第一章

C++语言入门

张琪曼和楚继光终于考入了梦想中的圣魔法学院，开学的第一天，他们就看到了信息学奥赛魔法战队招生的公告，酷爱计算机的他们当然不会放弃啦，因为信息时代的魔法师们早已借助于计算机编程来实现强大的魔法了。可是，参加信息学奥赛魔法战队并不是一件很容易的事，他们必须要在短短半年内从一个普通的魔法学徒升级到高级魔法师。他们能做到吗？

我的第一个程序

奥赛魔法战队日常训练用的编程软件（编译器）是 Bloodshed Dev-C++，版本号为 4.9.9.2（Win7 以上操作系统需下载更高版本），它是一个可视化集成开发环境，可以实现 C 或 C++程序的编辑、编译、调试和运行等功能。

Dev-C++的安装方法很简单，鼠标双击运行网站下载的 devcpp 安装软件，如图 1.1 所示。

下载资源中还提供了 devpas 安装程序和代码管理软件安装程序，其中 devpas 安装程序是 pascal 语言的编译软件（非官方指定用比赛软件）。代码管理软件是本书作者自行开发的基于数据库管理的源代码管理工具，可有效分类、管理 Java、C 或 C++、Pascal 等各类源代码，并可对各类源代码进行简单的编辑、编译和运行。

当出现选择语言时，默认选择“English”即可（此处无法选择 Chinese/中文），如图 1.2 所示。

图 1.1

图 1.2

单击 I Agree 按钮接受软件的许可证协议，如图 1.3 所示。

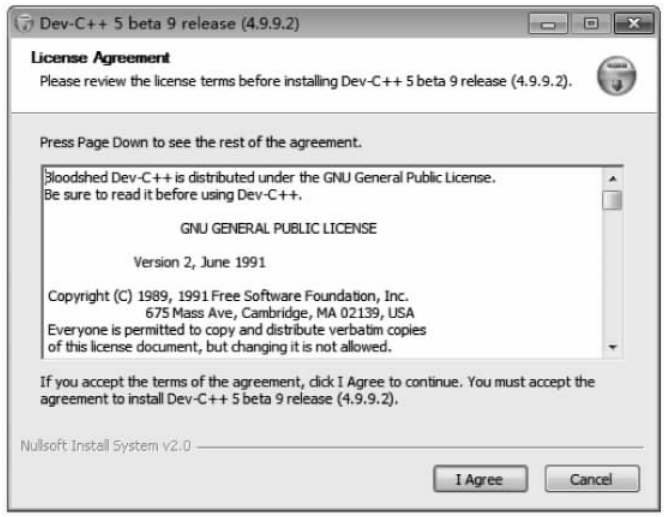


图 1.3

单击 Next 按钮默认安装全部内容，如图 1.4 所示。

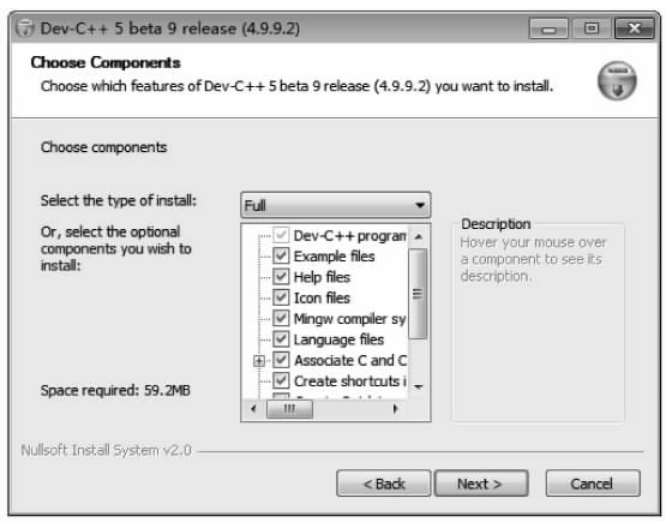


图 1.4

选择安装路径后，单击 Install 按钮安装，如图 1.5 所示。

如无特殊要求，此处建议选择默认安装路径安装，这样下载资源中附送的单机评测软件就可以正常使用了。

安装过程中会有一系列的设置选择，一般选择默认（此处可选择界面为中文）即可。安装完毕后，在桌面双击图标或者在“开始”菜单里选择 Dev-C++，即可启动 Dev-C++集成开发工具，如图 1.6 所示。

双击打开 Dev-C++的编辑界面如图 1.7 所示。

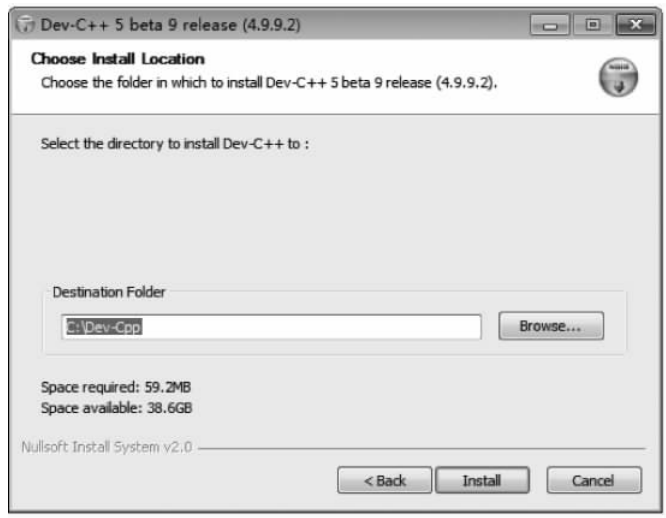

Dev-C++ 5 beta 9 release (4.9.9.2)
Choose Install Location
Choose the folder in which to install Dev-C++ 5 beta 9 release (4.9.9.2).
Select the directory to install Dev-C++ to :
Destination Folder
C:\Dev-Cpp
Browse...
Space required: 59.2MB
Space available: 38.6GB
Nullsoft Install System v2.0
< Back
Install
Cancel

图　1.5


Adobe
Autodesk
Bloodshed Dev-C++
Dev-C++
License
Uninstall Dev-C++
Bloodshed Dev-Pascal
BuddyRobot
Cena
Dell
Dell Webcam
Embarcadero RAD Studio 2010
ESET
freeime
返回
搜索程序和文件
图片
音乐
游戏
计算机
控制面板
设备和打印机
默认程序
帮助和支持
关机

图　1.6


Dev-C++ 4.9.9.2
菜单栏
工具栏
工程管理区
代码编辑区
编译输出区
状态条
Ready

图　1.7

如不习惯 Dev-C++默认的英文界面，可以单击菜单栏 Tools 中的 Enviroment Options 选项打开环境设置对话框，将 InterFace 栏中的 Language 选项改为 Chinese，如图 1.8 所示。

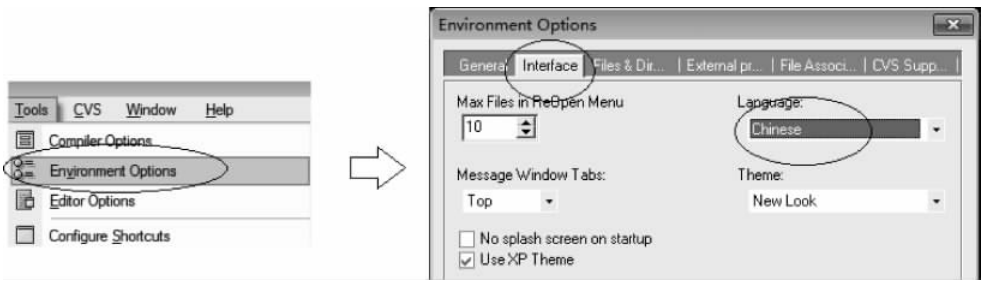


图 1.8

代码编辑区的编辑操作与 Microsoft Word 的操作很相似，例如剪切：Ctrl＋X；复制：Ctrl＋C；粘贴：Ctrl＋V；恢复：Ctrl＋Z；重作：Ctrl＋R……

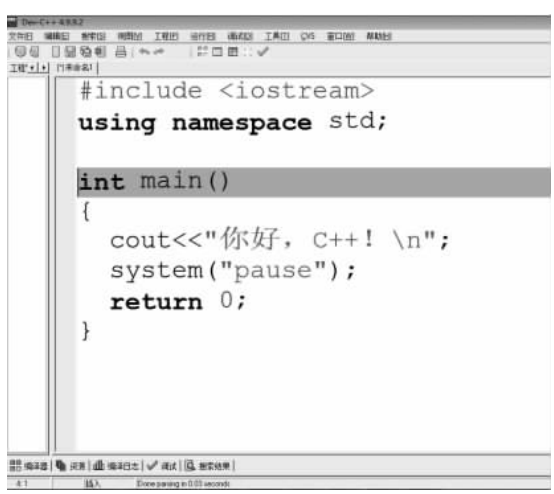
如果感觉代码编辑区域太小，按 F12 键可切换为全屏编辑状态。

在编写第一个 C++程序之前，需新建一个源代码文件，新建源代码的方法有三种，方法一是直接单击工具栏中的“新建源代码”图标，方法二是打开“文件”菜单，单击“源代码”子菜单，方法三是直接按快捷键 Ctrl＋N，如图 1.9 所示。

图 1.9

【例题描述】 我的第一个程序

在代码编辑区输入图 1.10 所示的代码。



```
#include <iostream>
using namespace std;

int main()
{
  cout<<"你好, C++! \n";
  system("pause");
  return 0;
}
```

图 1.10

程序由语句组成，除第一行预处理语句和 main 函数体的定义外，每条语句由分号(；)作为结束符。

注意 C/C++语言是对字母大小写敏感的，如“a”和“A”代表不同的意思。并且除双引号(" ")包含的字符串中的中文以及注释语句外，其他字符一般用英文半角格式输入。

#include <iostream>表示嵌入名为 iostream 的 C++标准头文件，此头文件包含输入输出的相关信息。因为我们编写的 C++程序几乎都涉及输入输出操作，所以此行语句为必写语句。

using namespace std;指定名字空间为 std。C++提供的名字空间技术可有效防止大型软件项目开发时容易出现的类、变量或函数同名的问题。由于 C++标准库中的所有组件都是在一个被称为 std 的名字空间中声明和定义的，所以我们编写的程序必须要声明 using namespace std。

main 是主函数名，函数体用一对大括号括住。函数是 C++程序中最小的功能单位。在 C++程序里，必须有且只能有一个名为 main()的函数，它表示了程序执行的开始点，程序运行时由 main 函数里的第一条语句开始执行，结束于 main 函数尾的最后一条语句。main()函数之前的 int(也可省略)表示 main()函数返回值是一个 int 类型的数(关于函数的返回值将在以后介绍)。

cout 是一个输出流对象，它是 C++系统预定义的对象，其中包含了许多有用的输出功能。输出操作由操作符“<<”来表示，其作用是将紧随其后的双引号中的字符串输出到标准输出设备(显示器)上。输出的字符串必须包含在一对双引号之中，其中“\n”表示换行。

system(“pause”)表示暂停。使用这条语句后，运行时屏幕上会出现“请按任意键继续…”的字样。如果没有这条语句，当运行程序时屏幕上的内容就会一闪而过而无法看清(更高版本的 Dev-C++使用这条语句会出现错误)。

return 0 表示函数返回值为 0，main 函数的返回值一般是返回给该程序的调用者(如操作系统)，即通知调用者程序正常结束。NOI 类型的比赛规定必须在最后一行加上这一句。

输入的程序代码要及时保存，C 语言源文件的后缀名为 C，C++语言源文件的后缀名为 C++/ CPP。C 语言是 C++语言的子集，C++语言向下兼容 C 语言，所以一般默认保存为 C++类型，如图 1.11 所示。

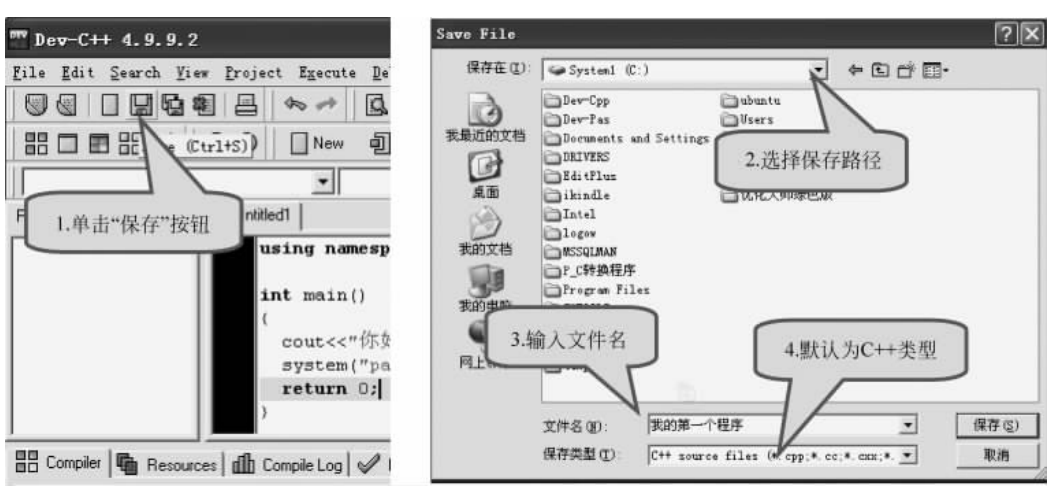


图　1.11

以 C 或 CPP 为后缀名保存的文件叫程序的源文件，源文件是不可以直接运行的，源文件首先要编译成 obj 目标文件，并与系统提供的库函数连接后得到可执行的 exe 文件后方可运行，如图 1.12 所示。

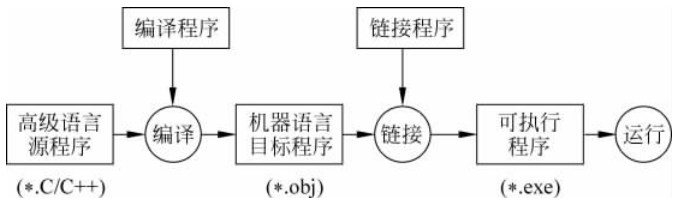


图 1.12

编译源代码如图 1.13 左图所示，打开“运行”菜单后单击“编译”，如果出现图 1.13 右图所示的消息框，则表示编译成功。

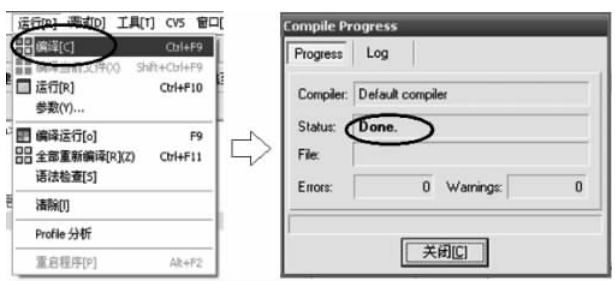


图 1.13

哎呀，急死我了，我的程序写好了，可是总是报错而无法运行，我左看右看上看下看就是找不到错误，你能帮我找找吗？

请尝试找出图 1.14 所示的错误。

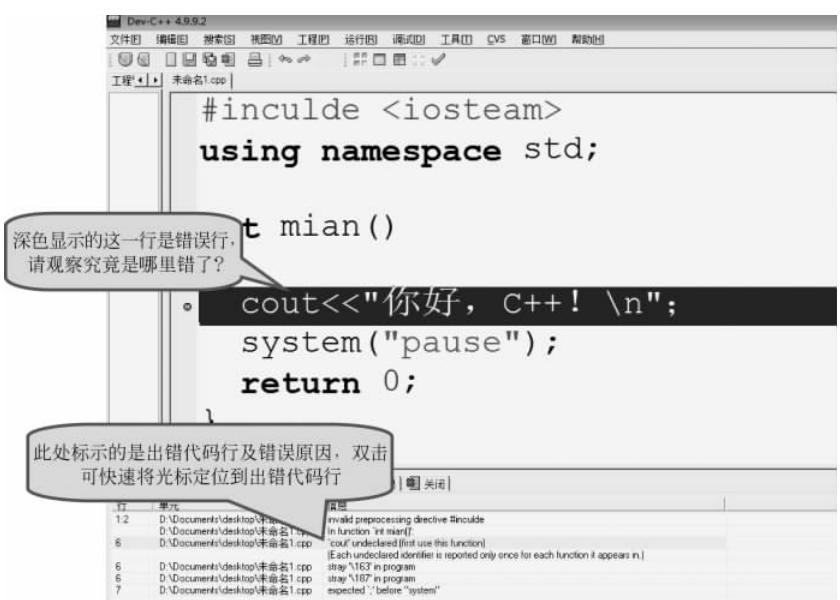


图 1.14

第一个错误是把 include 写成了 inculde。

第二个错误是把 iostream 写成了 iosteam。

第三个错误是把 main 写成了 mian。

第四个错误是输出字符串这行语句末的分号写成了中文全角字符的分号。

你可真是够粗心的。

编译程序能及时地发现一般的语法错误、输入错误或者简单的逻辑错误并给出相应的错误原因，但一些深层次隐蔽的错误还是需要自己去发现和调试。学好编程能有效地矫正粗心的毛病。

一般地，只要看到有类似于 stray '\163' in program 的错误提示，就是表示代码里有错误的全角字符格式的字符输入了。

又如'cout' undeclared 的错误提示，就是表示输入输出头文件写错了。

编译成功后，就可以运行程序观看结果了，打开“运行”菜单后单击“运行”选项，屏幕将出现如图 1.15 所示的一个 DOS 窗口，看到输出的文字了么？呵呵，第一个 C++程序成功运行了！

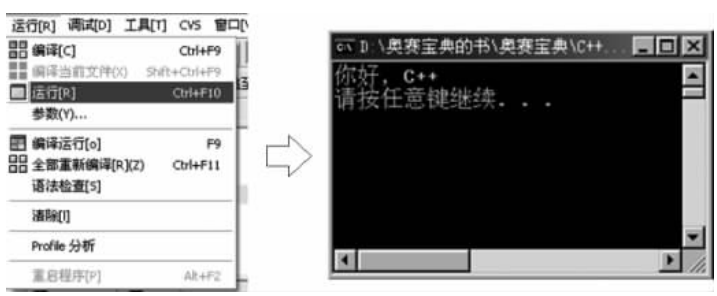


图　1.15

编译好的程序，可以离开编译器环境运行。直接单击生成的 exe 可执行文件即可运行程序，如图 1.16 所示。exe 可执行文件也可拷贝到其他安装有 Windows 操作系统的 PC 上运行。

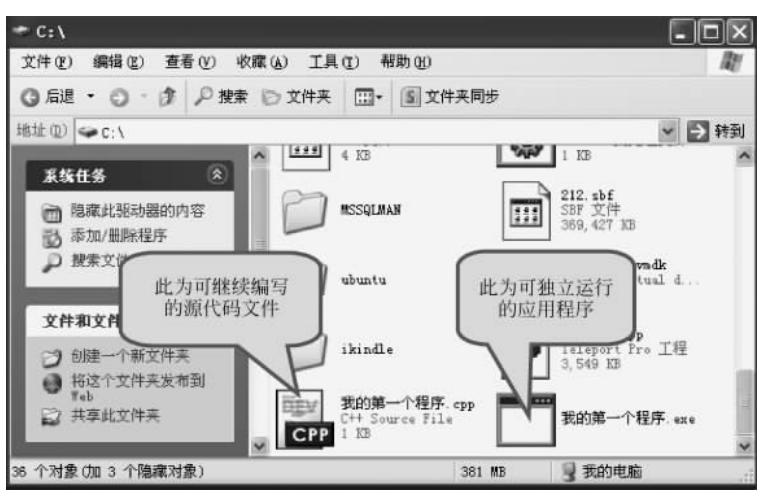


图　1.16

cout 可以随心所欲地在屏幕上输出各种字符，例如，我们可以这样写：

```
cout << "你好,C 语言\n";
cout << "我是中学生\n";
cout << "我喜欢编程\n";
```

可以看到，每一行输出的结尾均有“\n”换行。输出效果如图 1.17 所示。

【上机实践】

请尝试编程输出如下的两棵并排的树：

```
   *          *
  ***        ***
 *****      *****
*******    *******
   *          *
   *          *
```

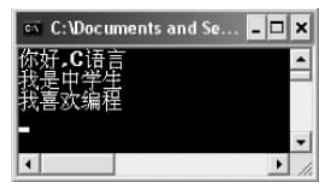


图 1.17

标准数据类型

C++语言提供了丰富的数据类型，如整数类型、实数类型（浮点数）、字符类型等。每种数据类型均有其取值范围，Dev-C++（4.9.9.2）是 Windows 平台下的 32 位编译器，基本数据类型的取值范围如表 1.1 所示（方括弧内的部分是可以省写的，例如，signed int 与 int 等价）。

表 1.1

	类 型	字节长度	取值范围
整数类型	[signed]short	2(16 位)	−327 68～32 767
	[signed]int	4(32 位)	−2 147 483 648～2 147 483 647
	[signed]long	4(32 位)	−2 147 483 648～2 147 483 647
	[signed]long long	8(64 位)	$-(2^{63}-1)\sim 2^{63}-1$
布尔类型	bool	1(8 位)	true 或 false
字符类型	char	1(8 位)	−128～127
实数类型	float	4(32 位)	−3.4E-38～3.4E+38，有效位 6～7
	double	8(64 位)	−1.7E+308～1.7E+308，有效位 15～16
	long double	16(128 位)	−3.4E+4 932～1.1E+4 932，有效位 18～19

现在的计算机普遍使用二进制（即 0 和 1 两个数码）来存储数据。二进制位（bit）是计算机存储信息的最小单位，代表 1 个二进制数位，其值为 0 或 1，可以表示两个状态/数值。

8 个连续的二进制位为一个字节，可以存放 1 个西文字符的编码（ASCII 码）。1 个汉字占两个字节。

表 1.1 列出的整数类型的取值范围均包含有负数，如果程序中需要用到的某数据类型的取值范围仅为正数，并且永远不可能取负数时，则可以使用无符号的数据类型，其取值范围如表 1.2 所示。

表　1.2

类　　型	字节长度	取 值 范 围
unsigned short	2	0～65 535
unsigned long	4	0～4 294 967 295
unsigned int	4	0～4 294 967 295
unsigned long long	8	$0\sim2^{64}-1$

使用无符号类型的整数类型，正整数的最大数据范围扩大了一倍啊。这对于某些数据规模较大的题目来说，如果不涉及负数的运算，使用无符号类型的整数类型，倒是很方便的。

sizeof 函数可用于获取各数据类型的字节长度，例如获取 long 型字节长度可这样写：

```
cout <<sizeof(long);
```

了解所使用的编译器支持的各数据类型的字节长度很有必要，可以这样写：

```
cout<<"int 的字节长度为"<<sizeof(int)<<endl;
cout<<"short 的字节长度为"<<sizeof(short)<<endl;
cout<<"long 的字节长度为"<<sizeof(long)<<endl;
cout<<"long long 的字节长度为"<<sizeof(long long)<<endl;
cout<<"bool 的字节长度为"<<sizeof(bool)<<endl;
cout<<"char 的字节长度为"<<sizeof(char)<<endl;
cout<<"float 的字节长度为"<<sizeof(float)<<endl;
cout<<"double 的字节长度为"<<sizeof(double)<<endl;
cout<<"long double 的字节长度为"<<sizeof(long double)<<endl;
```

【例题描述】 两整数相加

输入两个整数，计算两整数相加的和。

```
1  /*
2    这是一个加法程序
3  */
4  #include <iostream>
5  using namespace std;
6
7  int main()
8  {
9    int a,b,c;                                    //定义变量 a,b,c
10   cout <<"请输入 a,b 的值:";                    //显示提示信息
11   cin >> a >>b;                                 //从键盘输入 a 和 b 的值
12   c=a+b;                                        //计算 a 和 b 的和,并把结果放在 c 中
13   cout <<a<<"+"<<b<<"="<<c<<endl;  //显示结果
14   system("pause");
15   return 0;
16 }
```

第 1～3 行中的/*…… */是 C++语言里的注释语句，用于注释多行的语句。

//也是 C++语言里的注释语句，用于注释当前这一行的语句。

所谓注释语句只是方便阅读程序的人理解程序所写的语句，对编译和运行不起任何作用，当编译程序时，将自动忽略注释语句，所以注释语句的多少不影响程序运行的快慢。为便于理解，一般注释语句可用中文表示，当然也可以用英语或汉语拼音作注释。适当地添加注释语句是编程的一个好习惯。

int a,b,c;定义了 a,b,c 三个整型变量，所有的变量必须先定义后使用。

第 11 行是输入语句。cin 代表标准输入流设备即键盘，输入操作由运算符“>>”来表达，它表示通过标准输入设备输入右边的数据。cin>>a>>b 这条语句是表示从键盘输入两个整数，cin 一次可输入多个变量的值，注意输入时数与数之间要以空格间隔。

第 12 行是表达式语句，即计算 a+b 的值，并将结果存储在 c 中。

第 13 行最后的 endl 用于换行，相当于前面用过的'\n'。endl 与'\n'有细微的差别，endl 除了输出换行外，还刷新了流，在 Cena 评测系统（一个可自动评测代码是否正确的软件）中，有时使用 endl 会随机地导致程序无法通过。

我发现一个小技巧，就是在调试程序过程中，如果暂时不想运行某段代码，其实无须将它删除，只需将这段代码注释掉就可以了。

可是我又发现一个问题，就是我不小心在输入数据时多输入或者少输入了数字，甚至输入的根本就是错误字符，可是程序仍然会得出结果，虽然这结果是错误的。

从键盘输入数据时，需严格按输入语句要求的格式输入，例如上例中，当 a=1,b=2 时，cin>>a>>b 语句的正确输入格式应该是输入半角字符 1 2 后回车，其中 1 和 2 之间以一个或多个空格分隔，但不能是其他符号。诸如从键盘输入“a=1,b=2”，或者“1,2”之类的格式均是错误的。

需要注意的是，如果输入数字的时候错误地输入了其他字符，在 C 语言里会将该字符自动转换为该字符相对应 ASCII 码，但 C++语言会直接忽略该字符，而直接代之以内存中当前位置的原始储存值。

此外，输入输出语句应严格按照格式编写，例如不能写成 cin>>a,b,c;或者 cout<<a+b=c 之类的。即输入输出的字符串、各变量之间应使用“>>”或“<<”分隔。

C++程序对用到的所有数据都必须指定其数据类型。程序中常需要对一些变量预先设置初值。C++语言允许在定义变量的同时使变量初始化。例如：

```
int a=5;                    //指定 a 为整型变量,初值为 5
float f=3.45;               //指定 f 为实型变量,初值为 3.45
char c= 'a ';               //指定 c 为字符变量,初值为'a '
```

其中诸如 int a=5 这样的语句，相当于下面两条语句：

```
int a;
a=5;
```

也可以使被定义的变量的一部分赋初值。如：

```
int a,b,c=5;                //定义了整型变量 a,b,c,其中 c 的初值为 5,注意 a,b,c 之间用逗号分隔。
```