

高等院校信息技术课程学习辅导丛书

数据结构学习辅导

宁正元 易金聪 编著



清华大学出版社

高等院校信息技术课程学习辅导丛书

数据结构学习辅导

宁正元 易金聪 编著

清华大学出版社
北京

内 容 简 介

本书是为了配合各级各类学校《数据结构》课程的教学,按照本科教学大纲编写的辅助教材。考虑到信息类专业学生考研的需要和软件水平考试、高新技术考试、全国计算机等级考试的三、四级考试等考生的实际需要,在内容的选材上兼顾了广度和深度的要求。全书内容共分 11 章。前 10 章在概括主要内容和知识点的基础上,先列出各知识点的基础题目,然后给出用于巩固和提高学习效果的应用题、算法设计题和上机实验题。最后 1 章中给出了部分题目的解析或答案。

全书内容丰富,题型多样,涉及面广,实用性强,紧扣《数据结构》课程主要内容,是作者讲授《数据结构》课程二十多年教学经验和体会的结晶。本书可满足高校在校学生的学习需要和各级各类考生考前复习的需要,也可供高校教师和其他专业技术人员参考。

版权所有,翻印必究。举报电话:010-62782989 13501256678 13801310933

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。

本书防伪标签采用特殊防伪技术,用户可通过在图案表面涂抹清水,图案消失,水干后图案复现;或将表面膜揭下,放在白纸上用彩笔涂抹,图案在白纸上再现的方法识别真伪。

图书在版编目(CIP)数据

数据结构学习辅导/宁正元,易金聪编著. —北京:清华大学出版社,2005.8

(高等院校信息技术课程学习辅导丛书)

ISBN 7-302-10981-8

I. 数… II. ①宁…②易… III. 数据结构—高等学校—教学参考资料 IV. TP311.12

中国版本图书馆 CIP 数据核字(2005)第 048943 号

出 版 者: 清华大学出版社

<http://www.tup.com.cn>

社 总 机: 010-62770175

地 址: 北京清华大学学研大厦

邮 编: 100084

客户服务: 010-62776969

组稿编辑: 张 龙

文稿编辑: 霍志国

印 刷 者: 北京国马印刷厂

装 订 者: 三河市新茂装订有限公司

发 行 者: 新华书店总店北京发行所

开 本: 185×260 印张: 17.75 字数: 417 千字

版 次: 2005 年 8 月第 1 版 2005 年 8 月第 1 次印刷

书 号: ISBN 7-302-10981-8/TP·7280

印 数: 1~3000

定 价: 24.00 元

前言

数据结构是计算机科学与技术类各专业的必修课程,既是专业基础课程,也是专业核心课程。这不仅因为这一学科的许多课程大都用到数据结构的有关知识,更重要的是从事这一领域的研发者必须具有良好的复杂程序设计能力,而程序数据的结构、组织、管理和应用水平是复杂程序设计能力的最基本的素质要求。数据结构课程的教学目标就是使学生学会分析研究计算机加工的数据对象的特性,能够在应用研发实践中适当地选择数据结构和相应的有关算法,初步掌握算法的时间与空间性能分析技巧,得到复杂程序设计的训练。

近年来,数据结构课程不仅在计算机科学与技术类各专业开设,信息类的许多本科专业和其他相关专业也开设了数据结构课程。计算机科学与技术学科的计算机应用技术、计算机软件与理论、计算机系统结构、软件工程等专业的硕士生和博士生入学考试也都基本上把数据结构课程列为必考的专业基础课程之一,由此可见大学很多专业对数据结构课程给予的显著地位和重视程度。

然而,数据结构课程的学习难度较大。除了课程自身的内容多、介绍的方法技术多、在先修课程中涉及的专业基础知识少之外,数据结构课程有着它自身的一些特点和规律:第一,学生不易理解和接受数据的逻辑结构的抽象数据类型表示;第二,动态存储结构的动态性和递归技术的抽象性,使得学生对相应的知识不易掌握;第三,算法描述的形式化和程序设计语言化使许多学生望而生畏;第四,算法设计的灵活多样性以及算法分析等内容使得许多学生较难掌握。所有这些,都是学生感觉到这门课程难度大的原因。许多学生在课堂上一听就明白,拿起课本认真一看也能基本弄懂,但做作业,尤其是做算法设计题目时总觉得无从下手。究其原因,主要是听懂数据结构的内容和应用数据结构知识解决实际问题之间存在着相当长的距离;理解掌握算法分析和设计的各种方法、技术与灵活运用这些方法解决各种具体问题之间也存在着较远距离。逐步缩短进而消除这些距离是解决数据结构课程学习难的关键所在,其根本途径在于加强实践环节,多学、多用、多做、多练,熟能生巧,以期达到对所学知识和方法技术的融会贯通。

在认真总结了数据结构课程教学实践的基础上,结合新世纪大学生的特点和学习需求,我们编写了这本《数据结构学习辅导》,以期通过该书中的学习要点、主要内容和知识点,给学习数据结构课程的学生归纳出关于课程内容的-一个主线条;通过典型习题的解析给学习数据结构课程的学生一些解题示范或启发。加强课程学习的实践环节,促进课程

内容中各知识点与方法和技术的融会贯通,为整个专业的学习打下坚实的基础。

本书的内容是根据《数据结构》课程的教学大纲确定的。全书共分 11 章,包括绪论、线性表、栈和队列、串、数组与广义表、树、图、检索、排序、文件和模拟试题。在前 10 章中,先对本章内容的学习要点、主要内容及知识点进行了概括总结,归纳出本章知识主线条,有利于学生复习总结和理解掌握;基础知识类题目是针对本章基本内容给出的复习和巩固各知识点的题目,分为填空题、选择题和应用题 3 种题型;算法设计题都属于应用题的范畴,侧重于各种基本方法与技术的运用和算法分析与设计能力的训练;每章都给出了几道上机实验题目,通过实验使学生了解并学会如何运用数据结构知识去解决现实世界的实际问题,具备较复杂程序的初步设计能力。第 11 章给出了 8 套模拟练习题,主要是为了帮助学生在学完课程基本内容后可以自我检验学习效果;每套题目都有选择、填空、判断、应用和设计等各种题型,覆盖课程的主要知识点;题目的难度等级有易、较易、中、较难和难 5 种,可以适应不同读者考前复习和模拟练习。

本书的读者对象是学习数据结构课程的本、专科学生,尤其是有志报考信息类各专业研究生的考生和参加软件水平与资格考试以及全国计算机等级考试三、四级考试的考生,也可作为高校教师和其他专业技术人员的参考用书。组织这本辅助教材的主要目的是为了帮助在校同学学好数据结构这门课程,所以在使用过程中要注意以下 4 点:第一,与课程学习内容同步使用。这样有利于教材中知识点的理解和掌握,有利于巩固和提高课堂教学效果。第二,切忌照抄照搬。算法的设计具有不惟一性,对算法设计类题目,书中给出了一种或多种答案,要在学习、理解、领会的基础上自己动手设计算法并编写程序,这样才能获得好的效果。第三,遵循循序渐进的原则。书中内容按知识结构分类组织,同类内容按典型到一般、由易到难的次序排列,读者最好能按次序阅读学习。但由于算法的讲解是独立构件,也可以根据需要有选择地阅读。第四,学会举一反三,触类旁通。课程内容中的知识点是有限的,但运用所学知识和方法技术解决实际问题则是无限的。重在掌握基本原理、基本方法和基本技术,并学以致用和灵活运用。考虑到考研和软件水平考试等考生的需要,在本书的内容组织上选择了一些难度较大的练习题;考虑到各级各类学校在教材使用中的不一致性,本书内容以知识模块结构组织而不针对某一具体教材,可以与几乎所有的数据结构教材配套使用。

本书是按清华大学出版社 2003 年巢湖计算机教学研讨会的讨论意见编写的。其中,第 1,5,8,9 章和第 11 章由宁正元教授编写,第 2,3,4,6,7 章和第 10 章由易金聪副教授编写。全书由宁正元教授策划、统稿。在本书组稿期间,福州大学教授陈国龙博士提供了许多参考资料;西北大学教授康宝生博士提出过许多宝贵的意见和建议;福建农林大学计算机系李金铭主任和许多硕士生给予了许多支持帮助;清华大学出版社给予了许多支持和鼓励,在此一并致以深深的谢意。

由于时间仓促和作者水平所限,书中难免存在疏漏和错误,敬请业内同仁和广大读者不吝赐教。

作 者

2005 年 5 月

CONTENTS

目 录

第 1 章 绪论	1
1.1 学习要点	1
1.2 主要内容及知识点	1
1.3 基础知识题	2
1.3.1 填空题	2
1.3.2 选择题	3
1.3.3 应用题	4
第 2 章 线性表	10
2.1 学习要点	10
2.2 主要内容及知识点	10
2.3 基础知识题	11
2.3.1 填空题	11
2.3.2 选择题	12
2.3.3 应用题	13
2.4 算法设计题	15
2.5 实验题	33
第 3 章 栈和队列	43
3.1 学习要点	43
3.2 主要内容及知识点	43
3.3 基础知识题	45
3.3.1 填空题	45
3.3.2 选择题	46
3.3.3 应用题	47
3.4 算法设计题	48
3.5 实验题	60

第4章 串	66
4.1 学习要点	66
4.2 主要内容及知识点	66
4.3 基础知识题	67
4.3.1 填空题	67
4.3.2 应用题	67
4.4 算法设计题	68
4.5 实验题	76
第5章 数组与广义表	80
5.1 学习要点	80
5.2 主要内容及知识点	80
5.3 基础知识题	81
5.3.1 填空题	81
5.3.2 应用题	82
5.4 算法设计题	83
5.5 实验题	102
第6章 树	108
6.1 学习要点	108
6.2 主要内容及知识点	108
6.3 基础知识题	109
6.3.1 填空题	109
6.3.2 选择题	111
6.3.3 应用题	112
6.4 算法设计题	119
6.5 实验题	136
第7章 图	141
7.1 学习要点	141
7.2 主要内容及知识点	141
7.3 基础知识题	143
7.3.1 填空题	143
7.3.2 选择题	144
7.3.3 应用题	146
7.4 算法设计题	150
7.5 实验题	162

第 8 章 检索	168
8.1 学习要点	168
8.2 主要内容及知识点	168
8.3 基础知识题	169
8.3.1 填空题	169
8.3.2 选择题	170
8.3.3 应用题	172
8.4 算法设计题	178
8.5 实验题	191
第 9 章 排序	197
9.1 学习要点	197
9.2 主要内容及知识点	197
9.3 基础知识题	198
9.3.1 填空题	198
9.3.2 选择题	200
9.3.3 应用题	202
9.4 算法设计题	206
9.5 实验题	221
第 10 章 文件	226
10.1 学习要点	226
10.2 主要内容及知识点	226
10.3 基础知识题	227
10.3.1 选择题	227
10.3.2 判断正误题	228
10.3.3 应用题	229
10.4 算法设计题	229
10.5 实验题	233
第 11 章 模拟试题	237
模拟试题一	237
模拟试题二	239
模拟试题三	242
模拟试题四	245
模拟试题五	247
模拟试题六	250

模拟试题七.....	252
模拟试题八.....	255
模拟试题部分参考答案.....	257
参考文献.....	272

第 1 章 绪 论

1.1 学习要点

- (1) 领会数据、数据类型等基本定义和基本概念。
- (2) 理解数据之间的关系和数据结构的概念及内容。对于数据的逻辑结构和存储结构之间的关系,必须分清哪些是逻辑结构的性质,哪些是存储结构的性质。
- (3) 理解抽象数据类型的定义、表示和实现方法,能结合实际问题的列举抽象数据类型的各种应用,并能自定义满足实际需要的抽象数据类型。
- (4) 理解算法的定义、算法特征与算法的描述方法。
- (5) 能用 C 语言描述数据结构、数据类型、抽象数据类型及简单算法。掌握计算语句频度和估算算法时间复杂度的方法,对简单算法能通过比较算法的时间与空间复杂性评价性能的优劣。

1.2 主要内容及知识点

1. 数据结构的基本概念和术语

本节内容包括数据、数据元素、数据对象、存储结构和数据类型等概念术语的含义。数据结构指数据及数据之间的相互关系。一般的,数据结构包括以下 3 方面的内容:数据的逻辑结构、数据的存储结构(又称数据的物理结构)和数据的运算及实现。数据的逻辑结构通常有集合、线性结构、树型结构和图状或网状结构。数据的存储结构表示数据元素及其关系如何存放在计算机内存的问题,有顺序存储方法、链状存储方法、索引存储方法和散列存储方法 4 种基本存储方法。数据的运算指定义在数据的逻辑结构上的一组操作的集合。每种逻辑结构都允许有某些运算,这些运算实际上是在逻辑结构上对抽象数据所施加的一系列“抽象”的操作。

2. 数据类型和抽象数据类型

数据类型是具有相同性质的计算机数据的集合及在这个数据上的一组运算,是与数据结构密切相关的概念。抽象数据类型(ADT)是指一个数据模型及其定义在该数据模型上的一组操作。抽象数据模型的定义仅取决于它的一组逻辑特性,与其在计算机内部如何表示和实现无关。抽象数据类型是算法的一个数据模型连同定义在该模型上的作为

该算法构件的一组运算。

3. 算法描述与算法评价

算法是对特定问题求解方法和步骤的一种描述。算法必须具备 5 个重要特性:输入、输出、有穷性、确定性和可行性。算法可以用自然语言、数学语言或约定的符号来描述,也可用计算机高级程序语言来描述。设计一个好的算法可以从正确性、可读性、健壮性和高效率几个方面考虑。评价算法的优劣程度可从时间效率、空间效率和简单性进行评价。算法的时间效率一般可以通过分析程序的语句频度和时间复杂性来进行。算法的空间效率可用空间复杂度来度量。

1.3 基础知识题

1.3.1 填空题

1. 数据结构即数据的逻辑结构,包括 ①、②、③ 和 ④ 4 种类型。树型结构和图型结构合称为 ⑤。数据的存储结构即物理结构,包括 ⑥、⑦、⑧ 和 ⑨ 4 种基本类型。

2. 数据结构是研究数据的 ① 和 ②,以及它们之间的相互关系,并对这种结构定义相应的 ③,设计相应的 ④,而确保经过这些运算后得到的新结构是 ⑤ 结构类型。

3. 一个数据结构用二元组表示时,它包括 ① 集合 K 和 K 上 ② 的集合 R 。

4. 一个算法应具有 ①、②、③、④ 和 ⑤ 5 个特性。

5. 一个算法的时间复杂性是该算法包含的 ① 的多少,它是一个算法运行时间的 ②。一个算法的空间复杂性是指该算法在运行过程中临时占用的 ③ 的大小。

6. 一个算法的时间复杂性通常用它相对于问题的规模的 ① 形式表示。当一个算法的时间复杂性与问题的规模 n 大小无关时,则表示为 ②;成正比时,则表示为 ③;成对数关系时,则表示为 ④;成平方时,则表示为 ⑤。

7. ① 是描述客观事物的数、字符,以及所有能输入到计算机且被计算机程序加工处理的符号集合。② 是数据的基本单位,有时一个 ② 由若干个 ③ 组成,在这种情况下,称 ② 为记录,③ 是数据的最小单位,而由记录组成的线性表为 ④。被计算机加工的 ② 不是孤立无关的,它们彼此之间一般存在某种联系,通常将 ② 间的这种联系称为 ⑤。算法的计算量的大小称为计算的 ⑥。

解:

1. ① 线性结构 ② 树型结构 ③ 图型结构 ④ 集合 ⑤ 非线性结构
⑥ 顺序 ⑦ 链接 ⑧ 索引 ⑨ 散列

2. ① 物理结构 ② 逻辑结构 ③ 运算 ④ 算法 ⑤ 原来的

3. ① 数据元素 ② 二元关系

4. ① 有穷性 ② 确定性 ③ 可行性 ④ 0 或多个输入 ⑤ 1 或多个输出

5. ① 简单操作次数 ② 相对量度 ③ 存储空间
 6. ① 数量级 ② $O(1)$ ③ $O(n)$ ④ $O(\log_2 n)$ ⑤ $O(n^2)$
 7. ① 数据 ② 数据元素 ③ 数据项 ④ 文件 ⑤ 结构 ⑥ 复杂性

1.3.2 选择题

1. 下面程序的时间复杂性为_____。

```
for(i=0; i<m; i++)
  for(j=0; j<n; j++)
    A[i][j]=i*j;
```

- (A) $O(m^2)$ (B) $O(n^2)$ (C) $O(m \times n)$ (D) $O(m+n)$

2. 执行下面程序段, 语句 3 的执行次数为_____。

```
for(i=0; i<n-1; i++)
  for(j=n; j>i; j--)
    state;
```

- (A) $n(n+2)/2$ (B) $(n-1)(n+2)/2$ (C) $n(n+1)/2$ (D) $(n-1)(n+2)$

3. 下列程序的时间复杂性为_____。

```
i=0; s=0;
while(s<n)
{ i++;
  s=s+i;
}
```

- (A) $O(\sqrt{n})$ (B) $O(1)$ (C) $O(n)$ (D) $O(n^2)$

4. 下列程序的时间复杂性为_____。

```
for(i=0; i<m; i++)
  for(j=0; j<t; j++)
    c[i][j]=0;
for(i=0; i<m; i++)
  for(j=0; j<t; j++)
    for(k=0; k<n; k++)
      c[i][j]=c[i][j]+a[i][k]*b[k][j];
```

- (A) $O(m \times n \times t)$ (B) $O(m+n+t)$ (C) $O(m+n \times t)$ (D) $O(m \times t+n)$

5. 下列程序段的时间复杂性_____。

```
i=1; k=0; n=100;
do{k=k+10*i;
  i=i++;
}while(i!=n);
```

- (A) $O(1)$ (B) $O(n)$ (C) $O(i)$ (D) $O(i \times n)$

6. 下列程序段的时间复杂性为_____。

```
x=n;          /* n>1 */
y=0;
while((x>=(y+1)*(y+1))
       y=y+1;
```

- (A) $O(n)$ (B) $O(\sqrt{n})$ (C) $O(1)$ (D) $O(n^2)$

7. 算法指的是_____。

- (A) 计算机程序 (B) 解决问题的计算方法
(C) 排序算法 (D) 解决问题的有限运算序列

8. 某程序的时间复杂度为 $(3n + n\log_2 n + n^2 + 8)$, 其数量级表示为_____。

- (A) $O(n)$ (B) $O(n \log_2 n)$ (C) $O(n^2)$ (D) $O(\log_2 n)$

9. 若需要利用形参直接访问实参,则应把形参变量说明为_____参数。

- (A) 指针 (B) 引用 (C) void (D) 值

解：

1. (C) 2. (B) 3. (A) 4. (A) 5. (A) 6. (B) 7. (D)
8. (C) 9. (B)

1.3.3 应用题

1. 选择解决某种问题的最佳数据结构的标准是什么?

解：一般有如下两条标准：

- (1) 所需的存储空间量;
- (2) 算法所需要的时间。

算法所需要的时间又包括以下几点:

- (1) 程序运行时所需要的数据总量;
- (2) 源程序进行编译所需要的时间;
- (3) 计算机执行每条指令所需的时间;
- (4) 程序中的指令重复执行的次数,而本条正是讨论算法中的重点内容。

2. 根据大 O 表示法的意义, 证明大 O 表示法的加法法则。

解：首先给出如下加法法则：

若两个程序段的时间复杂性为 $T_1(n)=O(f_1(n))$ 和 $T_2(n)=O(f_2(n))$, 那么这两个程序段依次执行时的时间复杂性如下:

$$T(n) = T_1(n) + T_2(n) = O(\max(f_1(n), f_2(n)))$$

上式可以证明如下。由于

$$T_1(n) = O(f_1(n)), \quad T_2(n) = O(f_2(n))$$

则存在常数 c 和 n_0 , 使得当 $n \geq n_0$ 时, 有

$$T_1(n) + T_2(n) \leq c \times \max(f_1(n), f_2(n))$$

例如,

$$T_1(n) = O(n^2), \quad T_2(n) = O(n^3)$$

当 $n_0=1$ 时,对所有 $n \geq n_0$ 都有

$$n^3 + n^2 \leq n^3 + n^3 = 2n^3$$

当取 $c=2$ 时,则有 $T(n) \leq c \times n^3$,取 $T(n)=O(n^3)$,所以

$$T(n) = T_1(n) + T_2(n) = O(\max(f_1(n), f_2(n))) \text{ 成立。}$$

3. 将数量级 $O(1), O(n), O(n^2), O(n^3), O(n \log_2 n), O(\log_2 n), O(2^n)$ 按增长率从小到大排列。

解: 上面的几种类型的数量级中, $O(1)$ 为常量型, $O(n)$ 为线性型, $O(n^2)$ 为平方型, $O(n^3)$ 为立方型, $O(\log_2 n)$ 为对数型, $O(2^n)$ 为指数型, $O(n \log_2 n)$ 为线性对数型, 它们按增长率从小到大的顺序为 $O(1), O(\log_2 n), O(n), O(n \log_2 n), O(n^2), O(n^3), O(2^n)$ 。

4. 猴子吃桃问题。猴子第 1 天摘下若干个桃子, 当即吃了一半, 还不过瘾, 又多吃了一个; 第 2 天早上又将剩下的桃子吃掉一半, 又多吃了一个; 以后每天早上都吃了前一天剩下的一半零一个, 到第 10 天早上再吃时, 见只剩下一个桃子了。求第 1 天共摘多少个桃子。

解:

```
main()
{ int day, x1, x2;
  day=9;
  x2=1;
  while(day>0)
  { x1=(x2+1)*2;
    x2=x1;
    day--;
  }
  printf("桃子总数=%d\n", x1);
} /* main */
```

结果: 桃子总数为 1534。

5. 求两个自然数, 其和是 667, 其最小公倍数和最大公约数之比是 $120:1$ 。

解: 设两个自然数分别为 m 和 $667-m$ ($2 \leq m \leq 333$), 然后只要求这两个自然数的最小公倍数与最大公约数, 判断是否满足条件即可。其程序如下:

```
main()
{ int gcd();
  int gbs();
  int m, n, k;
  for(m=2; m<=333; m++)
  { n=gbs(m, 667-m);
    k=gcd(m, 667-m);
    if ((n==120*k) && (n%k==0))
      printf("\n%3d %3d", m, 667-m);
  }
} /* main */
```

```

int gcd(a,b)
int a,b;
{ int i;
  for(i=a;i>=1;i--)
    if (! ((a%i)|| (b%i)))
      return(i);
} /* gcd */
int gbs(a,b)
int a,b;
{ int i;
  i=b;
  while((i%a!=0)) i+=b;
  return(i);
} /* lcm */

```

6. 哥德巴赫猜想的命题之一: 大于 6 的偶数由两个素数组成。编写程序, 验证 4~100 之间偶数由哪些素数组成。

解: 基本思想是输入一个偶数 t , 将 t 分成两部分 t_1 和 t_2 , 而且 t_1 初值为 1, 然后 t_1 增加 1, 直到 t_1 为素数为止; 此时检查 $t_2 = t - t_1$ 是否为素数, 若不是, 重复以上过程, 直到 t_2 也为素数为止, t 的值在 4~100 之间变化。其程序如下:

```

#include "math.h"
#define N 100
main()
{ int i=0,t,t1,t2;
  int prime(); /* 判断素数函数 */
  t=4;
  while(t<=N)
  { t1=1;
    do
    { do
      { if (t1<3) t1++;
        else
          t1=t1+2;
      } while(! prime(t1));
      t2=t-t1;
    } while(! prime(t2));
    printf("%3d=%3d + %3d",t,t1,t2);
    i++;
    if (i % 4 ==0) printf("\n");
    t=t+2;
  }
} /* main */
int prime(n) /* 判断素数函数 */

```

```

int n;
{ int flag=1,k,i;
  k=(int) (sqrt(n));
  for (i<=k;i=2;i++)
    if( n % i ==0)
      { flag=0;
        break;
      }
  if (i>=k+1) flag=1;
  else flag=0;
  return(flag);
}/* prime */

```

7. 分析下列程序段,求用大 O 记号表示执行时间为 n 的函数。

① $i=1; k=0;$
 while($i \leq n-1$)
 { $k=k*10*i;$
 $i++;$
 }

解: $T(n)=O(n)$ 。

② $i=1; j=0;$
 while($(i+j) \leq n$)
 if ($i > j$) $j++;$
 else $i++;$

解: $T(n)=O(n)$ 。

③ $x=91; y=100;$
 while($y > 0$)
 if($x > 0$)
 { $x=x-10;$
 $y=y-1;$
 }
 else
 $x=x+1;$

解: 本程序段是著名的 McCarthy 函数

$$M(x) = \begin{cases} x-10, & x > 100 \\ M(M(x+1)), & x \leq 100 \end{cases}$$

对任何的 $x \leq 100, M(x)=91$, 所以此程序段实质上是一个二重循环, 对每个 $y(y > 0)$ 值, if 语句执行 11 次, 其中 10 次是执行 $x++$ 语句, 但它们与 n 无关, 所以 $T(n)=O(1)$ 。

④ for($i=0; i < n; i++$)

```

for(j=0;j<i;j++)
    for(k=0;k<j;k++)
        x=x+1;

```

解: $T(n) = O(n^3)$ 。

```

⑤ i=1;
do
    {j=1;
    do
        {printf("%d\n", i * j);
        j++;
        }while(j>n);
    i++;
    }while(i>n);

```

解: $T(n) = n(n+1)/2 = O(n^2)$ 。

```

⑥ x=n;      /* n>1 */
y=0;
while((x>=(y+1)*(y+1))
    y=y+1;

```

解: $T(n) = O(\sqrt{n})$ 。

8. 按增长率由小到大的顺序排列下列各函数: 2^{100} , $(3/2)^n$, $(2/3)^n$, $(4/3)^n$, $(n)^n$, $(n)^{2/3}$, $(n)^{3/2}$, $\sqrt{n}$, $n!$, n , $\log_2 n$, $n/\log_2 n$, $\log_2^2 n$, $\log_2(\log_2 n)$, $n\log_2 n$, $n^{\log_2 n}$ 。

解: 各函数的排列次序如下, 即 $(2/3)^n$, 2^{100} , $\log_2(\log_2 n)$, $\log_2 n$, $\log_2^2 n$, $\sqrt{n}$, $(n)^{2/3}$, n , $n\log_2 n$, $(n)^{3/2}$, $n/\log_2 n$, $(4/3)^n$, $(3/2)^n$, $n^{\log_2 n}$, $n!$, $(n)^n$ 。

9. 背包问题。有不同价值、不同重量的物品 n 件, 求从这 n 件物品中选取一部分物品的选择方案, 使选中物品的总重量不超过指定的限制重量, 但选中物品的价值之和为最大。

解: 可以按照以下算法编写递归和非递归程序。

【算法】背包问题, 找最佳方案。

```

try(物品 i, 当前选择已达到的重量和 tw, 本方案可能达到的总价值 tv)
{ /* 考虑物品 i 包含在当前方案的可能性 */
    if(包含物品 i 是可接受的)
    { 将物品 i 包含在当前方案中;
      if (i<n-1)
          try(i+1, tw+物品 i 的重量, tv);
      else /* 又一个完整方案, 因它比前面的方案好, 以它作为最佳方案 */
          以当前方案作为临时最佳方案保存;
      恢复物品 i 不包含状态;
    }
}

```