

# 面向对象程序设计

## C++编程初步

刘 卉

huiliu@fudan.edu.cn

# 前言

## □ 教材——Accelerated C++

- 某宝
- 互联网下载(.chm)

## □ 课程安排

- 单周上课, 双周上机
- 上课时, 不得使用任何电子设备
- 上机重要性等同上课

www.dbooks.org  
新世纪书局

(美) Andrew Koenig, Barbara E. Moo / 编著  
张明 / 译

技术经典  
著作文丛  
信息科学

## Accelerated C++ 中文版

通过示例进行编程实践

Accelerated C++ :  
Practical Programming by Example

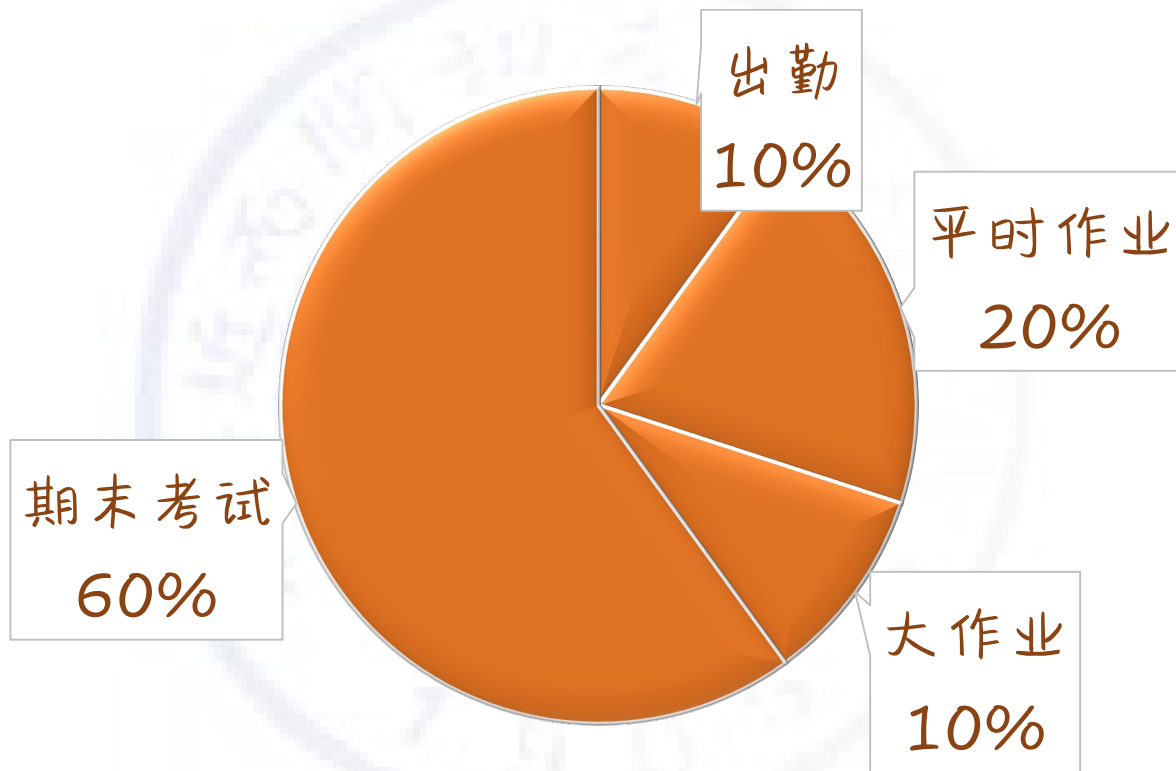


参与C++演化与变革的两位大师, 呈现给您世界最好的C++入门书籍

科学出版社



# 考核方式



## □ 作业

- 上机作业: 单周上课之后elearning上发布, 双周上机之后的星期天22:00截止.
- 迟交作业: 1) 下一次上课前, 通过邮件提交给助教, 抄送给老师;  
2) 评分降一级
- 大作业: 学期中间发布, 期末考试前截止.
- 如果参考他人作业或者网站代码, 请在程序注释中标明.

## □ 课程内容多, 进度快

- 部分内容需自学

## □ 快速学习如何编写实用的C++程序













- 从一开始就使用高级数据结构(标准库), 后面再解释这些数据结构的基础.
- 关注如何解决问题, 而不是探讨语言和标准库的特征.

## □ 抽象——可选择的忽略

- 忽略细节的能力: 成熟技术的特征.
- 适当地设计和选取抽象: 即便不理解它们的工作细节, 仍然能使用它们.

## □ 注意事项

- 这门课不会覆盖C++的所有知识, 但会讲授其中最重要的部分.
- 全部内容均遵循C++编程规范.
- 已有的C知识用处不大.

 C++	 JavaScript
 Java/C#	 PHP(Without MySQL)
 Ruby	 Pascal
 Perl	 Lisp
 Visual Basic	 Haskell
 Python	 C

\* <http://www.dataguru.cn/thread-190925-1-1.html>

复旦大学版权所有

## □ 两部分内容

### 1. 使用标准库编写程序

- 无需知道实现细节, 即可编写实用的C++程序.

### 2. 如何定义抽象

- 理解如何使用标准库→学习标准库所依赖的底层技术→编写自己的类.

# 第 0 章 入 门

---





# a small C++ program

```
#include <iostream>
int main()
{
    std::cout << "Hello, world!" << std::endl;
    return 0;
}
```

C:\Users\comet\Documents\C++\Examples\Chapter00\hello.exe

Hello, world!

1) `//`: 单行注释

2) `#include`指令

- 输入/输出是标准库的一部分, 而不是语言核心的内容.
- `iostream`: 支持流输入/输出的标准头文件.



HelloWorld

### 3) main函数

- 每个C++程序都必须包含一个main函数.
- return语句: 结束当前执行的函数, 返回到调用处(C++实现本身).
- 返回值: 整数类型, 指示程序是否成功执行.
  - 0: 运行成功.
  - 其他值: 有错误.

## 4) 使用标准库输出

```
std::cout << "Hello, world!" << std::endl;
```

- 名字空间(namespace): 相关名字的集合.
- std: 包含标准库定义的所有名字.

e.g. 标准输入/输出库iostream定义了cout和endl⇒通过std::cout和std::endl使用它们.

- std::cout: 标准输出流
- std::endl: 结束当前的输出行.



# 深入分析"Hello, world!"程序

## □ 表达式及其副作用

- 请求系统计算, 生成一个结果.
- 副作用: 影响程序或系统的状态, 且不是运算结果的直接作用.

[例1] 表达式 $3+4$ , 运算结果为7, 没有任何副作用

[例2] 表达式`std::cout << "Hello, world!" << std::endl`

- 运算结果: `std::cout`, 被忽略
- 副作用: 在标准输出流上输出"Hello, world!"并结束当前行.

## □ 表达式: 操作符和操作数

```
std::cout << "Hello, world!" << std::endl
```

- 操作数的类型决定了作用于它的操作符所产生的结果.
- std::cout的类型是std::ostream.
- <<: 流输出操作符, 支持链式输出操作.
- "Hello, world!": 字符串常量(string literal).
- std::endl: 控制符 (manipulator), 控制流输出.

## □ 作用域(scope)

1) std名字空间: 避免与自定义的名字冲突.

■ 作用域运算符::(scope operator)

□ 左边: 作用域名称

□ 右边: 该作用域中定义的名字

2) 花括号也是一种作用域

□ 每个函数都是一个作用域.

□ 每个复合语句也是一个作用域.



# 小结

## □ C++程序具有自由风格

- 不具有自由风格(不能跨行)的三种实体: 1) 字符串常量; 2) `#include name`; 3) `//注释`.

## □ C++有两种类型

- 内置于语言核心的类型, e.g. `int`, `char`, `double`, ...
- 语言核心之外定义的类型, e.g. `std::ostream`.

- 名字空间
- 名字定义和头文件
- main函数
  - 系统通过调用main函数执行程序.
  - ⚠ 好习惯: 在main函数中显式包含return语句.
- 表达式语句