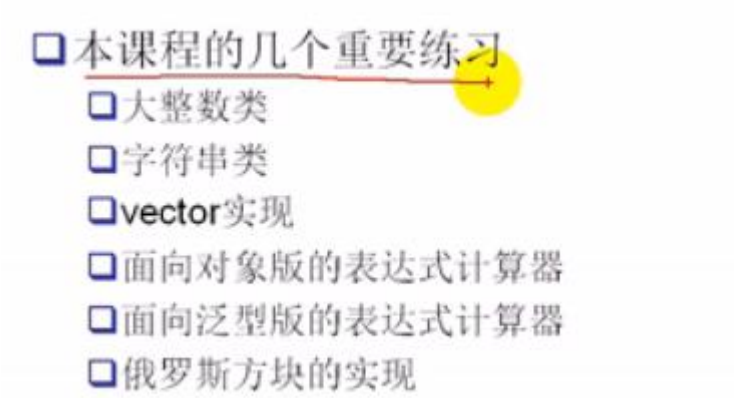
# 01 CPP 介绍

课程的目标是掌握**面向对象**的程序设计的方法.这个是最主要的，第二个目标是，初步掌握**面向泛型**的程序设计方法。

通过以下练习实现这两个目标 ：




第一个练习是编写一个大整数类，通过类的编写，掌握数据封装与数据抽象。

第二个例子是编写一个字符串类，在这个例子中，大家要掌握的是运算符的重载，内存管理和拷贝控制。

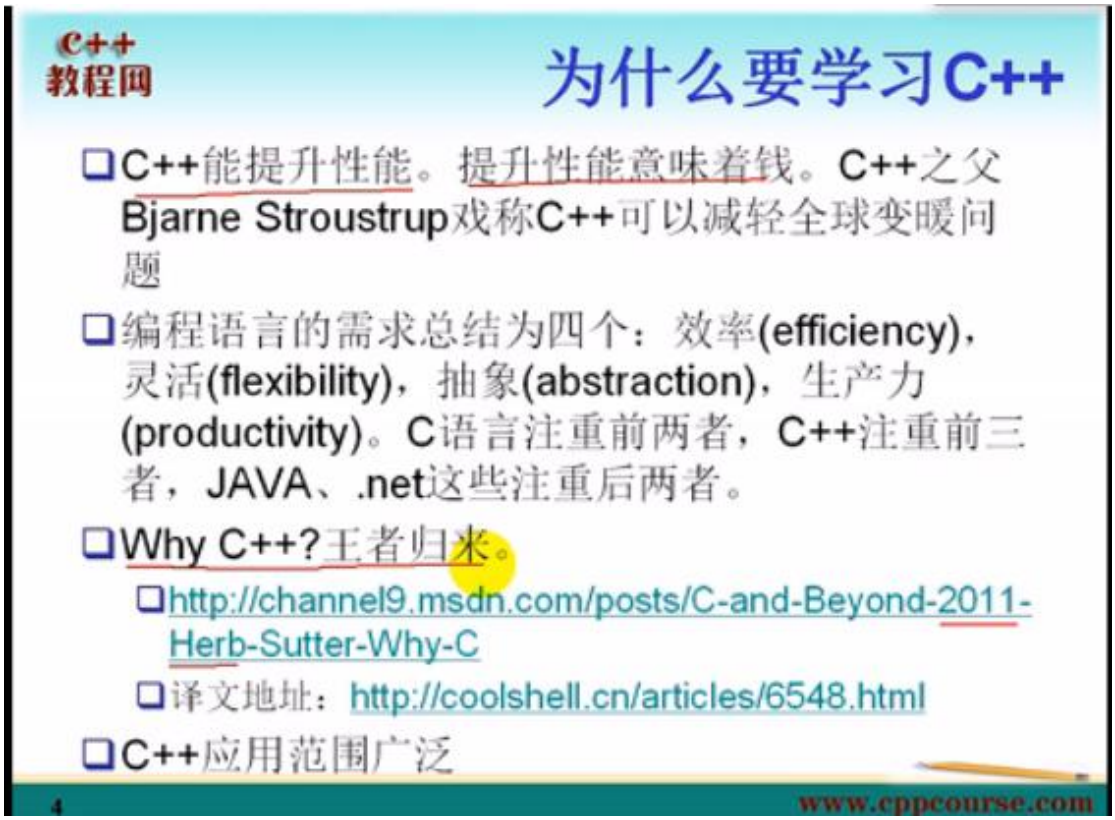
第三个例子是实现一个 Vector,这个例子我们使用模板的方式来实现，通过编写一个 Vector 类模板来熟悉模板编程。

加下来是两个表达式计算器的实现，一个是使用面向对象的方式来实现，一个是采用面向泛型的方式来实现。这个例子我们还可以进一步进行扩充，让他不仅仅能够支持表达式的解析，还能够支持变量的解析和函数的解析，也就是我们可以把它扩充为一个小型的编译器，当然要做到这一点，需要有一定的编译原理的基础.

第四个例子是采用面向对象的思想来实现俄罗斯方块。这是我们的课程安排的结果练习。

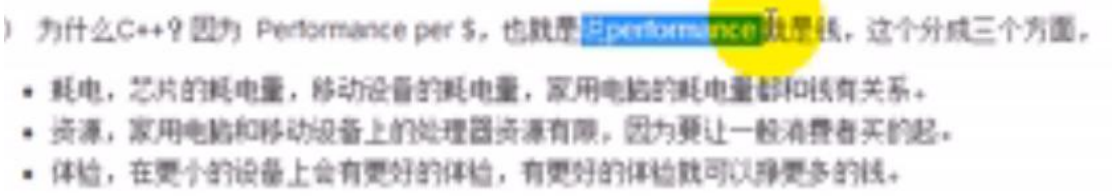
第一章 C++ 介绍

- 为什么要学习C++
- C++为什么难学
- C++ 11值得学习的新特性
- 几本推荐的书籍
- 开发工具选择



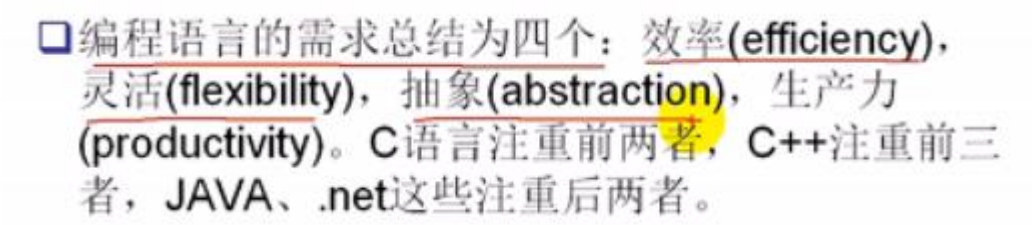

**1，为什么要学习 C++?**

C++能够提升性能，提升性能意味着钱。C++之父 Bjarne Stroustrup 戏称 C++可以可以减轻全球变暖的问题。




C 语言的效率比 C++要高，为什么不用 C 语言开发呢？

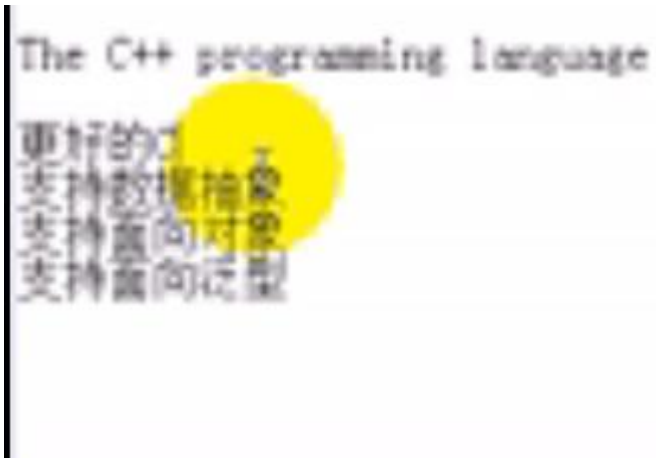
这就要知道编程语言的四个需求：




越抽象越容易开发大的系统，C 语言不适合开发大的系统。因而我们选择 C++语言。

C++的应用范围广泛，比如我们日常用的几乎所有的桌面软件都是用 C++开发的（浏览器/播放器、即时通讯工具、多媒体软件和游戏和服务器，搜索引擎的核心等等）。

**2.C++为什么难学？**


数据抽象对应数据结构中的 ADT。

面向对象的三个特征：**封装，继承和多态。**

数据抽象对应的封装和面向对象的封装有一定的区别，数据抽象是针对**值语义，**而面向对象和基于对象的封装强调的是**引用语义或者说是对象语义、指针语义**。光是讲这两者之间的区别就很复杂了，这里只做一个简单概括性的说明。

- **值语义可以拷贝和赋值，而对象语义不可以进行拷贝和赋值。**

现实生活中的大部分的事物都是不可以进行拷贝和赋值的，比如说员工账号，员工与账号信息是不可以进行拷贝和赋值的，而一些抽象的事物是可以拷贝和赋值的，比如说数据结构中多定义的一些数据结构，链表、数组、以及栈。这些对象都是可以拷贝与赋值的，而这些对象又不是现实生活中的一些具体的对象。那么它体现的是值语义，**数据抽象**在数据结构中会谈的比较多，数据抽象通常就是说对于一些数据结构的一些抽象，**将这些数据结构的取值范围隐藏起来，而仅仅只是暴露他的一些接口，这就是所谓的数据抽象。**

- 接下来我们看一下第五种编程方式：**函数式。**函数式的典型代表语言是 **LISP,ERLang,**这些语言都是函数式语言。

那么什么是函数式编程呢？它指的是一切皆函数，和面向对象不一样， 程序是以函数为中心的，甚至它不需要用 For 循环，它摒弃了 For 循环，而表现为**函数的递归调用**，也就是说它更多的是用递归的方式来编写程序，而不再使用 for 循环，一切皆函数的编程方式。我们这里不做深入的讨论，C++这里也支持函数式的编程。

- **第六个是泛型编程，面向泛型编程和面向对象编程，很多同学也是不理解的，面向对象的方式是有【程序=对象+对象+对象+对象与对象之间的消息传递】构成的，而泛型编程：【程序=对象+对象+抽象行为】，这些抽象行为能够施加到不同的对象上，而这些对象之间可能大相径庭。**

抽象行为能够施加到**大相径庭**、**不同类型**的对象之上。这就是面向泛型的一种编程思想。它强调的是一种**通用**的编程思想。也就是说不同对象之间的抽象行为是非常类似的，我们只需要将这种抽象行为施加在不同类型的对象之上。而这些不同类型的对象又是大相径庭的。而面向对象强调这些行为，并将这些行为封装在对象的内部，只有对象内部的行为能够对这个对象进行操作，所以说这是两者非常不一样的编程思想。

- 第七种编程方式是模板元编程方式，它可以看作是泛型编程的一个升华。她也是基于模板的一种编程方式，给出一个代码的产生规则，让编译器产生新的代码，来实现我们预期的功能，这样某些工作就别提前到了编译期来完成，增加了编译的时间，但是他却提高了运行的效率。

我们可以总结一下，模板元编程是指给出代码的产生规则，让编译器产生新代码，实现我们预期的功能，这样的话，能够实现某些预期的工作在编译期完成了，从而增加了编译时间，但是它提升了运行效率。这就是模板元编程，另外利用模板元编程可以进行**神奇的类型推导**。这里就不再举例，这里只做稍微的了解。

**这些编程方式都是比较难以理解的，我们没有必要去深究，了解即可，我们只需要知道，C++支持多种编程方式，像一把瑞士军刀，什么都能做，所以我们要掌握这几种编程方式的话，非常的苦难，这就是 C++为什么难学的原因，也是它非常具有魅力的原因。实际上面向对象的编程方式并不是 C++主打的一种编程方式，现在的一些有识之士正在说明面向对象的一些缺陷，提出了一种基于对象的编程思想。**

- **利用 Function 和 bind 这两个库逐步摆脱面向对象的束缚，关于这些内容的讨论，可以看课件 CSDN 中的一篇文章。**

# C++值得学习的新特性

- 智能指针的引入 ，如 shared_ptr/weak_ptr 等
- Rvalue reference 右值引用
- Function 和 bind 库，它是面向对象的救赎。摆脱面向对象的束缚，继承的束缚的一种编程思想。
- Lambda expression and closure Lamda 表达式和闭包
- 这是 C++要学习的四个特性，当然 C++不止这些特性，这里没有把它全部列出来，这四个是最值得学习的特征。

## 基本推荐书籍

- C++ Primer 第四版（第五版即将出来，会加入 C++11 的新特性），比较适合初学者。
- Effective C++ 3rd 工程实践的书籍
- C++编码规范
- 敏捷软件开发——原则、模式与实践（不是 C++的书，不仅适用于 C++还适用于其他的语言）
- 代码大全，第二版，不推荐 C++高质量编程这本书，已经过时。

## 开发工具的选择

- 目前最主流的编译有两个 GNU g++和微软的 Visual C++
- 本课程选择 VS2008 作为开发工具，再安装一个 VC 助手（Visual Assist X）.
- VS2008 开发比 VS2010 要高效率，但是不足是对 C++11 标准的支持不足。后面会有一门专门的课程来介绍 C++11 的新特性。

## VS2008 的使用

Ctrl +F5 防止程序一闪而过
安装助手的好处就是提供代码补全的功能。
在 window 下编写的程序，在 linux 下都是能运行的，只需要自己编写一个 makefile 即可。不需要做任何的更改。