

王道考研系列

2013年 计算机考研统考真题 命题思路分析

2013NIAN
JISUANJI KAOYANTONGKAOZHENTI
MINGTISILU FENXI

● 王道论坛 组编

更多计算机
考研和学习交流
尽在 www.cskaoyan.com

王道论坛由清华、北大、浙大、上交、哈工大、中科院等名校计算机研究生共同创办，致力于给报考计算机专业研究生的考生提供帮助和指导。王道论坛是唯一一家专注于名校计算机考研的论坛。

王道论坛组编的一系列计算机考研辅导书，融入了论坛众多名校高分选手的智慧，以及论坛交流精华，我们希望将其打造成“书本+在线”的学习方式。对于书中的疑难点，欢迎大家在论坛交流和讨论。

目前已有越来越多的名校采用上机的形式考查考生的动手编程能力，为方便大家练习，我们搭建了九度OJ(ac.jobdu.com)，收录了全国各大高校的复试上机题，希望能给考生复习上机考是供强有力的支持。



中南大学出版社
www.csupress.com.cn

王道考研系列

2013 年计算机考研统考真题 命题思路分析

王道论坛 组 编



中南大学出版社
www.csupress.com.cn

内 容 简 介

本书根据最新的考试大纲,在剖析 2009—2012 年统考真题的基础上,对统考真题所涉及到的知识点进行详尽分析。全书共 105 个考点,并针对考点进行了扩展性分析以及对考点出题思路及解答的分析,以便考生更容易的吃透知识点、掌握和巩固知识。

本书可作为考生参加计算机专业研究生入学考试的备考复习书,也可作为计算机专业的学生学习和巩固相关知识的辅导用书。

图书在版编目(CIP)数据

2013 年计算机考研统考真题命题思路分析/王道论坛组编.

—长沙:中南大学出版社,2012. 10

ISBN 978-7-5487-0703-5

I. 2... II. 王... III. 电子计算机—研究生—入学考试—
自学参考资料 IV. TP3

中国版本图书馆 CIP 数据核字(2012)第 243502 号

2013 年计算机考研统考真题命题思路分析

王道论坛 组编

☐责任编辑 陈海波

☐责任印制 文桂武

☐出版发行 中南大学出版社

社址:长沙市麓山南路

邮编:410083

发行科电话:0731-88876770

传真:0731-88710482

☐印 装 长沙市华中印刷厂

☐开 本 787×1092 1/16 ☐印张 14.75 ☐字数 366 千字

☐版 次 2012 年 10 月第 1 版 ☐2012 年 10 月第 1 次印刷

☐书 号 ISBN 978-7-5487-0703-5

☐定 价 36.00 元

图书出现印装问题,请与经销商调换

前言

最后 50 天，拼的是心态与意志！

你的身边会陆陆续续有一些意志不太坚定的同学放弃自己的梦想，他们的言行或许多少会影响到你的心情与心态。但我相信，一路用心坚持到最后的人，一定会收获属于自己的那份回报。成功并不是遥不可及，坚定自己的梦想，做最好的自己！

对于任何一门考试，真题永远都是最宝贵的资源，研究生入学考试更是如此。真题不仅仅需要实战模拟，更需要方方面面地吃透、再吃透。而由真题引申出的周边核心考点，也一定要牢固掌握，以前考过的知识点极有可能再次涉及，重要的考点往往容易反复出现。《2013 年计算机考研统考真题分类及命题思路分析》就是一本帮助你实现这样目的的一本书！

我们在认真、细致剖析 2009 - 2012 年统考真题的基础上，对统考真题所涉及到的知识点进行详尽分析。并总结统考的命题规律，针对命题规律命制了高质量的模拟题（即《王道 2013 年最后 8 套模拟题》），这样的规律可以指明考生在最后阶段的复习方向。也正是基于我们对命题规律的把握，去年王道论坛命制的《王道 2012 年最后 6 套模拟题》，不仅原题命中算法设计题，更是点中多道选择题和综合题的核心考点和相关思路。

在考研前的最后 30 天，同学们一定要抓住专业课复习的四大重点：①回顾以前复习中标注的重点和难点，毕竟知识都是容易遗忘的；②查漏补缺，特别是之前做错的题目，这是为了以后遇到类似的题不再出错；③仔细分析历年真题，真题涉及到的知识点一定是核心考点；④做几套高质量的模拟题。

我们都是 CS 的“屌丝”，唯有自强不息，才能实现“屌丝”的梦想！

祝福每一个有梦的“屌丝”！

王道论坛

目 录

第一篇 数据结构

第一章 绪论	(3)
真题分布及命题重点	(3)
考点1: 时间复杂度与空间复杂度	(3)
第二章 线性表	(6)
真题分布及命题重点	(6)
考点2: 线性表的顺序表示	(6)
考点3: 线性表的链式表示	(8)
第三章 栈和队列	(14)
真题分布及命题重点	(14)
考点4: 栈和队列的基本性质	(14)
考点5: 栈和队列的存储结构	(15)
考点6: 双端队列	(16)
考点7: 栈与队列的应用	(18)
考点8: 特殊矩阵的压缩存储	(21)
第四章 树与二叉树	(24)
真题分布及命题重点	(24)
考点9: 树的基本性质	(24)
考点10: 二叉树的定义与性质	(25)
考点11: 二叉树的遍历	(26)
考点12: 二叉排序树	(28)
考点13: 平衡二叉树	(31)
考点14: 树、森林与二叉树的转换	(34)
考点15: 线索二叉树的基本概念和构造	(36)
考点16: 哈夫曼树与哈夫曼编码	(38)
第五章 图	(40)
真题分布及命题重点	(40)
考点17: 图的基本概念	(40)
考点18: 图的存储及基本操作	(42)

考点 19: 图的遍历	(44)
考点 20: 最小(代价)生成树	(45)
考点 21: 最短路径	(46)
考点 22: 拓扑排序	(50)
考点 23: 关键路径	(51)
第六章 查找	(54)
真题分布及命题重点	(54)
考点 24: 顺序查找法和折半查找法	(54)
考点 25: B 树及其基本操作、B+ 树的基本概念	(57)
考点 26: 散列(Hash)表	(60)
第七章 排序	(63)
真题分布及命题重点	(63)
考点 27: 插入排序	(63)
考点 28: 交换排序	(65)
考点 29: 选择排序	(68)
考点 30: 二路归并排序	(70)
考点 31: 各种内部排序方法的比较	(72)

第二篇 计算机组成原理

第一章 计算机系统概论	(77)
真题分布及命题重点	(77)
考点 32: 计算机系统层次结构	(77)
考点 33: 计算机的性能指标	(78)
第二章 数据的表示和运算	(80)
真题分布及命题重点	(80)
考点 34: 数据的存储方式	(80)
考点 35: 定点数的表示与运算	(82)
考点 36: IEEE754 标准	(88)
考点 37: 各种精度数据的转换	(89)
考点 38: 浮点数运算	(91)
第三章 存储系统的层次结构	(93)
真题分布及命题重点	(93)
考点 39: 半导体随机存取存储器	(93)
考点 40: 主存储器与 CPU 的连接	(95)
考点 41: 低位交叉存储器	(96)
考点 42: 高速缓冲存储器(Cache)	(97)
考点 43: 虚拟存储器	(101)

第四章 指令系统	(105)
真题分布及命题重点	(105)
考点 44: 指令格式	(105)
考点 45: 指令的寻址方式	(107)
考点 46: CISC 与 RISC	(109)
第五章 中央处理器	(111)
真题分布及命题重点	(111)
考点 47: CPU 的功能和基本结构	(111)
考点 48: 指令执行过程	(112)
考点 49: 数据通路的功能和基本结构	(114)
考点 50: 控制器的功能和工作原理	(116)
考点 51: 指令流水线	(118)
第六章 总线	(123)
真题分布及命题重点	(123)
考点 52: 总线分类	(123)
考点 53: 总线的性能指标	(124)
考点 54: 总线标准	(125)
第七章 输入输出系统	(127)
真题分布及命题重点	(127)
考点 55: 外部设备	(127)
考点 56: I/O 接口	(128)
考点 57: 程序查询方式	(129)
考点 58: 程序中断方式	(129)
考点 59: DMA 方式	(133)

第三篇 操作系统

第一章 操作系统概述	(137)
真题分布及命题重点	(137)
考点 60: 操作系统的概念、特征和功能	(137)
考点 61: 内核态与用户态	(138)
考点 62: 中断、异常	(139)
考点 63: 系统调用	(139)
第二章 进程管理	(141)
真题分布及命题重点	(141)
考点 64: 进程与线程	(141)
考点 65: 进程状态与进程控制	(143)
考点 66: 处理机调度	(144)

考点 67: 进程同步与互斥	(147)
考点 68: 经典同步问题	(150)
考点 69: 死锁	(153)
第三章 内存管理	(156)
真题分布及命题重点	(156)
考点 70: 内存管理的概念	(156)
考点 71: 连续分配管理方式	(157)
考点 72: 非连续分配管理方式	(159)
考点 73: 虚拟页式存储管理	(161)
考点 74: 抖动	(168)
第四章 文件管理	(170)
真题分布及命题重点	(170)
考点 75: 目录结构	(170)
考点 76: 文件共享和文件保护	(171)
考点 77: 文件的操作	(172)
考点 78: 文件实现	(173)
考点 79: 磁盘组织与管理	(176)
第五章 设备管理	(179)
真题分布及命题重点	(179)
考点 80: I/O 控制方式	(179)
考点 81: I/O 软件的层次结构	(179)
考点 82: I/O 调度与缓冲区	(181)
考点 83: 设备分配与回收	(182)

第四篇 计算机网络

第一章 计算机网络体系结构	(185)
真题分布及命题重点	(185)
考点 84: 计算机网络的分层结构	(185)
考点 85: ISO/OSI 参考模型和 TCP/IP 模型	(186)
第二章 物理层	(188)
真题分布及命题重点	(188)
考点 86: 奈奎斯特定理和香农定理	(188)
考点 87: 电路交换、报文交换与分组交换	(189)
考点 88: 物理层接口和物理层设备	(191)
第三章 数据链路层	(192)
真题分布及命题重点	(192)
考点 89: 流量控制、可靠传输与滑动窗口机制	(192)

考点 90: 随机访问介质访问控制	(194)
考点 91: 局域网	(197)
考点 92: 局域网交换机及其工作原理	(198)
第四章 网络层	(200)
真题分布及命题重点	(200)
考点 93: IP 数据报	(200)
考点 94: 子网划分与子网掩码、CIDR	(203)
考点 95: ARP 协议与 ICMP 协议	(209)
考点 96: 路由协议	(210)
考点 97: 网络层设备	(211)
第五章 传输层	(213)
真题分布及命题重点	(213)
考点 98: TCP 段的分析	(213)
考点 99: TCP 连接管理	(216)
考点 100: TCP 可靠传输	(218)
考点 101: TCP 流量控制与拥塞控制	(218)
第六章 应用层	(221)
真题分析及命题重点	(221)
考点 102: 域名系统 DNS	(221)
考点 103: 文件传送协议 FTP	(222)
考点 104: 电子邮件 EMAIL	(223)
考点 105: 万维网 WWW	(223)

第一篇 数据结构

- 第一章 绪 论
- 第二章 线性表
- 第三章 栈和队列
- 第四章 树与二叉树
- 第五章 图
- 第六章 查找
- 第七章 排序

第一章 绪论

真题分布及命题重点

年份	分值	单选题	综合题
2009	0 + 0	—	—
2010	0 + $\sqrt{}$ *	—	分析算法的时空复杂度
2011	2 + $\sqrt{}$	分析程度段的时间复杂度	分析算法的时空复杂度
2012	2 + $\sqrt{}$	分析递归程序的时间复杂度	分析算法的时间复杂度

命题重点：

1. 根据给定代码段的特点，分析算法的时间复杂度。
2. 时空复杂度分析是每年必考考点，或单独命题或结合算法题，务必重点掌握。

考点1：时间复杂度与空间复杂度

1. 【2011年,1】设 n 是描述问题规模的非负整数，下面程序片段的时间复杂度是_____。

```
x = 2;
while (x < n/2)
    x = 2 * x;
```

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

【答案】A。

【解析】在程序中，执行频率最高的语句为“ $x = 2 * x$ ”。设该语句共执行了 $T(n)$ 次，则 $2^{T(n)+1} \leq n/2$ ，故 $T(n) = \log_2(n/2) - 1 = \log_2 n - 2$ ，得 $T(n) = O(\log_2 n)$ 。

【考点笔记】时间复杂度

时间复杂度 $T(n)$ 是指算法中所有语句的频度(执行次数)之和。人们关心的是当 n 趋于无穷时 $T(n)$ 的数量级，而非 $T(n)$ 的准确大小，因此以 $T(n)$ 的数量级来表征时间复杂度。

例如 $T(n) = n^3 + n^2 + n$ ，可认为时间复杂度 $T(n) = O(n^3)$ ，这里数量级最大的一项必定是由最深层循环的语句贡献的，称之为基本运算。由于 $T(n)$ 与算法中基本运算的频度 $f(n)$ 同数量级，所以通常采用基本运算的频度的数量级 $O(f(n))$ 来分析算法的时间复杂度，

* 注：“+”前为单选题的总分值，每个单选题2分；“+”后为综合题的总分值，“ $\sqrt{}$ ”表示有综合题部分涉及到本章知识点。

记为 $T(n) = O(f(n))$ 。

最终归结为一句话：将算法中基本运算的执行次数的数量级作为时间复杂度。

时间复杂度的计算遵循两种规则：

加法法则： $T(n) = T_1(n) + T_2(n) = O(f(n)) + O(g(n)) = O(\max(f(n), g(n)))$

乘法法则： $T(n) = T_1(n) \times T_2(n) = O(f(n)) \times O(g(n)) = O(f(n) \times g(n))$

如，设 $a\{\}$ 、 $b\{\}$ 、 $c\{\}$ 三个语句块的时间复杂度分别为 $O(1)$ 、 $O(n)$ 、 $O(n^2)$ ，则

1) $a\{$

$b\{\}$

$c\{\}$

$\}$ 的时间复杂度为 $O(n^2)$ ，

满足加法法则；

2) $a\{$

$b\{$

$c\{\}$

$\}$

$\}$ 的时间复杂度为 $O(n^3)$ ，满足乘法法则。

注意：在有些情况下，算法中基本运算的执行次数还随问题的输入数据集不同而不同。

2. [2012 年, 1] 求整数 $n(n \geq 0)$ 阶乘的算法如下，其时间复杂度是_____。

```
int fact(int n){
    if(n <= 1) return 1;
    return n * fact(n - 1);
}
```

A. $O(\log_2 n)$

B. $O(n)$

C. $O(n \log_2 n)$

D. $O(n^2)$

【答案】B。

【解析】本算法是一个递归运算，即算法中出现了调用自身的情形。递归的边界条件是 $n \leq 1$ ，每调用一次 $\text{fact}()$ ，传入该层 $\text{fact}()$ 的参数值减 1。采用递归式来表示时间复杂度有

$$T(n) = \begin{cases} O(1) & n \leq 1 \\ T(n-1) + 1 & n > 1 \end{cases}$$

则 $T(n) = T(n-1) + 1 = T(n-2) + 2 = \cdots = T(1) + n - 1 = O(n)$ ，故时间复杂度为 $O(n)$ 。

【考点拓展】递归算法

递归是算法领域的一个重要思想。在后续的二叉树中也有递归的应用。递归算法可以通过迭代或栈改写为非递归算法。以本题为例，也可写成以下两种形式：

迭代方式	栈方式
<pre>1) int fact(int n){ int i,result=1; for(i=n;i>1;i--) result*=i; return result; }</pre>	<pre>2) int fact(int n){ Stack S; int top,result=1; while(n>1) Push(&S,n--); while(Pop(&S,&top)) result*=top; return result; }</pre>

递归算法的特性:

(1) 一个算法直接或间接调用自身。

(2) 必须有一个明确的递归结束条件, 称为递归出口。

由于递归算法需要反复进行函数调用与返回, 运行效率较低。实现递归的关键是建立递归调用工作栈。当有多个函数构成嵌套调用时, 按照后调用先返回的原则, 因此递归调用需要开辟栈以存放每一层的返回点、局部变量等, 栈的大小也就是递归深度和递归算法空间复杂度的大小。

【考点笔记】空间复杂度

空间复杂度 $S(n)$ 指算法运行过程中所使用的辅助空间的大小, 通常结合算法题考查。

若输入数据所占空间只取决于问题本身, 和算法无关, 则只需分析除输入数据和程序之外的临时分配的额外空间。算法原地工作是指算法所需辅助空间是常量, 即 $O(1)$ 。

第二章 线性表

真题分布及命题重点

年份	分值	单选题	综合题
2009	0 + 15	—	单链表的遍历操作
2010	0 + 13	—	顺序表的循环移动
2011	0 + 15	—	顺序表的查找/遍历操作
2012	0 + 13	—	单链表的查找/遍历操作

命题重点:

1. 顺序表的性质、各种操作(查找、遍历、移动、交换、删除等)及灵活应用。
2. 单链表的性质、各种操作(查找、遍历、移动、交换、删除等)及灵活应用。
3. 本章是算法题命题的基础,近4年的算法题都是基于线性表的。
4. 考研不会直接考查线性表操作的代码,而往往是带有一定的技巧性。算法题喜欢挑选微软、谷歌等公司的面试题,读者在学习时应注意思维的培养。这也是为什么很多读者熟练记住了线性表的各种操作代码,但遇到此类题时却不知从何下手的缘故。
5. 算法题考查的重点是思想,在写代码时不应过于拘泥于每个细节。

考点2: 线性表的顺序表示

1. 【2010年,42】 设将 $n(n > 1)$ 个整数存放到一维数组 R 中。试设计一个在时间和空间两方面都尽可能高效的算法。将 R 中保存的序列循环左移 $p(0 < p < n)$ 个位置,即将 R 中的数据由 $(X_0, X_1, \dots, X_{n-1})$ 变换为 $(X_p, X_{p+1}, \dots, X_{n-1}, X_0, X_1, \dots, X_{p-1})$ 。要求:

- (1) 给出算法的基本设计思想。
- (2) 根据设计思想,采用 C 或 C++ 或 Java 语言描述算法,关键之处给出注释。
- (3) 说明你所设计算法的时间复杂度和空间复杂度。

【解析】 考查顺序结构线性表的元素的移动操作,将数组中的序列循环左移。

(1) 算法的基本设计思想:

可以将这个问题看作是 把数组 ab 转换成数组 ba (a 代表数组的前 p 个元素, b 代表数组中余下的 $n-p$ 个元素), 先将 a 逆置得到 $a^{-1}b$, 再将 b 逆置得到 $a^{-1}b^{-1}$, 最后将整个 $a^{-1}b^{-1}$ 逆置得到 $(a^{-1}b^{-1})^{-1} = ba$ 。设 `Reverse` 函数执行将数组元素逆置的操作, 对 `abcdefgh` 向左循环移动 3 ($p=3$) 个位置的过程如下:

`Reverse(0, p-1)` 得到 `cbadefgh`;

Reverse(p, n-1)得到 cbahgfed;

Reverse(0, n-1)得到 defghabc。

注: Reverse 中, 两个参数分别表示数组中待转换元素的始末位置。

(2)使用 C 语言描述算法如下:

```
void Reverse(int R[], int from, int to) {
    int i, temp;
    for(i=0; i < (to-from+1)/2; i++)
        { temp=R[from+i]; R[from+i]=R[to-i]; R[to-i]=temp; }
} //Reverse

void Converse(int R[], int n, int p){
    Reverse(R, 0, p-1);
    Reverse(R, p, n-1);
    Reverse(R, 0, n-1);
}
```

(3)上述算法中 3 个 Reverse 函数的时间复杂度分别为 $O(p/2)$ 、 $O((n-p)/2)$ 和 $O(n/2)$, 故所设计的算法的时间复杂度为 $O(n)$, 空间复杂度为 $O(1)$ 。

【另解】借助辅助数组来实现。

算法思想: 创建大小为 p 的辅助数组 S , 将 R 中前 p 个整数依次暂存在 S 中, 同时将 R 中后 $n-p$ 个整数左移, 然后将 S 中暂存的 p 个数依次放回到 R 中的后续单元。

算法的时间复杂度为 $O(n)$, 空间复杂度为 $O(p)$ 。

【考点笔记】顺序表的定义

顺序表是指用一组连续的存储单元, 依次存储线性表中的各个元素, 从而使得逻辑上相邻的元素在物理位置上也相邻, 因此可以随机存取(根据首元素地址和元素序号)表中任一元素。

顺序表的缺点也很明显, 如元素的插入和删除需要移动大量的元素, 插入操作平均需要移动 $n/2$ 个元素, 删除操作平均需要移动 $(n-1)/2$ 个元素, 而且存储分配需要一段连续的存储空间, 不够灵活。

【考点笔记】顺序表的插入操作

对于插入算法, 若表长为 n , 则在第 i 位置插入元素, 则从 a_n 到 a_i 都要向后移动一个位置, 共需移动 $n-i+1$ 个元素, 平均时间复杂度为 $O(n)$ 。代码片段如下:

```
//判断 i 的范围是否有效, 否则非法
//判断当前存储空间是否已满, 否则不能插入
for(int j=L.length; j>=i; j--) //将第 i 个位置及之后的元素后移
    L.data[j]=L.data[j-1];
L.data[i-1]=e; //在位置 i 处放入 e, 数组从 0 开始存储
L.length++; //线性表长度加 1
```

【考点笔记】顺序表的删除操作

对于删除算法, 若表长为 n , 当删除第 i 个元素时, 从 a_{i+1} 到 a_n 都要向前移动一个位置, 共需移动 $n-i$ 个元素, 平均时间复杂度为 $O(n)$ 。代码片段如下:

```

//判断 i 的范围是否有效
for(int j=i; j<L.length; j++)          //将第 i 个位置之后的元素前移
    L.data[j-1]=L.data[j];
L.length--;                             //线性表长度减 1

```

【考点笔记】顺序表的查找

(1)按序号查找,顺序表具有随机存取(根据首元地址和序号)的特点,时间复杂度为 $O(1)$ 。

(2)按值 x 查找,主要运算是比较操作,比较的次数与值 x 在表中的位置有关,也与表长有关,平均比较次数为 $(n+1)/2$,时间复杂度为 $O(n)$ 。

考点 3: 线性表的链式表示

2. [2009 年, 42] 已知一个带有表头结点的单链表, 结点结构为:

data	link
------	------

。假设该链表只给出了头指针 list。在不改变链表的前提下, 请设计一个尽可能高效的算法, 查找链表中倒数第 k 个位置上的结点(k 为正整数)。若查找成功, 算法输出该结点的 data 域的值, 并返回 1; 否则, 只返回 0。要求:

- (1)描述算法的基本设计思想。
- (2)描述算法的详细实现步骤。
- (3)根据设计思想和实现步骤, 采用程序设计语言描述算法(使用 C、C++ 或 Java 语言实现), 关键之处请给出简要注释。

【解析】考查链表的查找操作, 查找链表中倒数第 k 个结点。

(1)算法的基本设计思想:

问题的关键是设计一个尽可能高效的算法, 通过链表的一趟遍历, 找到倒数第 k 个结点的位置。算法的基本设计思想: 定义两个指针变量 p 和 q , 初始时均指向头结点的下一个结点(链表的第一个结点)。 p 指针沿链表移动, 当 p 指针移动到第 k 个结点时, q 指针开始与 p 指针同步移动; 当 p 指针移动到最后一个结点时, q 指针所指示结点为倒数第 k 个结点。以上过程对链表仅进行一遍扫描。

(2)算法的详细实现步骤:

- ①count = 0, p 和 q 指向链表表头结点的下一个结点;
- ②若 p 为空, 转⑤;
- ③若 count 等于 k , 则 q 指向下一个结点; 否则, count = count + 1;
- ④ p 指向下一个结点, 转②;
- ⑤若 count 等于 k , 则查找成功, 输出该结点的 data 域的值, 返回 1; 否则, 说明 k 值超过了线性表的长度, 查找失败, 返回 0;

⑥算法结束。

(3)算法实现: