改，并取名为 B 语言，提取 CPL 的精华，并用 B 语言编写了第一个 UNIX 系统。1973 年，AT&T 贝尔实验室的丹尼斯·里奇（Dennis Ritchie）在 BCPL 和 B 语言的基础上设计出了一种新的语言，取 BCPL 中的第二个字母为名，这就是大名鼎鼎的 C 语言。随后不久，UNIX 的内核（kernel）和应用程序全部用 C 语言改写，从此，C 语言成为 UNIX 环境下使用最广泛的主流编程语言。

C 语言是一种常规用途的语言，可用来编写任何形式的程序，而 UNIX 操作系统直接促成了它的成功与普及。如果想维护 UNIX 操作系统，就需要使用 C 语言。C 语言和 UNIX 操作系统的配合是如此的天衣无缝，以至于不久以后，不仅系统程序，就连 UNIX 下运行的几乎所有商业程序都开始用 C 语言来编写。随着 C 语言的流行逐渐出现了为其他流行操作系统编写的 C 语言版本，C 语言的应用开始不受 UNIX 机器的局限。不过，虽然 C 语言非常流行，但它并不是完美无缺的。

C 语言的特殊性在于它虽然是一种高级语言，但也包含了低级语言的许多特点。C 语言其实处在一种非常高级的语言和一种低级的语言之间，其优点和缺点都很突出。类似于汇编语言，C 语言程序可直接操纵计算机的内存；此外，C 语言又具有高级语言的许多特点，所以比汇编语言更容易理解和编写。这使得 C 语言成为编写系统程序的理想选择，但对于其他程序（有时甚至包括一些系统程序），C 语言并不像其他语言那样容易理解。另外，它也不像另外一些高级语言那样提供了对自动检查的完美支持。

为了解决上述问题以及 C 语言的另外一些缺陷，AT&T 贝尔实验室的本贾尼·斯特劳斯特卢普（Bjarne Stroustrup）在 20 世纪 80 年代初开发了 C++ 语言。C++ 并不是对 C 语言的功能做简单的改进和扩充，而是

一种本质性革新。C 语言的大多数特性都成为 C++ 的一个子集，所以大多数 C 程序其实也是 C++ 程序（反之则不成立，许多 C++ 程序都绝对不是 C 程序），这对于继承和开发当前已广泛使用的软件是非常重要的，可节省大量的人力和物力。和 C 语言不同，C++ 具备了“面向对象程序设计”的能力。OPP 是一种功能非常强大的编程技术，可以使得程序的各个模块的独立性更强，程序的可读性和可理解性更好，程序代码的结构性更加合理。这对于设计和调试一些规模大的软件是非常重要的。再者，用 C++ 设计的程序具有扩充性强的特点，这对于编写一些大的程序而言是非常重要的。

1.1.4　C++ 语言的特点

C++ 是 C 语言的超集。C++ 完全兼容 C 语言，具有 C 语言的简洁、紧凑、运算符丰富、可直接访问机器的物理地址、使用灵活方便、程序书写形式自由等特点。大多数 C 语言程序代码略做修改或不做修改即可在 C++ 继承环境下运行。

C++ 是一种面向对象的程序设计语言。C++ 作为一种面向对象的程序设计语言，它使程序的各个模块间更具独立性，程序的可读性更好，代码结构更加合理，设计和编制大型软件更为方便。

C++ 代码简洁高效、可移植性强。

1.2　C++ 语言的词汇

C++ 语言的词汇由字符集、标识符、关键字构成。

1.2.1　字符集

任何一种语言都是由一些基本符号构成的，这些基本符号的集合就构成了相应语言的字符集。如汉语中，一个个独立的汉字通过一定的语法（词法）构成词，然后通过各个词构成句子，通过句子组成文章。这里的单个的汉字就是汉语中的一个个字符，所有的汉字构成汉语的字符集。计算机编程语言与此类似，也是由语法所允许的单个的字符构成词，然后由词构成语句，由语句构成源程序。C++ 语言的字符集由 26 个大小写英文字母、10 个数字字符以及一些特殊字符构成，具体如下：

英文字母：A~Z、a~z。

数字字符：0~9。

特殊字符：空格 ! # % ^ & * _ + = - ~ < > / \ . , : ? ‘ ’ “ ” ( ) [ ] { }。

1.2.2　标识符

标识符是字符集中的字符按照一定规则构成的具有一定意义的最小语法单位。程序中的一些实体，如前面提到的变量、函数，都需要一个名字来标识它们，这些实体的名字就叫标识符，C++ 中标识符的构成具有以下规则。

标识符只能由字母、数字和下划线 3 类字符构成。

第一个字符只能是英文字母或下划线（_），不可以以数字字符打头。如 Abc、X1、x1、_x1、x2、x3、desk、books 等都是合法的标识符，而 5you 不是合法的标识符，但 you5、You5、_you5 都是合法的标识符。

在 C++ 中，字母的大小写是有区别的。如 x1、X1 是不同的标识符。

标识符不可以和关键字同名（因为关键字具有特殊的用途），否则会使编译器产生混淆。

注意：标识符的长度往往会受具体的编译器的限制。给变量、函数等起名字（标识符）时，最好使名字具有意义，以提高程序的易读性。

1.2.3 关键字

关键字是事先规定的、有特殊用途的标识符，它是 C++ 语言预先定义的词法符号，不能在程序中用作其他用途，它们的意义以后会逐渐介绍，C++ 常用的关键字如表 1-1 所示。

表 1-1 C++ 常用的关键字

asm	auto	bool	break	case	catch
char	class	const	const_cast	continue	default
delete	do	double	dynamic_cast	else	enum
explicit	export	extern	false	float	for
friend	goto	if	inline	int	long
mutable	namespace	new	operator	private	protected
public	register	reinterpret_cast	return	short	signed
sizeof	static	static_cast	struct	switch	template
this	throw	true	try	typedef	typeid
typename	union	using	unsigned	virtual	void
while	volatile	wchar_t	—	—	—

1.3 C++ 程序的开发步骤

C++ 语言程序开发主要包括编辑、编译、连接、运行 4 个步骤。

1.3.1 编辑

程序是计算机系统能够识别和执行的一组指令。每一条指令都指挥计算机执行特定的操作。用高级语言编写的程序称为源程序。C++ 的源程序是以“.cpp”为后缀的文本文件。在编程时，我们利用某个编辑程序，把 C++ 源程序代码输入计算机中，并以“.cpp”为后缀保存到外存中。还有一种后缀为“.h”的源文件，称为头文件（header files），用来说明其他函数库提供的功能或者由编程人员进行类定义。有些头文件是不带后缀的，如 iostream 文件。

1.3.2　编译

计算机的 CPU 只能识别由 0 和 1 组成的二进制机器语言指令，而不能识别使用高级语言编写的指令，如 c=a*b；因此，必须利用 C++ 编译器将编辑好的源程序转换成二进制机器代码形式的目标文件，这个过程就是编译，目标文件的文件名通常为“*.obj”。

1.3.3　连接

一个 C++ 程序中的代码可以分别放到多个源文件中，而每个源文件都是分别编译生成目标文件的，因此为了得到一个完整的可执行程序，必须通过一个连接程序把这些目标文件以及程序中用到的一些系统功能或其他第三方提供的功能所在的目标文件连接起来，最终形成一个可执行的二进制文件，可执行文件的后缀通常为“*.exe”。

1.3.4　运行

运行是让操作系统将某个可执行文件装入内存，运行其中的可执行代码，从而得到输出的结果。

在编辑、编译、连接、运行的过程中都有可能发现程序错误，如编译程序发现源文件中有语法错误，连接程序发现某个使用到的外部库函数不存在，运行程序的结果与预期的不一致，等等。这些都会导致程序员要返到前面的步骤中进行修改，然后从改正错误的步骤重新开始调试运行，这个过程可能需要重复多次，直到程序产生正确的结果为止。一般在编译和连接时，系统给出的出错信息分为两种，一种是错误（error），另一种是警告（warning）。警告是一些可能会影响到后续操作的轻微的错误（如定义了一个变量，但是一直没有使用），但编译连接操作可以成功。应该尽量减少警告信息，因为它表示你的程序存在一定的隐患，若出现错误，则编译连接操作将失败，必须重新编辑源程序。

1.4　Visual C++ 2017 集成开发环境

Visual C++ 2017 是用来编写 C++ 程序的主要工具。

首先要创建项目，如图 1-1 所示，单击“文件”菜单中的“新建”，选择“项目”，或者直接按“Ctrl+Shift+N”组合键，都会弹出如图 1-2 所示的对话框。

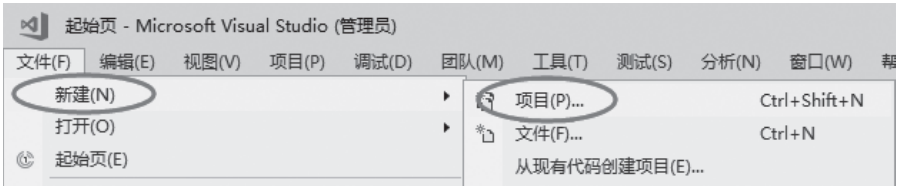

图 1-1　创建项目

图 1-2　新建空项目

选择“空项目”，填写好项目名称，选择好存储路径，可取消勾选“为解决方案创建目录”，单击“确定”按钮即可。

注意：这里一定要选择“空项目”而不是“ Windows 控制台应用程序”，因为后者会导致项目中自带很多莫名其妙的文件，不利于对项目的理解。另外，项目名称和存储路径中最好不要包含中文。

单击“确定”按钮后，会直接进入如图 1-3 所示的项目可操作界面，我们将在这个界面完成所有的编程工作。

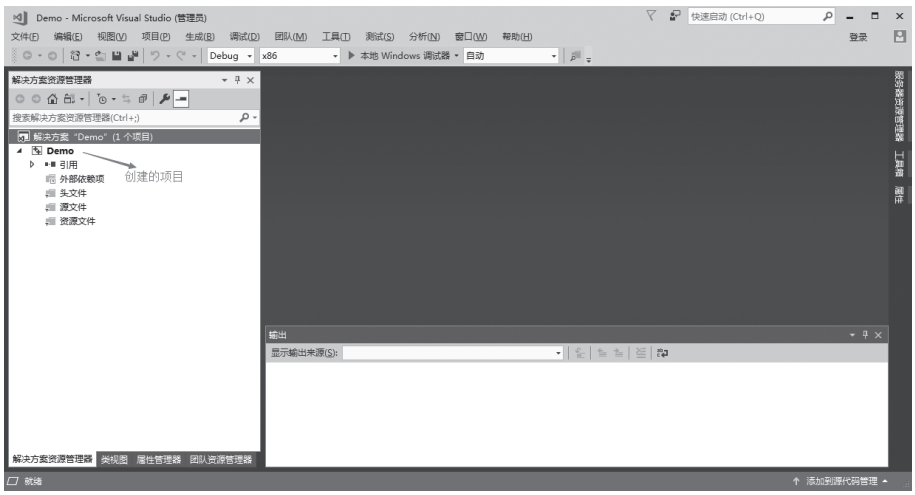

图 1-3　创建好的项目

在“源文件”处右击鼠标，在弹出的菜单中选择“添加→新建项”，如图 1-4 所示。

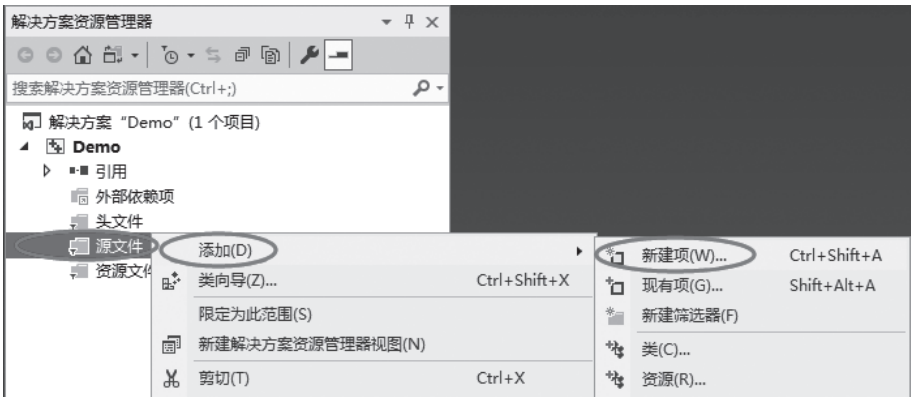

图 1-4　添加新文件

单击“C++ 文件”，输入名称和选择存放路径，最后单击“添加”，如图 1-5 所示。

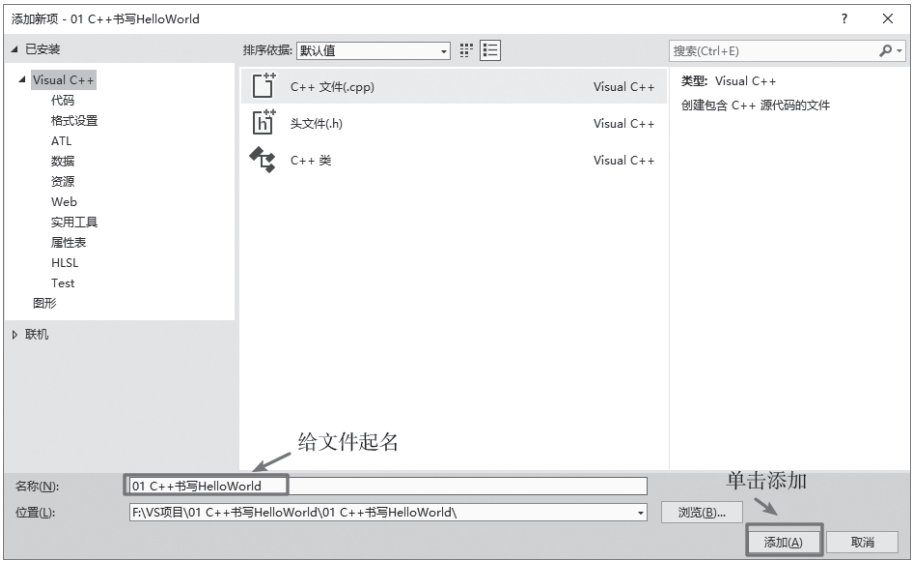

图 1-5　添加源文件

在创建好 C++ 文件之后，就可以进行程序的编写，每种语言都有其特定的语言格式，书写的程序需要满足其格式才能正确地运行。

1.5　简单 C++ 语言程序的构成

下面先介绍几个简单的 C++ 程序，然后从中分析 C++ 程序的特点和构成。

【例 1-1】输出“hello，world！”。代码如下：

```
/* 输出 hello，world！ */
#include <iostream>
using namespace std;
int main（ ）
{
    cout << "hello,world!" << endl;
    return 0;
}
```

运行结果如图 1-6 所示。

图 1-6　运行结果（例 1-1）

程序分析：

（1）“/*”和“*/”之间的所有行都被认为是注释，它对程序的行为没有任何影响，编程人员可以用它们在代码中包含简短的解释或说明。本例中的注释是关于程序文件内容的一个简要描述。

（2）“#include <iostream>”是以“#”开头的预编译指令行，#include < 文件名 > 通常用于将其他代码功能包含到当前程序中。在本例中，指令“#include<iostream>”告诉预处理程序把“iostream”标准文件包含进来。在这个特殊的文件（iostream）中包含 C++ 标准输入 / 输出的功能，本例用到了它的 Cout（标准输出）来实现输出功能。

（3）“using namespace std;”告诉编译器包含 std 名称空间中的功能。C++ 标准库中的所有功能都被包含在 std 名称空间中。因此为了使用这些功能（如“#include<iostream>”引入的输出功能），在使用标准库的 C++ 程序里这一行是经常出现的。

（4）“main”是主函数的名字，每个 C++ 程序都必须包含一个 main 函数，所有 C++ 程序的起始运行点都是从 main 开始。一个函数由两个部分组成：函数头（如“int main（ ）”）和函数体（由“{}”括起来）。

（5）“cout << "hello,world!" << endl;”的功能是向标准输出设备（通常为显示器）输出一行字符。“cout”表示 C++ 中的标准输出流，双引号内的字符串被原样输出，“endl”是换行符，即在输出“hello,world!”后让光标换到下行的行首。“cout”是在 std 名称空间中定义的，定义的代码在 iostream 标准文件中，这就是为什么要在代码中包含 iostream 文件并且声明将要使用这个特殊名称空间的原因。

（6）“return 0;”表示 main 函数将要结束，同时返回一个值给调用本程序的操作系统。main 函数的返回代码“0”通常被理解为程序在运行期间没有任何错误并按照预先设计的那样工作，这是结束一个 C++ 程序时最常用的方法。

【例 1-2】从键盘输入两个整数，并输出这两个整数的和。代码如下：

```
#include <iostream>
using namespace std;
int main ( )
{
    int x, y;    // 定义两个整型变量
    cout << " 请输入两个整数，用空格分隔 , 按回车键（Enter）："<<endl;
    cin >> x >> y; // 从键盘输入两个整数
    cout << x<<"+"<<y<<"="<< x + y<<endl;   // 输出字符串和两个整数的和
    return 0;
}
```

运行结果如图 1-7 所示。

图 1-7　运行结果（例 1-2）

程序分析：

（1）“int x, y;”表示定义两个变量 x 和 y，即系统分配两块内存，并把这两块内存分别取名为 x 和 y。

（2）“//”（双反斜杠）用来表示注释，从“//”开始到本行结束的所有的字符都是注释。

（3）“cin”表示 C++ 的输入流，“cin>>x>>y;”表示从标准输入设备（通常为键盘）中接收两个整数到变量 x、y 中。

（4）“cout << x<<"+"<<y<<"="<< x + y<<endl;”表示连续输出字符串和表达式“$x+y$”的值到标准输出设备。

【例 1-3】从键盘输入两个整数，利用独立的函数求这两个数的和，并输出。代码如下：

```
#include <iostream>
using namespace std;
int add（int x, int y）
{
    return（x + y）;
}
int main ( )
{
    int x, y, sum; // 定义 3 个整型变量 x、y、sum
    cout << " 请输入两个整数，用空格分隔，按回车键（Enter）："<<endl;
    cin >> x >> y; // 从键盘获取两个整数
    sum = add（x, y）; // 利用函数 add 求 x、y 的和
    cout << x << "+" << y << "=" << sum << endl;
    return 0;
}
```

运行结果如图 1-8 所示。

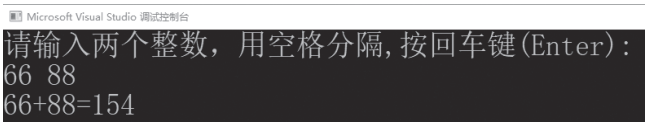

图 1-8　运行结果（例 1-3）

程序分析：

（1）本例定义了一个函数“add”，在 C++ 中，一个函数可以看作是完成某个独立功能的代码的集合。add 函数由函数头“int add（int *x*, int *y*)”和其后由大括号括起来的函数体组成，函数头指定了函数的返回值类型（本例为整数 int)、函数名（本例为 add)、函数需要的参数以及类型（本例为两个整数 *x*、*y*)。

（2）main 中的“sum=add（*x*, *y*）;”引用了 add 函数的功能来实现两个数求和。其执行的顺序：先将 main 中 *x*、*y* 的值交给 add 中的 *x*、*y*，然后执行 add 中的代码完成两个整数的相加，返回的和存放到 sum 对应的内存空间中。

由以上 3 个例子可以看出，C++ 程序的基本构成如下：

（1）逻辑上，一个 C++ 程序由一些函数、类、全局变量 / 对象构成，一个可执行的 C++ 程序必须有且仅有一个 main 函数，此函数是 C++ 程序的入口，即第一个执行的函数。

（2）C++ 函数由函数头和函数体构成，函数头用来说明如何使用函数，函数体用来包含实现函数功能的代码。

（3）物理上，一个 C++ 程序可以放在一个或多个 C++ 源文件（后缀名为“.cpp”的文件）中，每个源文件都可以包含许多函数、类、全局变量、对象的定义等。

（4）在一个源文件中，如果要使用其他的功能，应当用“#include”将该功能包含进来。

1.6　案例实践

利用所学 C++ 程序知识，尝试编写以下程序。

1. 案例说明

编写一个 C++ 程序，输出以下信息：

```
**************************
江苏人民欢迎您!
**************************
```

2. 编程思路

本程序有如下两个重点：

（1）要求输出字符串，需要使用标准库中的输出功能，因此需要将 C++ 的 iostream 头文件包含进来，同时利用 using 语句来使用 std 名称空间。

（2）使用 3 个 cout 语句来分别输出 3 行。每行输出的最后都要让光标换行，同时要注意“江苏人民欢迎您!”前面需要插入空格。

3. 程序代码

所编写的 C++ 程序代码如下：

```
/* 本程序输出指定的信息 */
#include <iostream> // 引入标准库输入输出头文件
using namespace std; // 使用 std 名称空间
int main ( )
{
    cout << "****************************\n";
    cout << " 江苏人民欢迎您！ \n";
    cout << "****************************\n";
    return 0;
}
```

运行结果如图 1-9 所示。

图 1-9　运行结果（案例实践）

本章小结

本章首先介绍了程序设计方法和程序设计语言的发展历程，然后介绍了 C++ 语言的发展历史，分析了 C++ 程序的基本构成，并详细地介绍了 Visual C++ 2017 中开发简单的 C++ 程序的步骤。本章需要掌握的知识点如下：

（1）程序设计语言的发展经历了低级语言到高级语言的过程。

（2）C++ 是 20 世纪 80 年代初由 AT&T 贝尔实验室在 C 语言的基础上借鉴其他面向对象程序设计语言的特性而开发的。它是当今非常流行的一种可以支持结构化、面向对象、泛型等程序设计方法的程序设计语言，它既可用于编写系统软件，也可用于编写应用软件。

（3）C++ 程序由函数和类构成，任何一个 C++ 程序都必须包含一个名为 main 的主程序，它是 C++ 程序执行的起点。

（4）C++ 语言中的字符集是指所有可用的字符的集合，包括 52 个大写和小写的字母，10 个数字字符以及其他一些特殊字符。标识符是指给函数、变量、类等起的名字，C++ 语言中的标识符必须符合一定的规则。关键字是一种有特殊用途的标识符，它是 C++ 语言预先定义的词法符号，不能在程序中用作其他用途。

（5）开发 C++ 程序时，需要编辑、编译、连接、运行 4 个步骤，这 4 个步骤中的任何一个步骤都可能出现错误，此时必须退回到前面的步骤中修改，然后重新开始。因此 C++ 程序的开发是一个不断反复进行编辑、编译、连接、运行的过程，直到获得预期的结果。

（6）Visual C++ 2017 是目前 Windows 操作系统中开发 C++ 程序的一个常用的集成开发环境，使用 Visual C++ 2017 进行 C++ 程序开发可以节省开发时间，提高开发效率。

巩固练习