NOIP C++ 编程语言

# 先行课---计算机工作原理

计算机现在几乎无处不在，改变着我们生活的方方面面。虽然我们每天都在用着这项革命性的技术，但很少有人知道“计算机是如何工作的？”

我们学习信息比赛的同学都将是未来信息化社会的栋梁，为了学习计算机程序设计，我们首先需要了解计算机的工作原理。

位于美国西雅图的计算机科学非营利组织，code.org，作为它的计算机科学发现系列的一部分，录制了一个名为“[计算机是如何工作的](https://www.bilibili.com/video/av35082389?from=search&seid=16791620252283005022)”系列共六个小视频，由微软帝国的创始人、计算机软件领域的传奇人物比尔盖茨领衔做宣传介绍，建议大家在学习本课程时首先要完成该系列视频的学习！



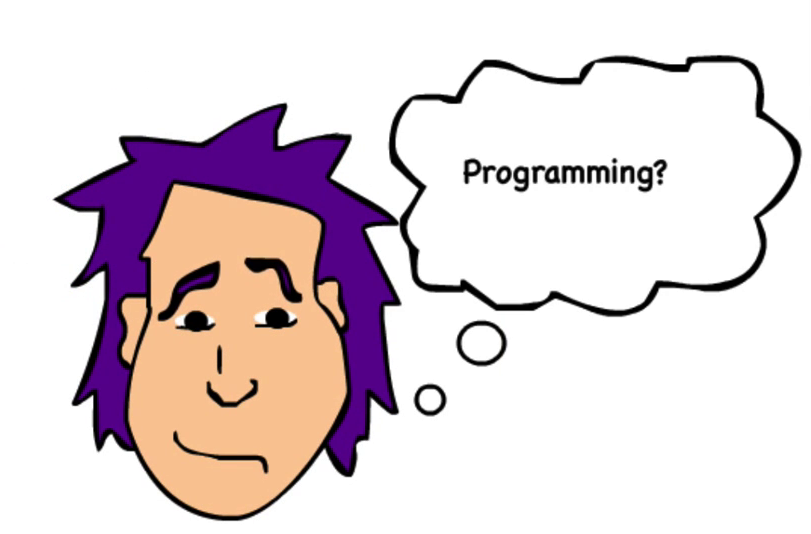
事实上，不管你是对每天用的设备感到好奇，或者想在未来做一些创新的东西，都应该从学习“计算机是如何工作的”开始。

这里是视频链接：[计算机是如何工作的](https://www.bilibili.com/video/av35082389?from=search&seid=16791620252283005022)

# 第0章 C/C++快速介绍

## 程序，编程和编程语言（Programming & Programming Language）

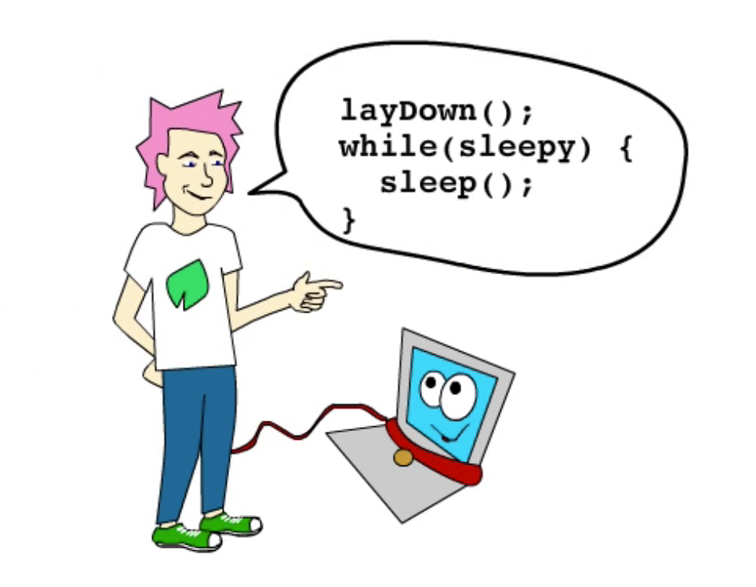
如果你之前没有学习过编程，你可能很想知道什么是编程（Programming）？



程序：Program，剧本，节目，计算机完成某种特定任务而执行的一系列指令。

编程：Programming，简言之，就是编写程序（Writing Program）。

当我们编写程序的时候，我们是使用一种看起来怪异格式的语言来给计算机下达一系列的命令（Command）。你可以认为计算机就是一个非常听话的小狗，不管你让它做什么它都会照你说的去做。



而给计算机下达命令，我们需要使用计算机能够听懂的语言，这就是编程语言（Programming Language）。

### 机器语言

计算机能够直接读懂和执行的语言，即CPU能够识别的指令，叫做机器语言，它包含控制和运算（算术（加减乘除）和逻辑运算（与或非））等基本指令，是计算机唯一能够理解的语言。

### 编程语言，编译器与解释器

人类生活中使用的语言叫做自然语言（比如我们使用的汉语普通话、英语等）。

为了能够编程（给计算机下达指令），我们需要使用双方都能理解的语言。比如我们可以直接使用机器语言进行编程（早期的编程就是直接使用机器语言，但是这样编程效率很低），或者让计算机能够理解自然语言，这样我们就可以直接使用汉语或者英语给计算机下达指令了。但自然语言过于复杂，而且有歧义，计算机很难理解（自然语言理解是目前很热门的一个人工智能研究领域）。

解决方案是引入一个中间人-翻译：为了便于人与计算机的交流（编程），人们设计了各种介入自然语言和机器语言的中间语言，即编程语言。

编程语言消除了自然语言的歧义性，容易翻译成机器语言。我们使用编程语言编写程序，这样的程序叫做**源代码**，源代码需要被翻译成机器语言代码才能被计算机执行。

机器语言，汇编语言，高级语言。

### 高级语言与低级语言

根据编程语言是更接近于机器语言还是更接近于自然语言，将编程语言分为低级语言和高级语言。

我们一般称接近于机器语言的为低级语言，比如汇编语言和C语言；更接近于自然语言的为高级语言，比如C++/Java/Python/Go语言等。

源代码翻译成机器语言代码的方式主要有两种：

事先翻译与现场翻译

### 编译与翻译

一种是编译：在源代码的执行前，事先使用编译器（compiler）将整个源代码翻译成机器语言程序后才能在计算机上执行。

另一种是解释：有一些语言，用它编写的程序不是事先翻译成机器程序，而是由一个解释器在程序运行时，一行行动态解释来翻译成机器语言指令来执行。

前一种需要事先编译成机器指令程序的语言称为编译型语言，后一种在运行时由解释器翻译成机器指令执行的语言称为解释性语言。

Pascal/C/C++语言属于编译型语言，而Python属于解释性语言。Java则是。。。有争议

## 操作系统与编程

操作系统，资源的管理者，1， 提高资源利用率2，方便使用。

Windows，Linux，macOS,Unix

OS 提供进程-虚拟计算机（i/o, cpu,memory）

OS的发展：无OS，单用户，字符，多用户分时，虚拟存储。

为虚拟计算机编程：认为我们拥有一个虚拟的计算机。

## UNIX与C语言

C语言最初是由贝尔实验室（美国）的丹尼斯·里奇 (Dennis Ritchie)和肯·汤普森（Kenneth Thompson）为开发UNIX操作系统而设计的，是一种通用的程序设计语言。其特点包括简洁的表达式、流行的控制流和数据结构、丰富的运算符集等。

一般说C语言是一种相对“低级”的语言。这种说法并没有什么贬义，它仅仅意味着C 语言更接近于机器语言，可以直接处理大部分计算机能够处理的对象，比如字符、数字和地址。由于它很适合用来编 写编译器和操作系统，因此被称为“系统编程语言”。



图一 丹尼斯·里奇 (右)和肯·汤普森（左）

**由于他们“研究发展了通用的操作系统理论，尤其是实现了UNIX操作系统”，1983年丹尼斯·里奇与肯·汤普逊一起获得了计算机界最负盛名、最崇高的、被称为计算机界的诺贝尔奖”之称的图灵奖。1999年两人为发展C语言和Unix操作系统一起获得了美国国家技术奖章。一般称丹尼斯·里奇为C语言之父，肯·汤普森为Unix系统之父，或者并称C语言与Unix系统之父。**

## C++ 语言

C++语言是由同样来自于贝尔实验室的本贾尼·斯特劳斯特卢普（Bjarne Stroustrup）在C语言的基础上引入并扩充了面向对象的概念，发明的一种新的程序语言。为了表达该语言与C语言的渊源关系，它被命名为C++。而本贾尼·斯特劳斯特卢普博士被尊称为C++语言之父。



图二 本贾尼·斯特劳斯特卢普

C语言是一门面向过程（Procedure Oriented）的编程语言，C++语言在C语言的基础上增加了面向对象（Object Oriented）和泛型编程（Generic Programming)的支持。

面向过程的编程思想认为程序是由数据以及对数据的处理（即过程）组成的，而面向对象的编程思想认为，程序，就像我们的自然世界一样，是由对象（Object）以及对象间的交互组成的。（Object翻译成“对象”并不是很好理解，翻译成客体、物体更合适）

C语言是一门简洁优雅的“低级语言”，但C++则是拥有众多高级特性的“高级语言”。事实上，由于拥有众多的高级特性，C++语言是最难于学习掌握的一门编程语言。幸运的是，信息比赛里只会使用到C++语言的一个非常小的子集，学习掌握起来还是相对容易的。不过，适当地掌握C++语言的一些高级特性，特别是标准库的使用，可以极大方便我们程序的编写，提高编程的效率，达到事半功倍的效果。

## 编程工具：编辑器（editor），编译器（compiler）以及集成开发工具（IDE）

源程序是使用某种编程语言（比如C++）编写的文件，你可以使用任何你熟悉的编辑器来编写源程序。

Windows操作系统上的写字板，记事本，word等，MacOS/Linux上的vi，vim，emac等通用编辑器都可以用来编写C++程序。

比如下面是一个C++源程序：



我们说，计算机能够直接执行的程序是机器语言程序，需要使用编译器，将C++源程序翻译成机器语言程序，这个处理过程称为编译（compiling）。

现在我们最常使用的是使用GNU编译工具集（GCC，GNU Compiler Collection)，其中用于执行C++源程序编译的编译器叫做g++。

比如将上面的源程序保存为helloworld.cc,下面的命令将它编译为可执行的机器语言程序，在Mac和Linux，生成的可执行文件名字默认为a.out：

$ g++ helloworld.cc

下面是使用hexdump工具显示的上面C++程序编译成的机器语言程序的开头几行（这是在Mac电脑上的可执行文件，Windows电脑上跟这个不一样）：



为了方便软件开发工作，人们设计了集成开发环境（IDE，Integrated Development Environment）。IDE将编辑器、编译器以及用于排除源代码错误的调试器以及其它辅助开发的工具集成在一起，特别是，这些工具可以为某种语言的程序开发高度定制，比如，编辑器中的语法高亮显示、自动代码补齐、边编辑时边进行语法检查等，图形化的调试器等，极大地方便了开发人员的工作，提高了开发的效率，建议初学者在学习编程时都要选用一款IDE来使用。

下面分别介绍下各种操作系统上推荐初学者使用的IDE：Windows系统上的Code::Blocks，Mac 系统上的Xcode， Linux平台的同学也建议Code::Blocks,不过这里不介绍它的安装。

注意，觉得安装IDE对于初学者还是一项很有挑战性的工作，建议请求学长或者老师帮忙安装。此外，这里还会介绍一个在线练习C++开发的网站，初学者使用它是非常方便的。

# 第1章　程序设计入门

## 1.0 说，你好！ --- Hello world!

*本程序主要介绍：*

*C++程序基本结构：函数，即处理，面向过程的编程风格*

*Main 函数：C++程序的入口，包含全部的处理*

*Include声明*

*标准输出对象*

*流输出操作符，链接*

*字符串常量*

*换行符*

学习一门新程序设计语言的惟一途径就是使用它编写程序。对于所有语言的初学者来说，编写的第一个程序都是相同的，即：

### 程序1-0：请打印出下列内容：

hello, world!

尽管这个练习很简单，但对于初学语言的人来说，它仍然可能成为一大障碍，因为要实现这个目的，我们首先必须编写程序文本，然后成功地运行编译，并加载、运行，最后输出到某个地方。掌握了这些操作细节以后，其它事情就比较容易了。

在 C++语言中，我们可以用下列程序打印出“hello, world!”：



**注意：上面图中代码的高亮显示是编辑器加上帮助我们更好地阅读代码的，你的编辑器不一定使用同样的高亮，或者没有高亮显示，没有关系，不要被它困扰。后面的代码展示会**

下面对程序本身做些说明。

#### 处理与存储

一个C++语言程序，是由函数和变量组成的。函数中包含一些语句，以指定所要执行的计算操作；变量则用于存储计算过程中使用的值。**函数和变量分别对应计算机四个任务中的处理（Processing）与存储（Storage）。**

在本例中，函数的名字为main。通常情况下，函数的命名没有限制，但main是一个特殊的函数名——每个程序都从main函数的起点开始执行，这意味着每个程序都必须在某个位置包含一个main函数。

#### include （包含）声明

#include <iostream>

这一行指示compiler（编译器）include（包含）iostream里的标准I/O（input/output 输入输出）设施的声明。如果没有这些声明，程序就找不到std::cout的声明了。（可以试着删掉#include <iostream>这一行，看看编译有什么错误）

#### 输出

std::cout << "Hello world!\n";

std::cout是C++语言标准库定义的一个标准字符输出设备，它对应着计算机四个任务中的输出（output）。

<<操作符（“输出到”）将它的第二个参数的内容输出到第一个参数。在这个例子里，它将字符串常量“Hello world!\n” 输出到标准输出流std::cout.

#### 字符串常量

用双引号括起来的字符序列称为字符串或字符串常量，如"hello, world\n"就是一个字符串。

字符串常量中的\n表示换行符，在打印中遇到它时，输出打印将换行，从下一行的左端行首开始。如果去掉字符串中的\n(这是个值得一做的练习)，即使输出打印完成后也不会换行。

<<操作符的输出不会添加自动换行，这意味着可以将例子中的字符串分多次输出以分阶段得到一个长的输出行，例如上面给出的第一个程序也可以改写成下列形式:



<< 操作符的多次输出可以串联，如下所示：



此外，各个例子中字符串中的\n也可以使用输出单独的std::endl(end of line)来替换，比如：



请注意，\n 只代表一个字符。类似于\n 的转义字符序列为表示无法输入的字符或不可见字符提供了一种通用的可扩充的机制。除此之外，C++ 语言提供的转义字符序列还包括:\t 表示制表符;\b 表示回退符;\"表示双引号;\\表示反斜杠符本身。

**练习 1-1** 在你自己的系统中运行“hello,world”程序。再有意去掉程序中的部分 内容，看看会得到什么出错信息。

**练习 1-2** 做个实验，当“hello,world”程序中的字符串中包含\c(其中 c 是上面的转义字符序列中未曾列出的某一个字符)时，观察一下会出现什么情况。

## 1.1 算术表达式

计算机的“本职”工作是计算，因此接下来先从算术运算入手，看看如何用计算机进行复杂的计算。

### 程序1-1　计算并输出1 + 2的值

*本程序主要介绍：*

*整数算术表达式：+ - \* / %*

1 #include <iostream>

2

3 int main()

4 {

5 std::cout << 1 + 2 << std::endl;

6 return 0;

7 }

下面做4个实验。

实验1：修改程序1-1，输出3-4的结果。

实验2：修改程序1-1，输出5×6的结果。

实验3：修改程序1-1，输出8÷4的结果。

实验4：修改程序1-1，输出8÷5的结果。

#### 整数算术与小数（浮点数）算术

数学里有整数算术和小数算术，计算机如何知道你要整数算术还是小数？

### 变量与输入

*本程序主要介绍：*

*变量：即存储*

*标准输入设备*

*完整的输入、存储、处理和输出*

*操作系统？？？待定*

1.1节的程序虽好，但有一个遗憾：计算的数据是事先确定的。为了计算1+2和2+3，下面不得不编写两个程序。可不可以让程序读取键盘输入，并根据输入内容计算结果呢？答案是肯定的。程序如下：

#include <iostream>

int main()

{

int a, b;

std::cin >> a;

std::cin >> b;

std::cout << a + b << std::endl;

return 0;

}

#### 变量 – 有名字和类型的一块存储

所有的计算机由四部分组成：输入，存储，处理，输出。

在C++语言中，所有变量都必须先声明（申请）后使用。声明用于说明变量的属性，它由一个类型名和一个变量表组成：

* 类型规定了变量可能的取值范围以及可以对其可以执行哪些操作；
* 变量在内存中占据一块存储空间，空间大小由其类型决定；
* 变量的存储空间里放置的是二进制**比特位（bit）序列**，它具体表示的值需要根据变量的类型解释；
* 一个变量就是一个有名字的对象。

例如，本例中定义了两个变量：a和b：



类型int表示其后所列变量为整数（integer），与之相对应的，float表示所列变量为浮点数（即，可以带有小数部分的数）。

除 int与float类型之外，C语高还提供了其它一些基本数据类型，例如：

char 字符

short 短整型

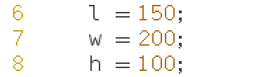
long 长整型

double 双精度浮点型

计算机内部使用二进制数字存放信息，信息的基本单位是比特（bit，binary digit），8个比特组成一个字节（byte）。字节是计算机存储的基本单位。类型的大小通常使用它占用的存储空间的字节数或者位数来表示，同一个类型的变量占据的存储空间的大小是相同的。

各种整型和浮点类型的大小取决于具体的机器，对于int类型，通常为4个字节（32位），其取值范围在-2147483648～2147483647 之间。float 类型通常也是4个字节（32位），它至少有6位有效数字，取值范围一般在10-38～1038之间。

在上面的程序中，最开始执行的计算是下列3个赋值语句：



它们为变量设置初值。各条语句均以分号结束。

最后三句，使用相应的算术表达式分别计算棱长总和、表面积和体积。

#### 输入

std::cin >> a;

std::cin >> b;

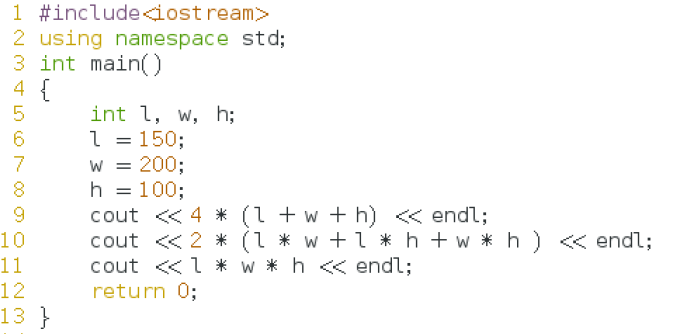
std::cin是C++语言标准库定义的一个标准字符输入设备，它对应着计算机四个任务中的输入（input）。

>>操作符（“输入到”）从它的第一个参数输入内容到第二个参数。在这个例子里，它从标准输入流std::cin先后输入两个整数到变量a和b中.

### 变量与算术表达式

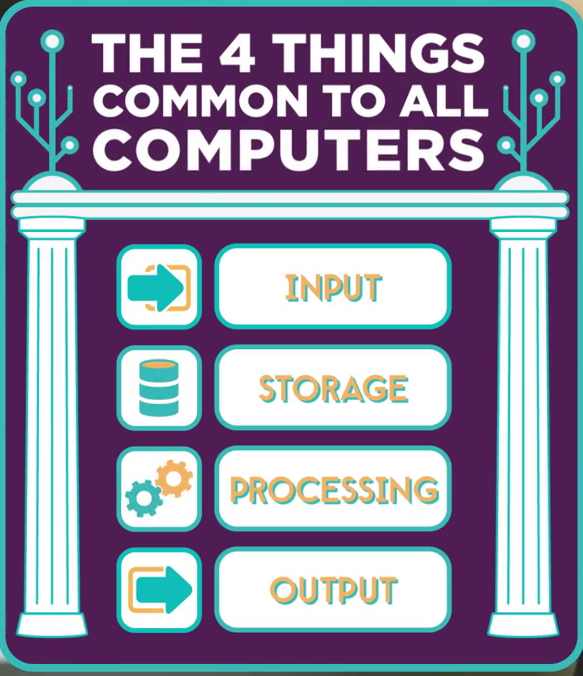
我们来看下一个程序，已知一个长方体的长、宽和高分别是150、200和100，求它的棱长总和、表面积和体积：

此程序中仍然只包括一个名为main 的函数定义。它比前面打印“hello, world”的程序长一些，但并不复杂。这个程序中引入了一些新的概念，包括m命名空间声明、变量和算术表达式。该程序如下所示：



#### using namespace 声明 --- 哪个小明？

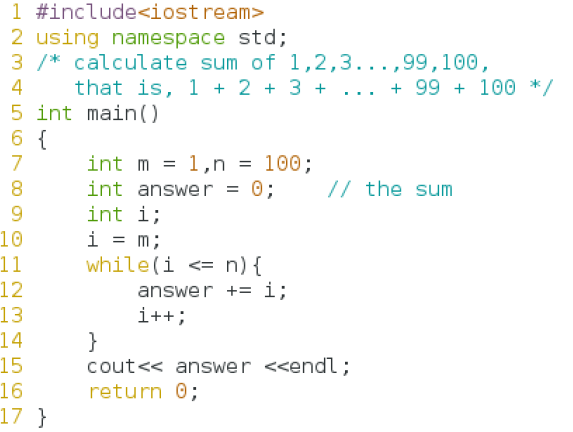
首先，using namespace std 这一行，表示使用名字为std的命名空间，即标准命名空间，它指示编译器在查找名字的时候到标准命名空间中去查找，这样我们在后面使用cout和endl这两个名字的时候就不需要写成**std::**cout和**std::**endl了。



## 变量与算术表达式

我们来看下一个程序，计算1 + 2 + 3 + … + 99 + 100的和：

因为计算机做算术运算特别快，特别是加法，我们在这个程序中利用计算机的这一个特长来给一个直观的做法---挨个加。此程序中仍然只包括一个名为main 的函数定义。它比前面打印“hello, world”的程序长一些，但并不复杂。这个程序中引入了一些新的概念，包括注释、声明、变量、算术表达式以及循环。该程序如下所示：



第3行和第4行：



称为注释，此处，它简单地解释，该程序是做什么用的。包含在/\*与\*/之间的字符序列将被编译器忽略。注释可以自由地运用在程序中，使得程序更易于理解。程序中允许出现空格、制表符或换行符之处，都可以使用这种注释。注释也能以 // 开始，直到行末为止。例如第8行，绿色部分也是注释，解释下前面的变量的用途：answer的值就是要求的sum。

