数据结构考研大纲与考试范围解析

清航教育校长 清航教学研究中心主任 殷人昆

一．考研试题范围分析

虽然 2011 年计算机专业考研大纲在知识点上看没有变化，历年考试的出题范围基本上没有超出大纲。但由于大纲中知识点列举不很详细，从深度上到底达到什么程度，试题的重点到底在哪里，很多人心中没有数。因此，只能通过分析历年的试题才可略知一二。下表是对 2009 年和 2010 年数据结构考试出题范围的一个对比。

2009 年	2010 年
1. 队列的应用:缓冲区	1. 进栈与出栈序列
2. 栈与队列的特性和进栈/出栈序列	2. 限制一端输出的双端队列的出队序列
3. 二叉树前序、中序、后序、层次序遍历方式	3. 前序、中序、后序、层次序线索二叉树的表示
4. 二叉树的性质和完全二叉树的性质	4. 平衡二叉树的插入和平衡化旋转
5. 平衡二叉树的定义	5. m 叉树叶结点的计算
6. 森林的二叉树表示	6. Huffman 树的定义和特点
7. 无向连通图的定义、顶点与边的关系	7. 无向连通图和多重连通图
8. m 阶 B 树的定义	8. 拓扑排序序列
9. 堆的定义、插入和重新形成堆的调整方法	9. 折半查找的性能分析
10. 几种排序方法	10. 快速排序递归次数
	11. 几种排序方法

在选择填空题部分，各主要知识点分布如下表所示。

	2009 年	2010 年
栈、队列与双端队列	2	2
树与二叉树、森林	4	4
图	1	2
查找	1	1
排序	2	2

在综合应用题方面，主要知识点的分布如下表所示。

2009 年	2010 年
41. 求解图的最短路径	41. 散列表构造、数据存储及性能分析
42. 算法设计：在单链表中遍历求倒数第 k 个结点位置	42. 算法设计：将一维数组中所有元素循环左移 p 位

从近两年的试题范围的对比中，可以发现其中的互补性。以 2010 年的试题为例，选择填空题的知识点涉及 5 章，其中以“树与二叉树”这一章的分量最重，以“查找”这一章的分量最轻。而在综合应用题中恰恰填补了空白，一道题压在了“查找”这一章，一道算法题压在了“线性表”这一章，该章在选择填空题中一道题也没有。

2009 年的情况类似。从此我们得到第一个结论：*考试大纲的每一部分都可能出题*，而且从份量上来看，是一视同仁的，只有“树与二叉树”这一章题的数量会多一些。

从考试的难度来看，2009 年是第一次联考，试题的难度低一些，基本是难度低的题。但 2010 年试题的难度高了一些，特别是“双端队列”、“多重连通图”、“海豚算法”等，在多数学校不会讲到的或不做要求的，也赫然出现在试卷中。从此我们得出第二个结论：考试

大纲虽然列出了考试范围，实际上很多边边角角都要注意，押题是害人的，要*立足于扎扎实实理解和掌握好主要的知识点*。

二．风险和机遇

有些考生在考试结束之后给我打电话，说：“我辛辛苦苦地把严老师习题集中的题都做了，可使还是没考好！”我们在批改报考各个学校的考生的试卷时也是遗憾多于惊喜，有些试题所涉及的知识点明明在课堂上都讲过，为什么很多考生还是弄得一团浆糊？

机遇对所有的人都是存在的，关键在于如何把握。分析很多考生的试卷答案，我认为问题发生在考生个人。

可能的原因如下。

1．有些考生对知识点的掌握不深入。例如 2010 年 41 题计算散列表的大小，回答得完全不着调，不知如何计算。

2．有些考生虽然看了一些书，也做了不少练习。然而可能是复习的内容太陈旧，或者某些问题以前的教材没有涉及到，导致复习面狭窄。例如，多重连通图的概念，一些考生连题都看不懂，无从下手。

3．有些考生复习功夫不够，把希望寄托在押题上，许多最基本的知识都不会。例如，线索二叉树都不知道，甚至散列表都不会做，画了一个很长的表。要知道，这些可是老师上课讲过的东西，也是许多习题集里常有的东西。

4．有些考生考试审题不仔细，一看会做，急急忙忙写出答案，信心满满，以为能得高分，下来一对答案就傻了眼。例如，2009 年第 5 题，问一棵完全二叉树在第 6 层有 8 个叶结点，这棵完全二叉树最多有多少结点？有的考生一看，这还不容易，高兴起来就漏掉了题目中还有“最多”的提法，恰恰供选择的答案中有一个选项，正是不考虑“最多”情况的计算结果，有些考生立刻就选定了这个选项，当然就丢分了。

5．有些考生进考场后太紧张，心情不能放松，影响了正常发挥。本来会答的也想不起来了。有位考生考完见了我直拍脑袋，说：“考前不知复习多少遍，考试时不知怎的就是想不起来！”

总之，考不好的原因有主观的，有客观的，也有心理上的。这些都可以视为风险。我们应正视这些风险，研究应对这些风险的策略，才能很好地备考，打好考研这一大的战斗。

三．复习建议

学生在初学时往往感到《数据结构》课程内容多、面授容易接受，但自学难度大，特别是在编写小程序时常常无从着手。为以有限的时间和精力学好这门课程。应当注意以下几点，有助于改善自学效果。

1．在学习本课程之初，必须注意复习在用 C/C++语言编写小程序时的语法规则和方法。为本课程的学习打下基础

C 语言与 Pascal 语言一样，是一种面向过程的语言。C 程序结构的特点是遵循“输入-处理-输出”的模式来解决问题。C++保留了 C 的面向过程编程的成分但引入了面向对象的成分，在复习 C/C++语言时，要注意：

(1) 函数的概念和相关问题。包括函数类型，函数特征，函数参数的传递。特别注意传值参数和引用参数在使用上的区别。还有函数的不同返回值类型间的区别。

(2) 函数中局部变量的作用域，它的创建和回收的有效范围。特别注意在函数中对局部变量的任何改变，因在退出函数过程时局部变量被释放而不能当作函数返回值返回。

(3) 类型（或类）的定义方式。使用 C 定义结构的复杂性远低于 C++或 Java，但要注意 C 指针的使用所带来的复杂性。

(4) 在 C/C++中的动态存储分配和动态存储回收方式。

(5) 程序中定义的变量或类的实例必须在使用前初始化，特别是用指针动态定义的结构

或数组，必须在使用前为它们分配存储空间，并在动态分配后判断这次分配是否成功。只有动态分配成功才能继续程序中要做的事情，但别忘了赋初值。

(6) 在 C/C++中的输入/输出文件的定义和使用。特别注意文件的连接、文件的打开与关闭、文件模式的作用。

2．在备考时首要任务是选好教材和辅导书。教材不一定采用某些所谓“权威”的教材，因为可能在内容上不够全面，在阐述上不够深入，不能反映较新的发展。在 C 语言版的教材方面，我个人比较倾向于选北京大学张乃孝的《算法与数据结构》（高教版）、西北大学耿国华的《数据结构》（高教版）、南京邮电大学陈慧南的《数据结构》（高教版）、清华大学朱明芳和吴及的《数据结构与算法》（清华版）。辅导书可选用的较多，它的作用在于提供考研的指导和相关可利用的习题。

3．在复习《数据结构》时，要注意知识体系

数据结构课程中的知识本身具有良好的结构性。从总体上来说，课程的主要内容是围绕着数组、线性表、栈和队列、树与二叉树、图、集合与查找结构等常用的数据结构和查找、排序等常用算法来组织的。其中有些结构是面向应用的，有些结构是面向实现的。在复习中要注意这两个层次以及它们之间的联系。

4．注意比较

在复习中应当注意从“横向”和“纵向”进行对比以加深理解。

纵向对比将一种结构与它的各种不同的实现加以比较，或从一般/特殊的角度、整体/部分的角度、实参/形参的角度建立相关类型的变量之间的联系；

横向对比包括具有相同逻辑结构的不同数据结构（如线性表、栈、队列和串）的比较，具有相同功能的不同算法的比较等。

5．注意复习和重读

有些内容在初读时难以透彻理解或熟练掌握，或者看起来似乎明白了，但用的时候容易犯晕。在继续学习的过程中如遇到或用到有关内容时，应当及时复习或重读，这往往能够化难为易，温故知新。

6．注意循序渐进

在进入具体内容复习之初，首先领会基本概念、基本思想，这一点极为重要。特别是在阅读算法之前，一定要先弄清其基本思想、基本步骤，这将大大降低理解算法的难度。如果读“懂”了算法而不知其基本思想，仍不算是真懂。可以通过事例学习以加深理解。

7．注意练习

习题练习是课程的基本要求之一。只看书不做题，不可能真正学会有关知识，更不能达到技能培养的目的。另一方面，做题也是自我检查的重要手段。此外，在做算法设计型的习题时，应优先考虑数据结构的定义，可以直接使用以前定义的数据类型和相应的操作，综合已学过的知识，以提高编程的能力。

8．算法思路的理解

算法及其思想、评价是本课程的重点之一，在课文中占了相当大的比例。在学习每一个算法时，建议首先理解问题的要求，在此基础上利用一个事例来模拟问题的求解，自己试着构思一下解题的思路，能够写出来更好。然后再阅读算法，在读懂之后，不要急于往下进行，而应停下来思考下列问题：该算法从逻辑上可以划分为哪几个部分（步骤）？每一部分的功能是什么？这一功能又是如何实现的？能否有别的方法实现？各部分之间的关系如何？如果问题的要求有所不同，应当如何修改算法？

将思考的结果记载下来，在复习时会有很大帮助。另外，这样的分析也是灵活运用所学知识的关键。

9．努力培养算法设计的能力

清航考研

在课程复习中最令学习者感到吃力的是设计和编写算法。算法设计水平是软件设计的基础。要提高算法的设计水平，首先需要掌握问题要求的基本内容，通过反复体会和练习来实现。通常需要根据具体问题的要求，选择合适的数据结构，设计有效的算法。有条件的学生应当在机器上多做练习。

10．在复习完每一章、节后，要及时地进行总结，用简短的文字记录这部分的内容，尽可能用自己的话复述一遍。在本课程全部学完后应对本课程进行全面系统的复习和总结，认真做模拟试卷。在进行总复习时，可通过对比各种数据结构的异同和他们之间的相互关系，加深对各种数据结构的理解。

在复习时，可以按所记录的要点反向地进行，即依据要点找出其具体内容。按这样的方法进行，也许还能够节省时间，提高学习效果。

最后应当强调的是，学习方法主要靠自己摸索，多总结，多思考，勤上机，勤交流，是把数据结构这门课程真正学好的关键。

四．考试指导

数据结构考试可能的题型有以下 2 种：

(1) 选择填空题：给出一些有关数据结构性质、特点及一些简单算法性能的不完全叙述，要求学生从题后给出的供选择的答案中选择合适的答案，补足这些叙述。

(2) 综合应用题：又有 2 种题型：

✧ 简答题：应用作图方法、简单计算或证明方法，使用给定数据建立或操作一些数据结构，或判断某些算法的正误。

✧ 编程题：给出算法设计要求，编制出部分算法程序，用来考察几个知识点的综合应用。

下面根据各种题型举例分析，说明解题方法及考试时的注意事项。

例题 1 单项选择题

以下关于图的遍历的说法中不正确的是（　　）。

A．遍历图的过程实质上是对每个顶点查找其邻接顶点的过程

B．深度优先搜索和广度优先搜索对无向图和有向图都适用

C．深度优先搜索和广度优先搜索访问顶点的顺序不同，它们的时间复杂度也不同

D．深度优先搜索需要使用一个递归栈，广度优先搜索需要使用一个辅助队列

【答案】C。

【解析】采用排除法解答。

深度优先搜索和广度优先搜索都是从某个顶点出发，访问它的没有访问过的邻接顶点，一直搜索下去，选项 A 正确。

这两种搜索对无向图和有向图都适用，选项 B 正确。

深度优先搜索是一个递归的过程，需要一个递归栈，广度优先搜索是一个分层访问的过程而非递归过程，需要用队列逐层保存结点，选项 D 正确。

最后剩下选项 C，不对。原因是若两个算法都用邻接表存储图时，它们的时间复杂度相同，都是 O(n+e)。其中 n 是顶点数，e 是边数。

例题 2 单项选择题

若设 S 是一个已有 10 个元素的顺序栈，栈中元素依次是 $A_1, A_2, \ldots, A_{10}$，栈顶在 A_{10}。Q 是一个已有 10 个元素的循环队列，队列中元素依次是 $B_1, B_2, \ldots, B_{10}$，队头是 B_1。如果想要把栈 S 中元素全部移入队列 Q，且使得队列中的元素依次排列为 $B_1, A_1, B_2, A_2, \ldots, B_{10}, A_{10}$，需要执行的栈和队列的操作的次数为（　　）。

A．100　　B．1000　　C．145　　D．290

【答案】A。

【解析】不需要使用排除法，可做具体计算。如果想在队列中得到 $B_1, A_1, B_2, A_2, \ldots B_{10}, A_{10}$，需首先把栈中的元素逆置为 $A_{10}, A_9, \ldots, A_1$，再从队列和栈中交替退出 $B_1, A_1, \ldots, B_{10}, A_{10}$ 并存入队列即可。为逆置栈中元素，可将它们按 $A_{10}, A_9, \ldots, A_1$ 退栈，并进队；再从队列中按 $A_{10}, A_9, \ldots, A_1$ 顺序退出并进栈就可以了。操作步骤如下：

(1) 栈中 10 个元素退栈，再进队。栈空，队列中有 $B_1, B_2, \ldots, B_{10}, A_{10}, A_9, \ldots, A_1$。

(2) 队列中前 10 个元素出队，再进队，后续 10 个元素出队，再进栈。栈中有 10 个元素，依次为 $A_{10}, A_9, \ldots, A_1$，队列中有 10 个元素，依次为 $B_1, B_2, \ldots, B_{10}$。

(3) 队列和栈中元素轮替出队、出栈，再进队。队列中元素依次为 $B_1, A_1, B_2, A_2, \ldots, B_{10}, A_{10}$。

总共有 20+40+40 = 100 次栈和队列的操作。

【短评】此类题主要是考核学生对基本知识的掌握程度，涉及范围较广。所有各章内容都可能牵涉到。主要考查对各种数据结构的定义、特点、性质（包括公式计算）、主要操作的性能分析（包括算法中多趟执行时每一趟的通项公式），因此要求复习时必须仔细看书。不提倡死记硬背，但要学会推导。另外，对待每一道题，特别涉及计算和作图的题目，一定要在草稿纸上试做，切忌轻敌。其实所谓“粗心”就是轻敌的结果。

例题 3 综合应用题

设一段正文由字符集{A, B, C, D, E, F }中的字母组成，这 6 个字母在正文中出现的次数分别为 {16, 20, 24, 6, 5, 29 }。

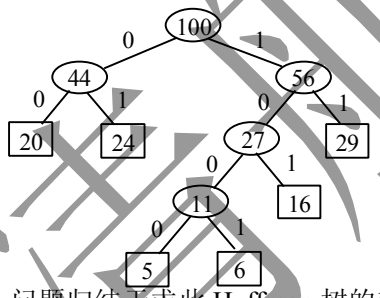
(1) 为这 5 个字母设计 Huffman 编码。

(2) 设每个字节由 8 个二进制位组成，试计算按（1）所得 Huffman 编码存储这段正文，总共需要多少个字节？

(3) 若这段正文开始部分的二进制编码序列为 01101100100011011000，请按（1）所得 Huffman 编码，将其译为正文。

【参考答案】

(1) 首先画出 Hffman 编码树，如图：



字母	A	B	C	D	E	F
编码	101	00	01	1001	1000	11

(2) 问题归结于求此 Huffman 树的带权路径长度

WPL = 3*16+2*20+2*24+4*6+4*5+2*29 = 238（bits）

共需字节数 = 238/8 = 29.75（字节）

(3) 二进制编码序列 01 101 1001 00 11 01 1000 转换成正文为 CADBFCE。

【短评】此类题要求对各章中许多细节比较清楚。事实上，数据结构课程中讲到的许多数据结构的细节是最基础的知识，将这些内容理解透彻了，以后学生在自主开发软件时将会得到回报的。在解答此类题时，不要立即提笔就答，应当把题看完，回忆课程相关知识点和解决问题的思路，如果做过类似的题目，就更好了。

例题 4 综合应用题

如果一棵二叉树采用二叉链表存储，试设计一个算法。查找并返回其后序序列中的第一个结点的地址，要求既不用递归也不用栈。

【参考答案】既不用递归也不用栈查找二叉树的后序序列的第一个结点是可行。算法的主要想法是，如果二叉树非空，可从根结点出发，沿左子女链一直走到底，如果最后到达的

结点的右指针为空，则该结点即为其后序序列的第一个结点；如果该结点的右指针非空，则对该结点的右子树施行同样的操作。

算法的描述如下。

```
BiTNode * postFirst ( BiTNode *T ) {
    BiTNode *p = T;
    while ( p != NULL ) {
        while ( p->lchild != NULL ) p = p->lchild;
        if ( p->rchild == NULL ) return p;
        else p = p->rchild;
    }
    return NULL;
};
```

【短评】编写算法的题可能是学生比较棘手的问题，特别是在考试这样一个氛围，时间又短促，想编出一个好算法不太容易。一个建议是首先仔细阅读试题，了解它到底要你干什么。然后用一个简单的例子走一下，总结每一步向下走用什么语句，再做归纳。也可以按照结构化程序设计的方法，先搭框架，再根据例子填入细节。总之，要想又快又好地编写出算法，还要靠平时的基本训练，多做题，自主做题，各种题型都见过了，考试时自然文思泉涌，笔下就流畅了。

五．结束语

数据结构课程是一门知识性、实践性都很强的课程。数据结构及算法编写与分析都是不容易的。学好它需要付出极大的劳动，在平时勤学多练。如果有碰运气或取巧的想法，十有八九通不过考试。只有基础打扎实，考试才能顺利过关。就怕平时不看书，不做作业或抄别人的作业，到考试时照样不会，自然一次次通不过了。最后，祝愿考生们通过自己的勤奋努力，取得优良的成绩。

清航考研