C语言与C++语言的区别

王子宇

四川大学计算机学院

C语言与C++语言作为两种重要的编程语言，对刚接触编程的人有着重要意义。C++语言作为C语言的继承，与C语言有着紧密的联系，也有所区别。本文将综合前人的经验和个人的总结，论述C++与C语言的区别。

要论述这两门语言，自然要先对它们有所了解。

C语言是一种通用的编程语言，广泛用于系统软件与应用软件的开发。于1969年至1973年间，为了移植与开发UNIX操作系统，丹尼斯·里奇与肯·汤普逊，以B语言为基础，在贝尔实验室设计、开发出了C语言。

C语言具有高效、灵活、功能丰富、表达力强和较高的可移植性等特点，在程序员中备受青睐，成为最近25年使用最为广泛的编程语言。目前，C语言编译器普遍存在于各种不同的操作系统中，例如Microsoft Windows、macOS、Linux、Unix等。C语言的设计影响了众多后来的编程语言，例如C++、Objective-C、Java、C#等。

二十世纪八十年代，为了避免各开发厂商用的C语言语法产生差异，由美国国家标准局为C语言订定了一套完整的国际标准语法，称为ANSI C，作为C语言的标准。二十世纪八十年代至今的有关程序开发工具，一般都支持匹配ANSI C的语法。目前2011年12月8日，国际标准化组织（ISO）和国际电工委员会（IEC）发布的C11标准是C语言的第三个官方标准，也是C语言的最新标准，该标准更好的支持了汉字函数名和汉字标识符，一定程度上实现了汉字编程。

特性：

C语言是一个有结构化程序设计、具有变量作用域（variable scope）以及递归功能的过程式语言。

C语言传递参数均是以值传递（pass by value），另外也可以传递指针（a pointer passed by value）。

不同的变量类型可以用结构体（struct）组合在一起。

只有32个保留字（reserved keywords），使变量、函数命名有更多弹性。

部分的变量类型可以转换，例如整型和字符型变量。

通过指针（pointer），C语言可以容易的对内存进行低级控制。

编译预处理（preprocessor）让C语言的编译更具有弹性。[[1]](#footnote-1)

C++是一种广泛使用的计算机程序设计语言。它是一种通用程序设计语言，静态数据类型检查，支持多重编程范式，例如过程化程序设计、数据抽象化、面向对象程序设计、泛型程序设计、基于原则设计等。

比雅尼·斯特劳斯特鲁普博士在贝尔实验室工作期间在20世纪80年代发明并实现了C++。起初，这种语言被称作“C with Classes”（“包含类别的C语言”），作为C语言的增强版出现。随后，C++不断增加新特性。虚函数（virtual function）、操作符重载（operator overloading）、多重继承（multiple inheritance）、模板（template）、异常处理（exception）、RTTI（Runtime type information）、名字空间（namespace）逐渐纳入标准。1998年，国际标准组织（ISO）颁布了C++程序设计语言的第一个国际标准ISO/IEC 14882:1998，目前最新标准为ISO/IEC 14882:2017。根据《C++编程思想》（Thinking in C++）一书所评述的，C++与C的效率往往相差在±5%之间。

C++语言发展大概可以分为三个阶段：第一阶段从80年代到1995年。这一阶段C++语言基本上是传统类型上的面向对象语言，并且凭借着接近C语言的效率，在工业界使用的开发语言中占据了相当大份额；第二阶段从1995年到2000年，这一阶段由于标准模板库（STL）和后来的Boost等程序库的出现，泛型程序设计在C++中占据了越来越多的比重性。当然，同时由于Java、C#等语言的出现和硬件价格的大规模下降，C++受到了一定的冲击；第三阶段从2000年至今，由于以Loki、MPL(Boost)等程序库为代表的产生式编程和模板元编程的出现，C++出现了发展历史上又一个新的高峰，这些新技术的出现以及和原有技术的融合，使C++已经成为当今主流程序设计语言中最复杂的一员。

在《C++语言的设计和演化》（1994）中，Bjarne Stroustrup描述了他在设计C++时，所使用的一些原则。知道这些原则有助于理解C++为何会是现在这个样子。以下总结了一些原则，详尽的内容可参阅《C++语言的设计和演化》：

C++设计成直接的和广泛的支持多种程序设计风格（过程化程序设计、数据抽象化、面向对象程序设计、泛型程序设计）。

C++设计成给程序设计者更多的选择，即使可能导致程序设计者选择错误。

C++设计成尽可能与C兼容，借此提供一个从C到C++的平滑过渡。

C++避免平台限定或没有普遍用途的特性。

C++不使用会带来额外开销的特性。

C++设计成无需复杂的程序设计环境。[[2]](#footnote-2)

对两种语言有了基本了解，现在来谈谈它们的区别。

C和C++的关系：就像是win7跟win8的关系。C++是在C的基础上增加了新的理论，玩出了新的花样。所以叫C++。

Ｃ语言是一种结构化编程语言。它层次清晰，便于按模块化方式组织程序，易于调试和维护。它的重点在于算法和数据结构。C程序的设计首要考虑的是如何通过一个过程，对输入（或环境条件）进行运算处理得到输出（或实现过程（事务）控制）。Ｃ语言的表现能力和处理能力极强。它不仅具有丰富的运算符和数据类型，便于实现各类复杂的数据结构。它还可以直接访问内存的物理地址，进行位(bit)一级的操作。由于Ｃ语言实现了对硬件的编程操作，因此Ｃ语言集高级语言和低级语言的功能于一体。既可用于系统软件的开发，也适合于应用软件的开发。此外，Ｃ语言还具有效率高，可移植性强等特点。因此广泛地移植到了各类各型计算机上，从而形成了多种版本的Ｃ语言。

C++ 是在C的基础上改进后的一种编程语言。C++，首要考虑的是如何构造一个对象模型，让这个模型能够契合与之对应的问题域，这样就可以通过获取对象的状态信息得到输出或实现过程（事务）控制。 所以C与C++的最大区别在于它们的用于解决问题的思想方法不一样。之所以说C++比C更先进，是因为“ 设计这个概念已经被融入到C++之中 ”。

C 语言与 C++ 的区别有很多：

1，全新的程序 程序思维，C 语言是面向过程的，而 C++ 是面向对象的。

2，C 语言有标准的函数库，它们松散的，只是把功能相同的函数放在一个头文件中；而 C++ 对于大多数的函数都是有集成的很紧密，特别是 C 语言中没有的 C++ 中的 API 是对 Window 系统的大多数 API 有机的组合，是一个集体。但你也可能单独调用 API。

3，特别是 C++ 中的图形处理，它和语言的图形有很大的区别。C 语言中的图形处理函数基本上是不能用在中 C++ 中的。C 语言标准中不包括图形处理。

4，C 和 C++ 中都有结构的概念，但是在 C 语言中结构只有成员变量，而没成员方法，而在 C++ 中结构中，它可以有自己的成员变量和成员函数。但是在 C 语言中结构的成员是公共的，什么想访问它的都可以访问；而在 VC++ 中它没有加限定符的为私有的。

4，C 语言可以写很多方面的程序，但是 C++ 可以写得更多更好，C++ 可以写基于DOSr程序，写 DLL，写控件，写系统。

5，C 语言对程序的文件的组织是松散的，几乎是全要程序处理；而 C++ 对文件的组织是以工程，各文件分类明确。

6，C++ 中的 IDE 很智能，和 VB 一样，有的功能可能比VB还强。

7，C++ 对可以自动生成你想要的程序结构使你可以省了很多时间。有很多可用的工具如加入 MFC 中的类的时候，加入变量的时候等等。

8，C++ 中的附加工具也有很多，可以进行系统的分析，可以查看 API；可以查看控件。[[3]](#footnote-3)

9，类。类对于初学者，它是一个累赘。类的封装使得初学者对程序产生厌倦，感到不适和麻烦。

10，引用。引用在C++中最好尽量不要用它，除非万不得已。引用对于初学者就更容易产生混淆，不知道哪个是引用，哪个是变量。

11，函数的重载,。初学者学函数的重载好像没什么坏处，但是，这会使初学者潜意识里对C语言的变量类型的重要性产生淡化，要记住C语言是对变量类型最敏感了的，变量的类型在C语言里的重要性是不言而喻的。

12，流操作符。和上面同样的道理，使得对变量类型的重要性产生淡化，有时会产生使初学者莫名其妙的结果。

13，操作符重载。典型的高级应用，初学者可能根本用不着，会让他们觉得C++很难，门槛高，看不懂。

14，继承，以及虚函数，看起来深奥，实用价值很低。

对语言本身而言，C是C++的子集，那么是什么样的一个子集？从上文可以看出， C实现了C++中过程化控制及其它相关功能，而在C++中的C，相对于原来的C还有所加强，引入了重载、内联函数、异常处理等等，C++更是拓展了面向对象设计的内容，如类、继承、虚函数、模板和包容器类等等。 再提高一点，在C++中，数据封装、类型已不是什么新鲜事了，需要考虑的是诸如：对象粒度的选择、对象接口的设计和继承、组合与继承的使用等等问题。

所以相对于C，C++包含了更丰富的“设计”的概念，但C是C++的一个自洽子集，也具有强大的功能，同样值得学习[[4]](#footnote-4)

对于技术细节上来说，C++中新增的概念有：bool类型、引用、类、模板、异常。C++相对于C有变化的语法点有：变量分散定义、函数名重载、struct语法。下面进行逐一分析：

1.1 在项目里添加c文件

当文件以.c作为后缀时，编译器以C语言的语法对该文件进行编译；当文件以.cpp作为后缀时，编译器以C++的语法标准对其进行编译。所以，文件的后缀名是有这个约定的，不能够随便更改。

在添加文件时，输入文件名的后缀是.c，那在编辑这个文件时必须遵守C语言的语法。

1.2 变量分散定义

在C语言中，所有的局部变量必须在函数或复合语句的最前面集中定义，简单地说就是变量定义语句要放在前面。不然就报语法错误。

下面的main.c（C语言）代码中，局部变量a,b的均在函数体的前部定义，因而符合C语言的语法要求，

////////////// main.c //////////

#include <stdio.h>

void main()

{

int a; // 变量定义

int b; // 变量定义

a = 10;

printf("a=%d \n", a);

b = 11;

printf("b=%d \n", b);

}

下面的main.c（C语言）代码中，局部变量c的定义没有放在前面，不符合C语言的语法要求，因而编译报错，

////////////// main.c //////////

#include <stdio.h>

void main()

{

int a; // 变量定义

int b; // 变量定义

a = 10;

printf("a=%d \n", a);

b = 11;

printf("b=%d \n", b);

int c ; // 变量定义语句不能放在这里

c = a + b;

printf("c=%d\n", c);

}

显然，这样对于程序员来说是很不方便的。于是C++语言去掉了这个限制，让程序员可以随时随地的定义变量。

1.3 函数名重载

C++允许多个函数使用相同的函数，这称为函数名重载。而这在C语言中是万万不可以的，在C语言中，全局函数名是不能重复的。例如，在下面的main.c文件中，存在两个名称相同的全局函数do\_something，导致编译器报错，

////////////// main.c //////////

#include <stdio.h>

void do\_something(int x, int y)

{

printf("int , int \n");

}

void do\_something(double x, double y)

{

printf("double , double \n");

}

void main()

{

printf("test");

}

这个问题的本质是，在C语言中，以函数名来唯一区分一个全局函数。换句话说，函数名就是函数的唯一ID，显然ID这个东西是不能冲突的。

这导致了在C语言中，给函数起名字也成了一个比较麻烦的事情，对于功能类似的函数，由于不能起相同的名字，所以只好不停地加一些后缀以示区别，例如，

do\_something\_int\_int()

do\_something\_double\_double()

如果你将来阅读一些C语言的代码，相信会有此感受。

C++针对此问题的改进是：以函数名+参数列表来唯一区分函数。所以，在C++中函数名是允许相同的，只要参数列表有所不同就不会冲突。

在同一个项目中，一般要么全部是.c后缀，按C语法编写；要么是.cpp后缀，按C++语法来编写。但是也存在混用的情形，这种情况有一定难度，参考《C++与C的混合调用》。

1.4 struct定义

在C语言中, struct类型的定义必须加上struct的前缀，

////////////// main.c //////////

#include <stdio.h>

struct Object

{

int id;

char name[128];

};

void main()

{

struct Object obj; // C语言里必须加上struct前缀

obj.id = 1;

}

而在C++中，struct可以直接使用其类型名来定义

////////////// main.cpp //////////

#include <stdio.h>

struct Object

{

int id;

char name[128];

};

void main()

{

Object obj; // C++里不须要加struct前缀

obj.id = 1;

}

相比之下，应该是C++的语法更简洁一些。所以在用C语言编写代码的时候，老是要加一个struct前缀会让人觉得不耐烦，所以C程序员通常是这么来定义的struct的。在下面的代码中，使用typedef来定义一个object\_t的类型，

////////////// main.cpp //////////

#include <stdio.h>

typedef struct

{

int id;

char name[128];

}object\_t;

void main()

{

object\_t obj;

obj.id = 1;

}

在C语言的世界里，这么定义struct是通用模式，为的就是在使用的时候少写一个struct前缀。(可见C程序员有多么无耐)。

在需要定义链表的时候，C程序员有两种写法，

struct Object

{

int id;

struct Object \* next; //

};

或

typedef struct \_tag\_object

{

int id;

struct \_tag\_object \* next;

}object\_t;

同样的，C程序员会选择第二种定义方法，唯一目的就是为了在使用的时候少写一个关键字struct而已。

1.5 C++中的struct

实际上，C++对struct进行提全面的升级，以至于C++的struct几乎完全等效于class的语法了。凡是class支持的语法，包括成员函数、public/private、继承等所有的语法，struct也都全部支持且意义不变。也就是说，struct是class的同义语（略有差别，后面讲到）。

唯一的一点小区别：对于struct，如果没有声明任何修改符(public/protected/private)，则默认所有成员是public的。这一点上class相反，class默认是private的。[[5]](#footnote-5)

知乎上有人这样形象地比喻：

C 语言： 把大象放进冰箱 。

（1） 打开冰箱门 （2）把大象放进去 （3） 关上冰箱门。

C++的版本就会比较多了：

第一种 ：（1）打开冰箱门 （2）把大象放进去 （3） 关上冰箱门。

第二种 ：首先定义冰箱类 ：包含三个方法 （1）开门 （2）关门 （3）放置（参数是大象）然后 定义大象类。接下来，构建大象和冰箱对象，然后调用开门方法 ，放置，最后关门。

第三种 ：定义一个容器类 ：包含三个方法 （1）打开容器 （2）放入物品 （3）关闭容器 。然后定义物品类。接着定义冰箱类作为容器的继承者继承所有功能，定义大象继承物品所有功能。接下来你就可以像把物品放入容器一样将大象放进冰箱了。以后你可以用容器类和物品类来解决将猴子放进烤箱，将老虎关进笼子等等。

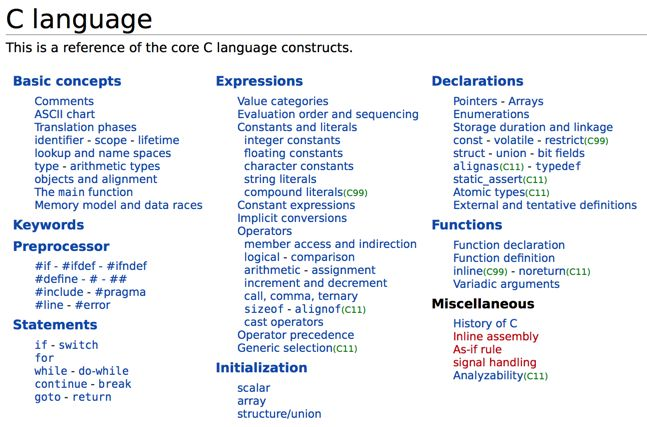
第四种 ：定义模板类，包含三种模板方法 （1）初始化各种准备条件 （2）对于动作的双方调用某种动作方法 （3）清理现场。然后将第三种方法定义的类和放置方法传递给模板就能完成将大象放进冰箱或者将老虎关进笼子的工作了。以后你可以定义各种各样的类和 方法来完成各种工作，比如将作业交给老师；把金鱼喂给猫咪；把煎蛋放进三明治等等

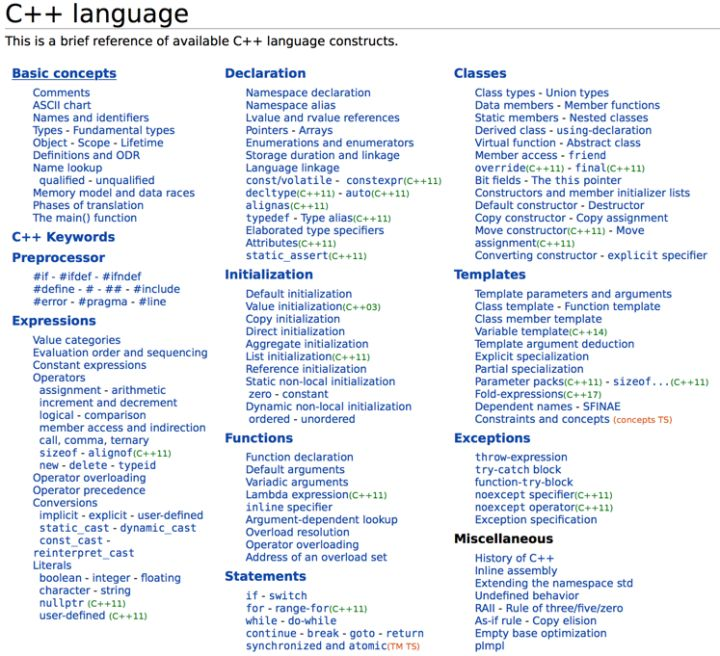
…

第N种 ：…

C++就是为你提供这N种方法的语言。[[6]](#footnote-6)

最后以两张对比图结束本文[[7]](#footnote-7)：





1. <https://en.wikipedia.org/wiki/C_(programming_language)>

   https://baike.baidu.com/item/c%E8%AF%AD%E8%A8%80/105958?fr=aladdin [↑](#footnote-ref-1)
2. https://en.wikipedia.org/wiki/C%2B%2B [↑](#footnote-ref-2)
3. <https://www.douban.com/group/topic/22414944/> [↑](#footnote-ref-3)
4. http://c.biancheng.net/cpp/html/798.html [↑](#footnote-ref-4)
5. http://blog.csdn.net/iamshaofa/article/details/44303883 [↑](#footnote-ref-5)
6. https://www.zhihu.com/question/28834538/answer/126989447 [↑](#footnote-ref-6)
7. http://en.cppreference.com/w/ [↑](#footnote-ref-7)