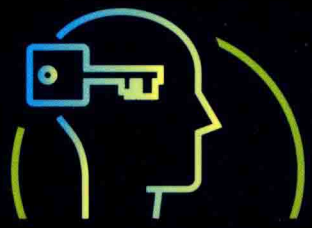


程序员修炼之道

——程序设计入门30讲

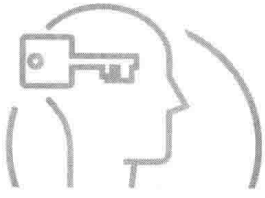
◎ 吕云翔 傅义 主编

目标性强 针对编程初学者，帮助初学者跨越编程的第一道门槛。

问题典型，回答生动 采用一问一答的编写形式，解决编程时最容易遇到的典型问题。

章节独立 可以任意选择感兴趣的章节进行阅读。

清華大學出版社



程序员修炼之道

——程序设计入门30讲

◎ 吕云翔 傅义 主编

清華大學出版社
北京

内 容 简 介

本书收录了与程序设计基础知识相关的 30 个问题。它们是大部分初次接触编程的读者共有的问题。这些问题的答案并不复杂，但是消化吸收它们却不是一个简单的过程。这需要读者培养计算思维，学习从程序的视角看问题。当你可以回答本书所有的问题时，相信你已经越过了程序设计的第一道门槛。

本书分为 6 部分，分别是：入门学堂、内存模型、初窥算法、面向对象、认识程序、编程之道。在入门学堂这部分中，主要介绍程序设计最基础的知识，例如如何编写第一个 Java 程序、第一个 C++程序，学习调试程序等。在内存模型这部分中，我们将学习指针、引用、栈和堆、参数传递等内存相关的知识。初窥算法部分围绕基础的数据结构和算法展开，如链表、递归算法、搜索算法等。在面向对象这部分中，我们将围绕面向对象程序设计的三大特性展开学习。认识程序部分则介绍更多程序设计方面的知识，如异常处理机制、输入输出流、多线程编程等。编程之道部分讲述提升代码质量的方法，编程不仅是一项工程性的工作，更是一项艺术工作，这一部分就围绕程序设计的艺术性来展开。

本书面向所有计算机相关专业的学生，也面向所有对程序设计感兴趣的入门学习者，只要对本书中的任何问题感到疑惑，并且想知道背后答案的读者，都可以阅读本书。

图书在版编目（CIP）数据

程序员修炼之道：程序设计入门 30 讲 / 吕云翔，傅义主编. —北京：清华大学出版社，2018

ISBN 978-7-302-49928-2

Ⅰ. ①程… Ⅱ. ①吕… ②傅… Ⅲ. ①程序设计-问题解答 Ⅳ. ①TP311.1-44

中国版本图书馆 CIP 数据核字（2018）第 064701 号

责任编辑：魏江江　薛　阳
封面设计：刘　键
责任校对：胡伟民
责任印制：刘海龙

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn, http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：北京鑫海金澳胶印有限公司
经　　销：全国新华书店
开　　本：185mm×240mm　　**印　　张**：10.75　　**字　　数**：170 千字
版　　次：2018 年 7 月第 1 版　　**印　　次**：2018 年 7 月第 1 次印刷
印　　数：1～2000
定　　价：39.50 元

产品编号：075267-02

前　言

计算机科学是一门专业性很强的学科，该学科思考问题、解决问题的独特方式将很多初学者拦在了门外。还记得高中刚接触力学的时候，很多题目让笔者望而却步，经过了反复琢磨，笔者才领悟到受力分析这一根本方法的诀窍，在此之后，所有的题目仿佛一下子变得简单了许多。相比物理，计算机的概念显得更为抽象，入门门槛也因此更高。不同的初学者因天赋不同，在入门这一过程中花费的时间长短不一。然而天才毕竟是少数，很多读者在建立计算思维的过程中遭遇重重困难，一部分读者甚至中途放弃。

当笔者在越过了阻碍初学者入门的这道门槛之后，回过头来看那些当初困扰笔者的问题，似乎并没有什么特别难的地方。笔者认为，大部分困难并非在于问题本身，难的是通过这些问题培养计算机独特的思维方式。

我们通过对北京航空航天大学大一大二软件工程专业学生的调研，搜集了他们在学习过程中遇到的困扰他们的问题。本书收录了其中出现频率最高的大部分问题，例如：什么是指针？对象是如何传递的？为什么静态方法不能调用非静态成员？编译和链接阶段发生了什么？等等。本书分为六部分，分别是：入门学堂、内存模型、初窥算法、面向对象、认识程序、编程之道。在入门学堂这一部分中，我们将学习程序的基本概念，掌握编程的基本方法。内存模型部分则涉及计算机体系结构中较

为重要的一部分——内存的知识，程序运行背后的内存模型是学习编程所需修炼的内功之一。初窥算法部分则介绍编程中常见的算法与数据结构，这是学习编程所需修炼的又一大内功。面向对象部分介绍当下最常见的软件开发方法。认识程序部分是关于程序设计更多的知识介绍，例如多线程编程、异常处理、输入输出等。编程之道部分介绍了编程之道，这些方法更多地是为了帮助我们写出高质量的代码。

本书共收录了 30 个常见的问题，我们认为这些问题是极具代表性的，相信大部分的初学者在遇到这些问题的时候都会想看到这些问题最通俗易懂的解答，而这正是我们撰写本书的目的。无论你是初学者还是已经具备了一定的编程能力的学习者，如果你对本书列出的某些问题还存有疑惑，不妨去阅读一下相应的解答，由于每一个问题都相对独立，读者可以挑选感兴趣的问题进行阅读，而不一定按照顺序从头读到尾。我们希望所有的初学者在阅读完本书之后，能对程序形成一个系统而清晰的认识，成功跨越学习编程的第一道门槛，发现编程的乐趣。

本书具有以下几个方面的特点。

目标性强：本书针对刚刚接触编程的计算机、软件工程相关专业的学生，旨在帮助读者建立计算机专业的思考方式，培养程序员的思维方式。书中收集了大部分初学者都会遇到的问题，通过形象生动的语言进行解答，帮助初学者跨越编程的第一道门槛。

问题典型，回答生动：本书采用一问一答的编写形式，行文类似《十万个为什么》。问题选取计算机相关专业学生在初学编程时最容易遇到的典型问题，范围涵盖内存模型、算法与数据结构、程序设计语言等多个方面。回答采用生动形象的语言，以尽可能多的类比让读者轻松理解问题答案。

受众广泛：本书适合刚接触编程的初学者，包括计算机、软件工程专业大一大二的学生以及热爱编程的自学者。本书也适合学习了编程一段时间的读者，帮助其梳理思路，温故知新。

章节独立：由于本书各章节的问题相对独立，读者可以任意选择感

兴趣的章节进行阅读，而不一定要按顺序从头读到尾，增强了阅读的灵活性和针对性。

本书的作者为吕云翔、傅义，另外，曾洪立、吕彼佳、姜彦华参与了部分内容的写作与资料整理的工作。

由于我们的水平和能力有限，本书难免有疏漏之处。恳请各位同仁和广大读者给予批评指正，也希望各位能将实践过程中的经验和心得分享给我们（yunxianglu@hotmail.com）。

编　者

2018 年 3 月

目　录 ▷▷▷

一、入门学堂

1. #include, using namespace std, int main 分别是什么意思？我的第一个 C 程序
2. import, public static void main, String[] args 分别是什么意思？我的第一个 Java 程序
3. 什么是数据类型？
4. 如何阅读项目源码？
5. 如何调试程序？

1. #include, using namespace std, int main 分别是什么意思？我的第一个 C 程序

本节的目的就是让读者看一看 C++程序长什么样，更重要的，我们希望读者能把原来初学时不明白的地方都弄明白。通过本节，读者会对 C++有一个大体的认识。本节的知识较为基础，如果对于示例代码 1.1 没有任何疑问，完全可以跳过本节。如果对 Java 语言更感兴趣，也可以直接进入下一节。

⋙ Hello world!

相信每个程序员接触的第一个程序都是“Hello world”，我们要认识的第一个 C++程序也不例外。

示例代码 1.1

```
#include <iostream>
#define HELLO_WORLD "Hello world!"
using namespace std;
```

```
int main()
{
    cout<<HELLO_WORLD<<endl;
    return 0;
}
```

文件包含

示例代码 1.1 的第一行#include <iostream>是文件包含指令，该指令的作用是在编译预处理时，将指定源文件的内容复制到当前源文件中，如图 1.1 所示。以示例代码 1.1 为例，在该段代码被编译之前，iostream 文件内容会被复制到当前文件的起始位置，替代原先的#include <iostream>。为什么要在文件的第一行写这样一句指令呢？我们希望在屏幕上打印“Hello world”，就需要用到标准输出 cout，这是一个负责程序对外输出的对象，而该对象是在 iostream 文件中定义的。简单地说，iostream 文件为我们提供了输入输出功能。

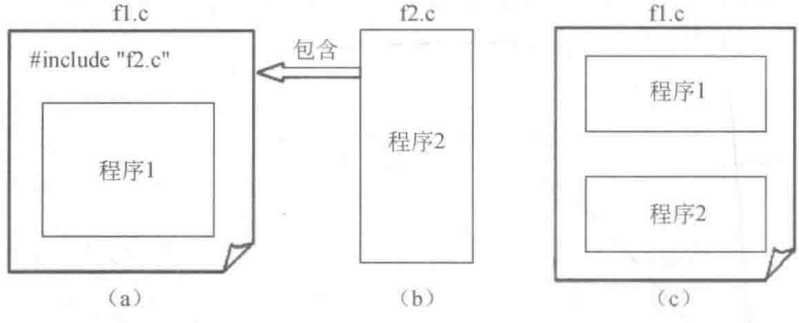


图 1.1　文件包含指令作用示意图

读者你也许注意到了，在#include <iostream>后面并没有添加分号，所以这一行并不是一条 C 语句，而是一个预处理指令。预处理指令是编译器在将程序编译为机器语言之前首先会对程序进行的预处理。常见的预处理指令包括文件包含、宏定义和条件编译，接下来我们进一步了解宏定义的概念。

宏定义

示例代码 1.1 的第二行#define HELLO_WORLD "Hello world!"是一条宏

定义，该指令的作用是在编译预处理时，将源文件中所有的 HELLO_WORLD 都替换为"Hello world!"，于是示例代码 1.1 的第 6 行 cout<<HELLO_WORLD <<endl;会变为 cout<<"Hello world!"<<endl;。宏定义也是一种预处理指令，该指令在编译器编译之前被执行。

很多初学 C 语言的同学分不清宏定义与 const 常量的区别。宏定义只是在编译预处理阶段进行替换，并不会在内存中生成对应的变量。而 const 常量是一个在内存中分配了空间的只读变量。所以这两者有本质上的区别。

⋙ 命名空间

示例代码 1.1 的第三行 using namespace std;表示使用命名空间 std。命名空间是指各种标识符的可见范围。C++标准程序库中的所有标识符都被定义在一个 std 的命名空间中。如果不在示例代码中使用 using namespace std;这一行语句，想要使代码通过编译，就需要将示例代码第 6 行的 cout<<HELLO_ WORLD<<endl;修改为 std::cout<<HELLO_WORLD<<std::endl;。

我们可以将命名空间想象成区号，将类名想象为一个电话号码。由于各省市的电话号码可能重复，就通过在电话号码前面加上区号使得该号码成为一个独一无二的表示。Java 中也有类似的机制，包名就如同区号，类名就如同电话号码。

⋙ main 函数

接下来代码进入了主体部分——main 函数。main 函数是 C++程序的入口函数，是程序执行的起点。该函数与其他的函数在形式上没有什么区别，也由返回类型、函数名和函数参数组成。

返回类型：C++规定，main 函数返回类型是 int 型。返回值用于告诉程序的调用者（即操作系统），程序的退出状态。若返回 0，则表示程序正常退出，若返回其他非 0 值，表示程序异常退出，返回其他数字的含义由系统决定。所以在示例代码 1.1 的第 7 行，我们定义了语句 return 0;用来告诉操作系统，函数正常执行完毕。该返回值并不属于打印到屏幕上的内容，很多初学

者在一开始会混淆返回值和标准输出的概念。

函数参数：C++中，main 函数一共有以下两种定义方式：

```
int main( )
int main( int argc, char *argv[] )
```

示例代码 1.1 采用的是第一种定义方式，即没有函数参数。本章最后的进阶部分给出了采用第二种定义方式的示例代码 1.2，并阐述了函数参数所表达的意义。

函数主体：我们通过语句 cout<<HELLO_WORLD<<endl;打印"Hello world!"到屏幕。cout 是标准输出流对象，调用后会向输出设备输出内容；<<负责向对象 cout 发送输出的字符串；endl 也是 iostream 中定义的一个对象，向标准输出发送 endl 类似于在控制台窗口中按下 Enter 键。

示例代码 1.1 的运行结果如图 1.2 所示。

图 1.2　示例代码 1.1 的运行结果

进阶

对于 main 函数的第二种定义方式，argc 表示传入 main 函数的参数的个数，argv[]存放着这些参数，在 argv[]的这些参数中，第一个参数是程序的全名。我们提供一个以第二种方式定义 main 函数的程序，见示例代码 1.2。

示例代码 1.2

```
#include <iostream>
using namespace std;
int main(int argc, char *argv[])
{
```

```
    cout<<argv[1]<<" "<<argv[2]<<endl;
    cout<<argv[0]<<endl;
    return 0;
}
```

在示例代码 1.2 的第 5 行 cout<<argv[1]<<" "<<argv[2]<<endl;中，我们向标准输出打印了 argv[]的第二和第三个参数，而在第 6 行，我们向标准输出打印了 argv[]的第一个参数，为了验证该参数即为程序的全名。

我们在 Visual Studio 中设置向 main 函数传入的参数，第一个参数为 Hello，第二个参数为 world!，两个参数之间以空格隔开，如图 1.3 所示。

示例代码 1.2 的运行结果如图 1.4 所示。

图 1.3　main 函数参数设置

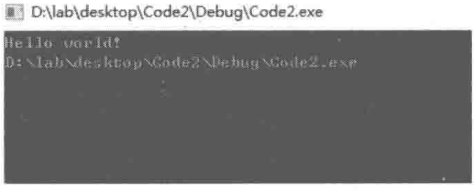


图 1.4　示例代码 1.2 的运行结果

如图 1.4 所示，程序第一行输出了"Hello world!"，即我们向 main 函数传递的参数。程序第二行输出了程序的全名，即 argv[0]。

2. import, public static void main, String[] args 分别是什么意思？我的第一个 Java 程序

在第 1 节中，我们已经认识了第一个 C++程序，通过该程序我们在屏幕上打印了“Hello world!”。本节中我们将学习第一个 Java 程序，通过这一节的学习，读者会初步认识 Java 的包机制、类定义和 main 函数。

⋙ Hello world

在第一个 Java 程序中，我们要完成的工作仍然是向屏幕输出“Hello

world!”。在这里我们故意把打印“Hello world!”的方法变得稍微复杂了些，目的是让读者认识一个更完整的程序。

示例代码 2.1

```
package program.chapter2;
import java.util.List;
import java.util.ArrayList;
public class Code1 {
   public static void main(String[] args){
       List<String> argsList = new ArrayList<String>();
       for(String arg : args){
           argsList.add(arg);
       }
       System.out.println(argsList);
   }
}
```

⋙ package 语句

程序的第一行是 package 语句，该语句的作用是规定当前类属于哪个包。

在 Java 中，同一个包中存放的类是功能相关的，包机制使得项目代码存放在一个合理有序的组织结构下，便于开发人员管理。

同时，包机制提供了类的多层命名空间，这一点与 C++中的命名空间类似，用于解决类的命名冲突。我们也许会遇到类名完全相同的两个类，例如有两个类的类名都是 A，这时候不同的包名为这两个类提供了不同的命名空间，我们就能通过包名告诉计算机我们使用的到底是哪个类了。若用电话号码做类比，包名即为区号，类名即为电话号码。包名一般全是小写字母，由一个或多个有意义的单词连缀而成，命名规则是：域名倒写.项目名.模块名.组件名。例如我们会发现有些包以 org.apache 打头，其对应的域名就是 apache.org。

⋙ import 语句

接下来的一行是 import 语句。我们在编写一个类时，经常会用到其他的

类，要正确引用这些类，就需要用 import 语句进行导入声明。在示例代码 2.1 中，我们为了使用 java.util.List 类，定义了 import java.util.List;语句。如果不在程序起始处定义 import 语句，程序中所有用到 List 类的地方都需要使用该类的全名，这就会使代码显得非常冗长。

进阶

一个 Java 编程高手通常对 Java 常用包非常熟悉，了解 Java 提供了哪些包，能够帮助自己知道利用 Java 可以实现哪些功能，而哪些功能实现起来是较为困难的。Java 的常用包如下。

java.lang：Java 语言的核心，提供了 Java 中的各种基础类。

java.util：实用工具包，提供了各种功能。

java.net：提供了网络编程相关的各种类。

java.io：包含了输入输出操作相关的类。

java.sql：包含了数据库编程相关的类。

java.awt：提供了用于构建图形用户界面的类。

感兴趣的读者可以通过阅读源码深入了解 Java 的包机制。

类定义

在定义了 package 语句和 import 语句之后，程序进入了主体部分，即对类的定义。当编写一个 Java 源代码文件时，此文件通常被称为编译单元，每个编译单元都必须有一个扩展名.java，而在编译单元内则至多可以有一个 public 类，该类的名称必须与文件的名称完全相同，包括大小写在内。

Java 中，类（内部类除外）有两种访问权限：

（1）public 访问权限。可以供所有类访问。

（2）默认访问权限。同一个包中的类可以访问该类，即包级访问权限。

在示例代码 2.1 中，我们定义的类名是 Code1，其访问权限是 public 级别的。如果读者想要了解关于面向对象更深入的知识，可以阅读本书的第四部分。

main 函数

类似 C 语言程序，main 函数也是 Java 程序的执行入口。main 函数与其他函数在形式上并无差异，也是由返回类型、修饰符、参数等构成的。下面以示例代码 2.1 为例，介绍 main 函数的各个组成部分。

返回类型：void。Java 程序中的 main 函数返回值必须为空，不允许为 int 或其他类型。

访问修饰符：public。为了使得该 main 函数可以直接被系统调用，必须设置访问修饰符为 public。

类修饰符：static。static 修饰符表明该函数类静态函数，即函数是属于类的，而不是属于对象的。因为 main 函数是程序的入口函数，系统是通过类来调用该 main 函数，而不是通过该类的任何对象来调用该 main 函数，所以必须设置类修饰符为 static。关于静态方法更深入的知识，感兴趣的读者可以阅读本书的第 20 节。

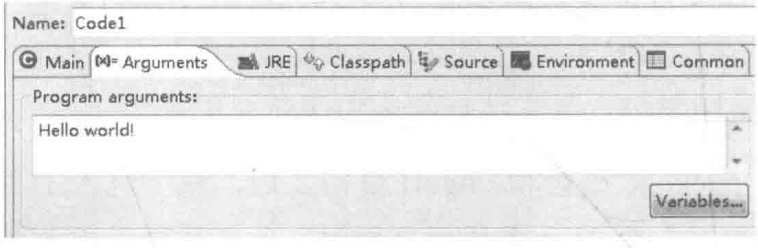


图 2.1　main 函数参数设置

参数：Java 中，main 函数的参数是一个 String 数组。该数组内容是用户在运行程序时设置的。用户可以通过 Eclipse 的 run configuration 设置 Arguments 为 Hello world!，如图 2.1 所示。这一方法类似于第 1 节中 Visual Studio 为 main 函数设置参数的过程。

函数主体：示例代码 2.1 在 main 函数中首先生成一个 List 对象，然后循环遍历 main 函数参数 args，将数组中的每个元素添加到 List 对象中，最后将 List 对象直接打印到控制台。如图 2.1 所示，我们向示例代码 2.1 的 main 函

数传入的参数是“Hello”和“world!”，所以控制台成功打印出了“Hello world!”，如图 2.2 所示。由于我们是直接将 List 对象打印到控制台的，所以输出的字符串包含了中括号，并且在元素之间通过逗号进行了连接。

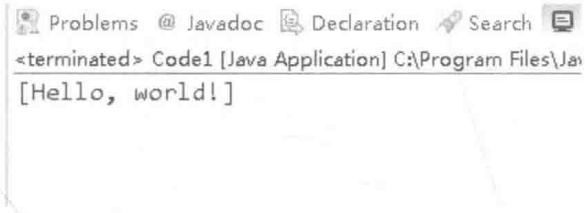


图 2.2　示例代码 2.1 的运行结果

3. 什么是数据类型?

对初学者来说，理解数据类型可能是一个难题。我们已经知道 int 代表整数，char 代表字符，float 代表浮点数，但是这些数据类型在内存中是如何存储的？数据类型对于计算机有什么意义？我们还不是十分清楚。

有些人的观点是，理解数据类型是一个循序渐进的过程，一开始的不理解并不会阻碍初学者打好编程基础，随着编写的代码越来越多，再来学习内存的知识，会更加容易。我也十分认同这种观点，数据类型及内存方面的知识不是一下可以吃透的，但如果读者仍充满了好奇，坚持要理解数据类型，阅读本节也是一个不错的选择。

⋙ 定义

在开始进入本节的学习之前，让我们先来看一下数据类型的定义，尽管定义可能有些枯燥：数据类型在数据结构中的定义是一个值的集合以及定义在这个值集上的一组操作。变量是用来存储值的所在处，它们有名字和数据类型。变量的数据类型决定了如何将代表这些值的位存储到计算机的内存中。