

大学计算机基础教育规划教材

可视化计算

程向前 陈建明 编著



1+X

清华大学出版社

大学计算机基础教育规划教材

可视化计算

程向前 陈建明 编著

清华大学出版社

北京

清华大学出版社

北京

内 容 简 介

本书系统地介绍了使用计算思维进行计算问题求解的核心理念,全书由8章组成,主要内容包括程序设计基础、算法分析要点、基本算法与策略、计算与数据模型、具有数据结构特色的算法以及计算工具的选择和评估。本书以最终用户开发为前提,着重阐述使用计算机科学的基本概念与方法设计计算过程,并充分利用软件工具的特点做到算法设计、运算过程和计算结果的可视化,使用估算和实验方法来分析和验证算法实现的效率。本书的算法设计采用了基于流程图的算法原型设计工具 RAPTOR,附录介绍了 RAPTOR 的图形操作。

本书可以作为高校非计算机专业的主要介绍计算思维或计算原理的计算机基础类课程的教材,也可以作为非计算机专业的程序设计、数据结构和离散数学等课程的入门参考书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

可视化计算/程向前,陈建明编著.--北京:清华大学出版社,2013.1

大学计算机基础教育规划教材

ISBN 978-7-302-30473-9

I. ①可… II. ①程… ②陈… III. ①可视化软件—软件开发—高等学校—教材 IV. ①TP311

中国版本图书馆 CIP 数据核字(2012)第 250863 号

责任编辑:焦虹 战晓雷

封面设计:常雪影

责任校对:李建庄

责任印制:王静怡

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

课件下载: <http://www.tup.com.cn>, 010-62795954

印 装 者:清华大学印刷厂

经 销:全国新华书店

开 本:185mm×260mm

印 张:23

字 数:569千字

版 次:2013年1月第1版

印 次:2013年1月第1次印刷

印 数:1~2500

定 价:39.00元

产品编号:048259-01

序

进入 21 世纪,社会信息化不断向纵深发展,各行各业的信息化进程不断加速。我国的高等教育也进入了一个新的历史发展时期,尤其是高校的计算机基础教育,正在步入更加科学,更加合理,更加符合 21 世纪高校人才培养目标的新阶段。

为了进一步推动高校计算机基础教育的发展,教育部高等学校计算机科学与技术教学指导委员会近期发布了《关于进一步加强高等学校计算机基础教学的意见暨计算机基础课程教学基本要求》(以下简称《教学基本要求》)。《教学基本要求》针对计算机基础教学的现状与发展,提出了计算机基础教学的指导思想;按照分类、分层次组织教学思路,《教学基本要求》提出了计算机基础课程教学内容的知识结构与课程设置。《教学基本要求》认为,计算机基础教学的典型核心课程包括大学计算机基础、计算机程序设计基础、计算机硬件技术基础(微机原理与接口、单片机原理与应用)、数据库技术及应用、多媒体技术及应用、计算机网络技术及应用。《教学基本要求》中介绍了上述六门核心课程的主要内容,这为今后的课程建设及教材编写提供了重要的依据。在下一步计算机课程规划工作中,建议各校采用“1+X”的方案,即“大学计算机基础”+若干必修或选修课程。

教材是实现教学要求的重要保证。为了更好地促进高校计算机基础教育的改革,我们组织了国内部分高校教师进行了深入的讨论和研究,根据《教学基本要求》中的相关课程教学基本要求组织编写了这套“大学计算机基础教育规划教材”。

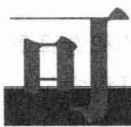
本套教材的特点如下:

- (1) 体系完整,内容先进,符合大学非计算机专业学生的特点,注重应用,强调实践。
- (2) 教材的作者来自全国各个高校,都是教育部高等学校计算机基础课程教学指导委员会推荐的专家、教授和教学骨干。
- (3) 注重立体化教材的建设,除主教材外,还配有多媒体电子教案、习题与实验指导,以及教学网站和教学资源库等。
- (4) 注重案例教材和实验教材的建设,适应教师指导下的学生自主学习的教学模式。
- (5) 及时更新版本,力图反映计算机技术的新发展。

本套教材将随着高校计算机基础教育的发展不断调整,希望各位专家、教师和读者不吝提出宝贵的意见和建议,我们将根据大家的意见不断改进本套教材的组织、编写工作,为我国的计算机基础教育的教材建设和人才培养做出更大的贡献。

“大学计算机基础教育规划教材”丛书主编
教育部高等学校计算机基础课程教学指导委员会副主任委员

冯博琴



本书是一个新的计算机教育时代的产物,计算思维就是这个时代的名称。一般认为科学思维主要分为理论思维、实验思维和计算思维三大类,所以理论、实验和计算成为推动人类文明进步和科技发展的三大支柱。而计算思维的主要内容则包括了问题求解、系统设计和人类行为理解。

在高等院校的计算机教育领域,一直存在两种截然不同的教材:一种是面向计算机和信息专业领域的,众多成熟和系统性很强的教材并形成学科教育的基础;另一种则是面向所有专业的计算机基础教材,由于其中的内容大部分是面向计算机基本应用的,所以技术选择对平台工具具有一定的依赖性,而且随着技术的变换而不断变化。

由于计算机与网络技术的迅猛发展,技术和平台工具选择的相互依赖会导致大部分专业工作者在学校所接触到的信息技术和产品不久就成为“明日黄花”并为更新的技术和产品所替代,甚至一种计算工具尚未完全掌握,新一代的工具又产生了。这种状况往往令人目不暇接,以至于无所适从。另一方面,计算机科学的基础内容,那些在计算机科学课程系列中被认为极为重要的核心理念和基本方法,虽然变化缓慢,但由于其枯燥艰深,难以为广大的其他专业工作者所了解和掌握。

在这个新的计算机教育时代,大学计算机课程的重要任务之一是让学习者体验到计算原理的相互影响以及问题有效解决的思维方式,并从中获得计算的愉悦。

本书的目的就是朝着这个方向努力。希望在有限的时间内,将计算机科学的核心理念传播给非计算机专业的读者,并且,希望这些理念是可以观察、验证和投入实际应用的——即使这种应用水平还属于初级阶段,可能存在很多需要改进的地方。

鉴于目前大部分高校学生具备的计算机和信息应用技术和认知基础,本书的重点放在以下方面。

(1) 程序设计基础:本书对读者的程序设计知识的要求为“零起点”,而介绍算法显然离不开最基本的程序设计概念。但本书的主旨并不是介绍程序设计,所以程序设计语言的选择和应用采用了不同于以往传统教材的做法,目的是用最短的时间让读者掌握一种可以进行算法讨论的程序设计工具。这部分内容包含在第1、2章中。

(2) 算法分析要点:算法是计算机科学的精髓,也是本书所强调的核心内容。算法分析的要点是通过理论估算和实验验证两种方式,比较算法设计的时间和空间复杂性;让读者熟悉和了解在计算机科学中算法设计普遍应用的“空间换时间”的基本原则。这部分出现在第2章。

(3) 基本算法与策略:与高校学生的计算机与信息基础结合,进行常用基本算法的

介绍,让读者可以直观地感受使用计算机进行问题求解的魅力,并养成分析问题和选择算法的习惯;而基本策略的引入则是计算思维的直接体现,可以让读者感受计算机科学中应用人类传统智慧的一些基本原则,并形成计算策略应用的基本规律。这部分内容包含在第3章中。

(4) 计算与数据模型:模型化的思想是科技发展的重要手段,这部分内容主要考虑两个方面:一是以图灵机为核心的计算理论的引入和验证,为读者提供有限状态机算法设计的基本训练;二是开始引入抽象数据类型,因为计算科学的两个核心概念——抽象和自动化都涉及基础理论和应用实现。这部分内容出现在第4章。

(5) 具有数据结构特色的算法:内容包括在第5~7章中,每一章的内容都与某种抽象数据类型相关,是本书实践计算思维教学的问题求解部分内容最为集中的部分,具有较强的实践性和应用价值。

(6) 计算工具的选择和评估:计算思维必定受到人和计算工具的能力限制。这部分介绍计算工具的选择原则和评估方法,以便读者明确认识计算工具的限制和最大限度地利用计算工具本身的能力,这也是本书的特殊视角的反映。由于作为算法评估工具的可视化程序设计语言除了进行算法原型的设计和分析之外,也可以将算法设计的结果转换成为其他的程序设计语言。这样,除了能够完成基本的算法功能外,符合应用需求的程序和计算环境就有了选择的必要。这部分内容出现在第8章。

本书可以作为一门类似于“大学计算机”性质课程的基础教材,供大学本科一、二年级的学生使用。同时也可以作为程序设计、数据结构和离散数学等课程的入门参考书。

作者谨向全体参与以“计算思维”为切入点的大学计算机课程和教学改革的同行致以敬意。采用一种全新的视角来开展这类课程的教学,不仅需要辛勤的付出和投入,而且需要具备挑战未知领域的勇气和魄力。

经过两年多的准备和实践,作者发现这种付出是非常值得的——当学生拿着自选专题的算法大作业,同教师和同学进行交流的时候,你会突然发现,在学期开始时甚至不知道分支、循环为何物的学生,到了期末已经可以与他人讨论非常专业的算法问题了。这是大学计算机基础课教师所追求的最高境界和所能获得的最好奖励。

在本书编写过程中,得到了国内外专家学者以及我们的学生的帮助,在此谨致谢意。

西安交通大学计算机教学实验中心主任、全国首届教学名师冯博琴教授在作者任教的单位中倡导开展以“计算思维”为切入点的新一轮大学计算机课程教学改革,并亲自推动了少年班“计算概论”、“大学计算机基础”的课程改革实验,使得作者有机会进行改革方案的创新和实践。

可视化程序设计工具 RAPTOR 的主创者之一 Martin C. Carlisle 博士,对作者提出的对该工具的设计改进做出了详尽和积极的答复。

西安交通大学李仁厚教授、黄尚恒教授、吴裕远教授评审了采用本书作为教材的校级选修课的大纲,并建议修改课程名称,本教材最后成书时也采用了该课程修改后的名称。

感谢上海交通大学过敏意教授、福州大学陈国龙教授、重庆邮电大学王国胤教授、西安交通大学邓建国教授、广东工业大学程良伦教授、湖南大学李仁发教授、桂林电子科技大学董荣胜教授、华南师范大学叶惠文教授、成都理工大学孙淑霞教授、深圳大学王志强

教授、肇庆学院胡忠望教授、华东理工大学陈志华副教授针对本书所提出的建设性意见和建议。

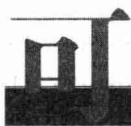
特别感谢西安交通大学周梦远同学、西安电子科技大学薛飞洁同学为本书中的程序样例的设计与调试提供了重要的帮助以及西安交通大学少年 103 班、少年 104 班全体同学为“计算概论”课程的算法设计大作业所付出的努力。

感谢清华大学出版社焦虹主任和张民编辑对本书立项和出版所做的努力。

对于本书存在的缺陷和不足,作者诚恳地期望各位同行和读者赐教。作者的电子邮件地址是 xqcheng@mail.xjtu.edu.cn。

作者

2012 年 11 月于西安交通大学



第 1 章	RAPTOR 程序设计基础	1
1.1	程序与算法的基本概念	1
1.1.1	为什么要学习程序设计	1
1.1.2	程序设计的发展阶段	2
1.1.3	为什么要使用 RAPTOR 可视化程序设计环境	6
1.1.4	程序设计语言的基本概念	8
1.1.5	RAPTOR 的特点	10
1.2	RAPTOR 基本程序环境	10
1.2.1	基本符号	10
1.2.2	变量	11
1.2.3	常量	15
1.2.4	输入语句	15
1.2.5	数据处理语句	16
1.2.6	过程调用语句	19
1.2.7	输出语句	20
1.2.8	注释	21
1.3	RAPTOR 控制结构	23
1.3.1	顺序控制	23
1.3.2	选择控制	23
1.3.3	决策表达式	24
1.3.4	循环控制	27
1.4	RAPTOR 数组变量	31
1.4.1	一维数组的创建	32
1.4.2	二维数组的创建	33
1.4.3	数组的运算	34
1.4.4	如何使用数组变量	34
1.4.5	平行数组	35
1.4.6	数组应用中的注意事项	35
1.5	RAPTOR 子程序和子图的定义及调用	35
1.6	计算问题求解的一般过程	40

1.6.1	理解问题	40
1.6.2	制定计划	41
1.6.3	执行计划	42
1.6.4	回顾与总结	42
1.6.5	使用计算机进行问题求解：程序开发周期	43
1.7	小结与回顾	44
	习题	44
第2章	算法设计与可视化	46
2.1	算法初步	46
2.1.1	算法的由来与定义	46
2.1.2	算法的特性	49
2.1.3	算法设计的要求	49
2.2	算法效率的度量	51
2.2.1	算法效率的基本估算方法	51
2.2.2	实验验证方法	53
2.3	算法复杂度	53
2.3.1	函数的渐近增长	54
2.3.2	算法时间复杂度定义	54
2.3.3	推算大 O 阶方法	55
2.4	计算的可视化问题	57
2.4.1	算法设计的可视化	57
2.4.2	计算过程的可视化	58
2.4.3	计算问题和结果的可视化	60
2.4.4	RAPTOR 与流程图规范之间的关系与差别	61
2.5	RAPTOR 算法设计常用子程序	63
2.5.1	随机数的产生与存储	63
2.5.2	将计算结果存储到文件	64
2.5.3	从文件中读入基础数据	65
2.5.4	RAPTOR 图形输出	67
2.5.5	RAPTOR 图形窗口中的用户交互	74
2.5.6	图形输出案例：随机漫步的模拟模型	75
2.6	小结与回顾	81
	习题	81
第3章	基本算法和策略	83
3.1	基本算法	83
3.1.1	蛮力法	83
3.1.2	分段函数	84
3.1.3	递推法	86

3.1.4	模运算	89
3.1.5	字符和字符串运算	89
3.1.6	递归	93
3.1.7	数论问题	98
3.1.8	组合计算	102
3.1.9	迭代法	105
3.2	基本策略	108
3.2.1	贪心策略	108
3.2.2	分治策略	110
3.2.3	回溯策略	111
3.2.4	动态规划	114
3.2.5	将递归算法转化为非递归的实现	117
3.2.6	基本算法策略的讨论	119
3.3	小结与回顾	121
	习题	121
第 4 章	模型化	124
4.1	从有限状态机到图灵机	124
4.1.1	什么是模型	125
4.1.2	如何建立模型	126
4.1.3	什么是有限状态机	126
4.1.4	如何设计和应用有限状态机	129
4.1.5	可视化有限状态机的实现案例：电子宠物游戏	130
4.1.6	什么是图灵机	136
4.1.7	如何使用 RAPTOR 模拟图灵机	139
4.1.8	有限状态机与图灵机的比较	148
4.2	使用 RAPTOR 实现抽象数据类型	149
4.2.1	对现实世界中的问题进行数据抽象	149
4.2.2	使用 RAPTOR 实现线性表	151
4.2.3	使用 RAPTOR 实现树	155
4.3	小结与回顾	164
	习题	164
第 5 章	排序与查找	166
5.1	排序	166
5.1.1	计算机如何进行排序	167
5.1.2	直接插入排序	169
5.1.3	桶排序	172
5.1.4	冒泡排序	172
5.1.5	快速排序	176

5.1.6	归并排序	178
5.1.7	排序算法的分析	178
5.2	查找	180
5.2.1	顺序查找	180
5.2.2	二分查找	181
5.2.3	分块查找	182
5.2.4	哈希查找	188
5.2.5	查找算法的分析	198
5.3	小结与回顾	199
	习题	199
第 6 章	信息论、哈夫曼编码与二叉树	201
6.1	问题的引入	201
6.1.1	信息表达	203
6.1.2	数字表达	204
6.1.3	字符表达	205
6.2	信息论基础	207
6.2.1	什么是信息	207
6.2.2	信息的量化	208
6.3	哈夫曼编码	212
6.3.1	哈夫曼编码原理	213
6.3.2	使用 RAPTOR 实现哈夫曼编码的算法	215
6.4	二叉树	220
6.4.1	二叉树的遍历	224
6.4.2	堆排序	227
6.4.3	二叉搜索树	228
6.4.4	平衡二叉树	236
6.5	小结和回顾	240
	习题	241
第 7 章	图论基础与应用	243
7.1	图的定义和常用术语	244
7.2	图的存储	246
7.2.1	邻接矩阵存储原理	246
7.2.2	使用 RAPTOR 建立已知图的邻接矩阵	248
7.2.3	邻接表存储原理	249
7.2.4	使用 RAPTOR 建立已知图的邻接表	252
7.3	图的遍历	253
7.3.1	深度优先搜索	254
7.3.2	广度优先搜索	255

7.3.3 求图的连通分量	256
7.4 图算法的应用	258
7.4.1 最小网络建设成本	258
7.4.2 最短的旅行路线	260
7.4.3 地图着色问题	266
7.4.4 商业网点的最省布点方案	269
7.5 在 RAPTOR 中实现图算法的可视化	271
7.5.1 图类算法的问题可视化	271
7.5.2 真实的地图与抽象图的叠加	274
7.5.3 棋类游戏的实现	276
7.6 小结与回顾	279
习题	279
第 8 章 计算工具的评估和选择	281
8.1 计算工具的精度评估	281
8.1.1 误差的来源	282
8.1.2 绝对误差、相对误差和有效数字	283
8.1.3 计算工具的精度设计与误差	284
8.1.4 误差分析与实践	289
8.2 从 RAPTOR 到 C++ 的算法环境转换	290
8.2.1 递归程序	290
8.2.2 文件输入输出	292
8.2.3 图形问题	294
8.2.4 RAPTOR 的转换和编译问题	297
8.2.5 RAPTOR 与 C/C++ 的比较与选择	300
8.3 面向对象的程序设计与 RAPTOR	303
8.3.1 OOP 的基本概念	304
8.3.2 封装	304
8.3.3 继承	305
8.3.4 多态性	305
8.3.5 典型的面向对象的程序语言	306
8.3.6 RAPTOR 对 OOP 的支持	306
8.4 逆向工程：从代码到流程图	312
8.4.1 CVF 介绍	312
8.4.2 CVF 流程图与 RAPTOR 的比较	315
8.4.3 AutoFlowchart 介绍	315
8.5 电子表格软件与数值计算	316
8.6 小结与回顾	320
习题	320

附录 A	RAPTOR 图形操作指南	322
A. 1	RAPTOR 图形坐标	322
A. 2	色彩	322
A. 2.1	设置颜色	323
A. 2.2	生成随机色彩	323
A. 3	绘图操作	323
A. 3.1	清理窗口	324
A. 3.2	画弧	324
A. 3.3	画圆	325
A. 3.4	画椭圆	325
A. 3.5	绘制可以旋转角度的椭圆	326
A. 3.6	绘制位图	326
A. 3.7	装载位图	327
A. 3.8	画线	327
A. 3.9	填色	327
A. 3.10	取得像素的色彩	328
A. 3.11	设置像素的色彩	328
A. 4	键盘操作	328
A. 4.1	取键值	329
A. 4.2	取键字符串	329
A. 4.3	判断某个键是否处于按下状态	330
A. 4.4	判断某个击键动作是否已经发生过	330
A. 4.5	等待键入	331
A. 5	鼠标操作	331
A. 5.1	取得鼠标按键与指针位置	331
A. 5.2	取得鼠标指针的 X 值	331
A. 5.3	取得鼠标指针的 Y 值	332
A. 5.4	判断鼠标键是否处于按下状态	332
A. 5.5	判断鼠标键是否被按下过	332
A. 5.6	判断鼠标键是否已经释放	333
A. 5.7	等待某个鼠标按键动作	333
A. 6	文本操作	333
A. 6.1	显示数字	334
A. 6.2	显示文本	334
A. 6.3	取得字模的高度	334
A. 6.4	取得字模的宽度	335
A. 6.5	设置字号	335
A. 7	窗口操作	335

A. 7.1	关闭窗口	335
A. 7.2	取得窗口的最大高度	336
A. 7.3	取得窗口的最大宽度	336
A. 7.4	判断图形窗口是否打开	336
A. 7.5	打开图形窗口	337
A. 7.6	设置窗口的标题栏	337
A. 7.7	平滑动画显示效果	337
术语对照表	339
参考文献	348
参考网站	349

第1章

RAPTOR程序设计基础



本章的学习目标:

- 为什么要学习程序设计?
- 程序设计主要经历了哪些发展阶段?
- 什么是可视化的程序设计环境?
- 为什么要使用可视化程序设计环境?
- 程序设计有哪些基本要素?
- 解释性程序与编译性程序的差别是什么?
- 为何选择 RAPTOR 作为程序和算法的可视化工具?
- RAPTOR 程序设计环境有哪些基本概念?
- 如何使用程序设计进行计算问题的求解?

1.1 程序与算法的基本概念

1.1.1 为什么要学习程序设计

从 20 世纪的 60 年代开始,一场新的信息革命把人类带到信息化社会,而信息技术的核心正是计算机技术。

21 世纪是一个信息化的社会,这个社会对人才素质和知识结构都提出了更高的要求。计算机的作用已不仅仅是一种工具,而是各个学科本身的重要组成部分之一。

作为一名优秀的技术工作者,不懂计算机程序设计,就不能真正理解计算机,也无法在自己所从事的工作领域内深入地应用计算机。

目前,虽然计算机应用软件及工具层出不穷,人们在工作中通常也要开发一些适用于自己的软件,或者针对具体的实际问题对一些软件进行二次开发。对于高等学校的学生,了解计算机科学,使计算机成为一种可以帮助人们思维的工具,显得尤为重要。

计算思维由美国 Carnegie Mellon 大学计算机科学系教授周以真女士于 2006 年提出。她认为,计算思维(computational thinking)是运用计算机科学的基础概念进行问题求解、系统设计以及人类行为理解等的一系列思维活动。计算思维代表着一种普遍的态度和一类普适的技能,每一个人都应热心于它的学习和运用。

周以真认为,计算思维具有以下特征:

- (1) 计算思维是概念化的抽象思维而不只是程序设计。
- (2) 计算思维是基本的而不是死记硬背的技能。
- (3) 计算思维是人的而不是计算机的思维方式。
- (4) 计算思维是数学和工程思维的互补与融合。
- (5) 计算思维是思想而不是人造品。
- (6) 计算思维面向所有的人和所有地方。
- (7) 计算思维关注依旧亟待理解和解决的智力上极具挑战性并且引人入胜的科学问题。

计算思维吸取了问题解决所采用的一般数学思维方法,现实世界中巨大、复杂系统的设计与评估的一般工程思维方法,以及复杂性、智能、心理、人类行为的理解等的一般科学思维方法。

计算思维建立在计算过程的能力和限制之上,由人和机器执行。计算方法和模型使我们敢于去处理那些原本无法由个人独立完成的问题求解和系统设计。

计算思维最根本的内容,即其本质(essence),是**抽象**(abstraction)和**自动化**(automation)。计算思维中的抽象完全超越物理的时空观,并完全用符号来表示,其中,数字抽象只是一类特例。

与数学和物理科学相比,计算思维中的抽象显得更为丰富,也更为复杂。数学抽象的最大特点是抛开现实事物的物理、化学和生物学等特性,而仅保留其量的关系和空间的形式,而计算思维中的抽象却不限于此。

计算思维的核心之一是算法思维,同时算法思维也是计算机科学的精髓。而算法思维的实现离不开程序设计,所以对于计算思维的学习和认识必须从程序设计开始。

1.1.2 程序设计的发展阶段

程序设计是一门技术,也是算法设计的基本工具,需要相应的理论、技术、方法和工具来支持。程序设计主要涉及以下3个问题:

- (1) 做什么:就是程序需要实现的功能。
- (2) 怎么做:就是如何实现程序的功能,在编程中,称为逻辑,即实现的步骤。
- (3) 如何描述:就是把怎么做用程序语言的格式描述出来。

程序设计主要经历了结构化程序设计和面向对象程序设计的发展阶段,程序设计环境则经历了文本化到可视化的发展过程。

1. 结构化程序

1966年,Böhm和Jacopini证明了程序设计语言仅仅使用顺序、选择和重复(循环)3种基本控制结构就足以表达出各种形式结构的程序设计方法。采用结构化程序设计方法编写程序,可使程序结构良好、易读、易理解、易维护,从而可以提高编程工作的效率,降低软件开发的成本。

(1) 顺序结构:是一种简单的程序设计结构,顺序结构自始至终严格按照程序中语句的先后顺序逐条执行,是最基本、最常用的结构形式,如图1-1(a)所示。