数据结构习题课讲义

朴珉赫

基本概念，基本数据结构定义及其主要操作，基本算法流程及其数据结构

第一章

时空复杂度分析，基本概念（逻辑结构，物理结构，数据的运算）

下列程序段的时间复杂度是。

```
int sum = 0;
for (int i = 1; i < n; i *= 2)
    for (int j = 0; j < i; j++)
        sum++;
```

对外层循环的i，第k次迭代时，我们有$i = 2^{k-1}$，因此保证$i < n$，则$2^{l-1} \leq n-1$，从而$i = 2^{\lfloor log(n-1) \rfloor}$。

对内层循环的j，对某个i，我们有$j = i - 1 = 2^{k-1} - 1$，因此有如下计算式：

$$\sum_{k=1}^{\lfloor log(n-1) \rfloor + 1} \sum_{j=0}^{2^{k-1}-1} 1 = 2^{\lfloor log(n-1) \rfloor + 1} - 1 \leq 2^{log(n-1)+1} - 1 = 2(n-1) - 1 = O(n)$$

第二章

本课程最最最重要的一章，务必掌握其中所有内容，要熟练掌握数据结构定义，操作及其时间复杂度，基本应用，常见算法题等等，建议要有一定量的练习，特别是算法设计题，本章节必考算法题。

已知由n（n>=2）个正整数构成的集合A，将其划分成两个不相交的子集A1和A2，元素个数分别为n1和n2，A1和A2中元素之和分别为S1和S2。设计一个尽可能高效的划分算法，满足|n1-n2|最小且|S1-S2|最大。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++语言描述算法，关键之处给出注释。

3）说明你所设计算法的平均时间复杂度和空间复杂度。

解：

1）根据快速排序的思想，把找到最佳的划分，把最小的[n/2]个数放到A1，其余的数放到A2。分组结果即为题意所求。

算法步骤：

1）若i=[n/2]，则划分结束。

2）若i<[n/2]，则枢轴及之前的所有元素均属于A1，继续对i之后的元素进行划分。

3）若i>[n/2]，则枢轴及之后的所有元素均属于A2，继续对i之前的元素进行划分。

```
int setPartition(int a[],int n){
    int pivotkey,low=0,low0=0,high=n-1,high0=n-1,flag=1,k=n/2,i;
    int s1=0,s2=0;
    while(flag){
        pivotkey=a[low]; //选择枢轴
        while(low<high){  //基于枢轴对数据记性划分
            while(low<high && a[high]>=pivotkey) --high;
            if(low!=high) a[low]=a[high];
            while(low<high && a[low]<=pivotkey) ++low;
            if(low!=high) a[high]=a[low];
        }
        a[low]=pivotkey;
        if(low==k-1) //若枢轴是第n/2个元素，划分成功
        flag=0;
        else if(low<k-1){
            low0=++low;
            high=high0;
        }else{
            high0=--high;
            low=low0;
        }
    }
    for(i=0;i<k;i++) s1+=a[i];
    for(i=k;i<n;i++) s2+=a[i];
    return s2-s1;
}
```

（3）算法的平均时间复杂度为O(n),空间复杂度为O(1).

第三章

本章核心内容，要熟练掌握栈和队列的性质，实现和应用，可以熟练使用它们，可以做点习题，比如选择题练练手。主要题型可能就是相关应用及其算法实现，或者给个入队入栈队列，给出输出等

略

第四章

本章的核心内容就是数数

5.3 设有上三角矩阵 An×n，

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ & a_{22} & a_{23} & \dots & a_{2n} \\ & & a_{33} & \dots & a_{3n} \\ & C & & \dots & \dots \\ & & & & a_{nn} \end{bmatrix}$$

将其上三角的元素逐行存于数组B[0..m-1]中（m充分大），使得B[k]=a_{ij}且k＝f_1(i)+f_2(j)+c。试推导出函数f_1、f_2和常数c（要求f_1和f_2中不含常数项）。

对矩阵A中任意一个元素a_{ij}，我们知道，它前面共有$i-1$行，则前$i-1$行一共有$n+(n-1)+\ldots+(n-(i-2))=\frac{(2n-(i-2)(i-1))}{2}$个元素。在第$i$行中，$a_{ij}$之前共有$j-i$个元素。因此$a_{ij}$在矩阵中的位置为：

$$p=(i-1)\times(n-\frac{i}{2}+1)+(j-i)+1=-\frac{i^2}{2}+(n+\frac{1}{2})i+j-n$$

又数组从0开始计数，因此序号为位次减1，有：

$$k=-\frac{i^2}{2}+(n+\frac{1}{2})i+j-n-1$$

$$f_1(i)=-\frac{i^2}{2}+(n+\frac{1}{2})i$$

$$f_2(j)=j$$

$$c=-n-1$$

第二到第四章实际上核心是线性表，我们数据结构课程的核心其实也是线性表。我们可以看到，第二章介绍了线性表这种逻辑结构的两种物理结构，也就是顺序表和链表。第二章的栈和队列，实际上是受限的线性表，我们施加一些约束，让线性表拥有了某种优良性质。第四章的数组，其实就是推广的线性表，将很多一样的线性表排列起来，组合成了多维数组，一维数组其实也就可以看做线性表了，为了方便，一般选取可以随机存取的顺序表实现数组。

第五章

除了第二章外最重要的一章，算法题必考。要掌握树的定义及其性质，二叉树的定义及其性质，各种遍历算法，注意二叉树一些独特的性质，比如一定要区分左右子树，因此画图时也要体现出来左右的区分。要掌握树的各种计算题。要注意森林的定义及其性质，注意各种表示法及其转换，注意各种遍历算法。注意树，森林和二叉树的转换。要掌握哈夫曼树。二叉树算法极大概率考察，也将是日后各位学习过程中必须要掌握的一类算法，它们的核心是掌握递归，或者说遍历算法，要注意返回值和边界条件。但同样也会有非递归算法，比如基于层次遍历的推广的算法。

6.4 一个深度为 H 的满 k 叉树有如下性质：第 H 层上所有结点都是叶子结点，其余各层上每个结点都有 k 棵非空子树。如果从 1 开始按自上而下、自左向右的次序对全部结点编号，问：

(1) 各层的结点数目是多少？

(2) 编号为 i 的结点的父结点(若存在)的编号是多少？

(3) 编号为 i 的结点的第 j 个孩子(若存在)的编号是多少？

(4) 编号为 i 的结点有右兄弟的条件是什么？其右兄弟的编号是多少？

1、对第n层，有k^{n-1}个结点

2、结点i的父节点为$\lfloor \frac{i-2}{k} \rfloor + 1$

证明：假设结点i在第n行，$(n = 1, 2, 3...)$，则其双亲结点j在第$n-1$行。我们知道以下结论：前$n-2$行一共有$\frac{k^{n-2}-1}{k-1}$个结点，前$n-1$行一共有$\frac{k^{n-1}-1}{k-1}$，则在第n层中结点i之前一共有$i - \frac{k^{n-1}-1}{k-1} - 1$ 个结点。

我们又有以下结论：若第$n-1$行的结点j前有m个结点，第n行结点i前有n个结点，结点i是结点j的第l个海泽，则有关系：$n = m * k + l - 1$

因此对结点j，在第$n-1$行中，它前面有$\lfloor \frac{i - \frac{k^{n-1}-1}{k-1} - 1}{k} \rfloor$个结点，那么总的来看，它前面有$\frac{k^{n-2}-1}{k-1} + \lfloor \frac{i - \frac{k^{n-1}-1}{k-1} - 1}{k} \rfloor$个结点，化简得：$\lfloor \frac{i-2}{k} \rfloor$个结点，由于从1开始编号，那么$j$的编号为$\lfloor \frac{i-2}{k} \rfloor + 1$

3、 结点i的第j个孩子结点编号为$k * (i-1) + j + 1$

4、当$(i-1)\%k!=0$时，结点i有右兄弟，其右兄弟为$i+1$

这个条件其实是保证i不是某个结点的最右侧的孩子，对于满k叉树，某个结点的最右侧孩子的编号减1之后是可以被k整除的，因为上面所有层的结点都有k个孩子

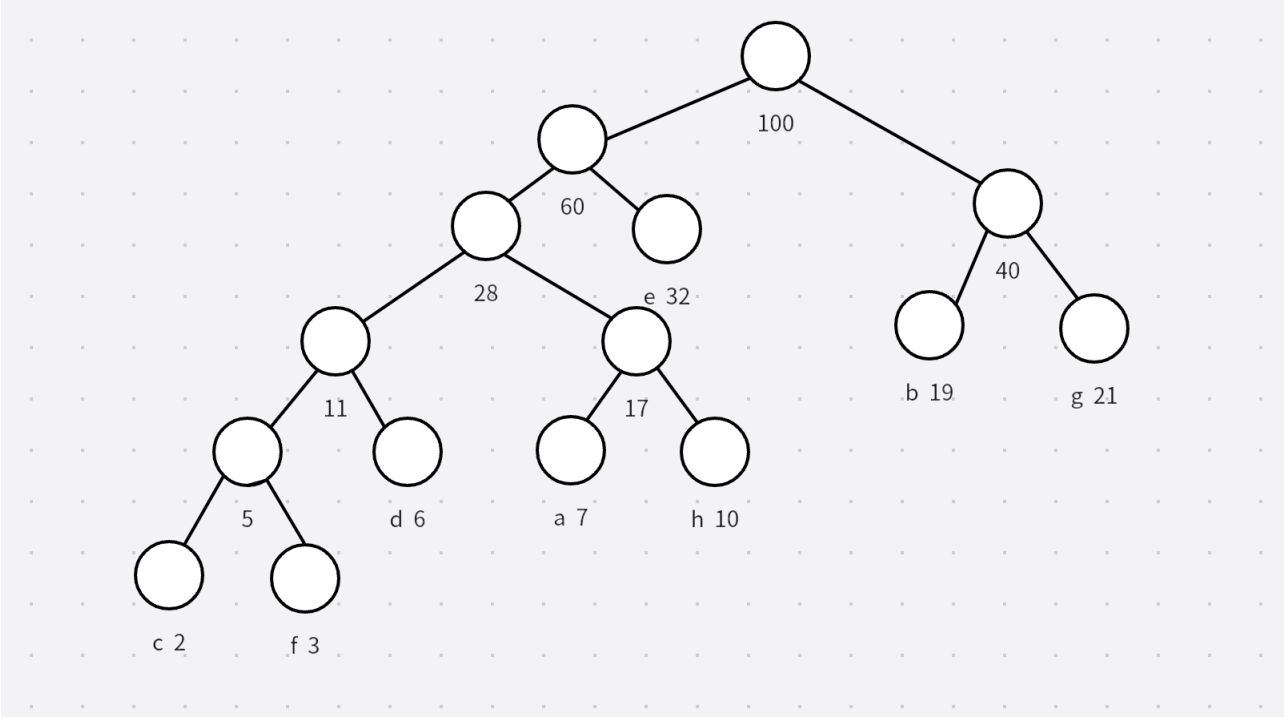
6.14 假设某个电文由(a, b, c, d, e, f, g, h)8 个字母组成，每个字母在电文中出现的次数分别为(7, 19, 2, 6, 32, 3, 21, 10)，试解答下列问题：

(1) 画出出 huffman 树；

(2) 写出每个字母的 huffman 编码；

(3) 在对该电文进行最优二进制编码处理后，电文的二进制位数。

1、如图，编码应该是左0右1，左1右0也可。



2、编码：

```
a:0010
b:10
c:00000
d:0001
e:01
f:00001
g:11
h:0011
```

3、电文位数为：$5*（2+3）+4*(6+7+10)+2*(32+19+21)=261$

注：本题非常重要，务必掌握

6.3 描述满足下列条件的二叉树形态：

(1) 先序遍历序列与中序遍历序列相同；

(2) 后序遍历序列与中序遍历序列相同；

(3) 先序遍历序列与后序遍历序列相同；

1、任意一个结点都没有左子树，或者为空树

2、任意一个结点都没有右子树，或者为空树

3、只有一个结点，或者为空树

第六章

本章概念繁多，需要牢记，两种主要的数据存储方式要注意，十字链表最好要会画。在编程实践中，邻接表实际上也可以用不同的方式去表达，比如所谓链式前向星，我们还可以直接存边。考试中仅允许使用邻接矩阵和邻接表。DFS和BFS要掌握，各种图算法要熟练手动运行。这一章应该是占50分的应用题中的主要部分，小题中也会考察。

7.1 已知有向图如图 7-1 所示，

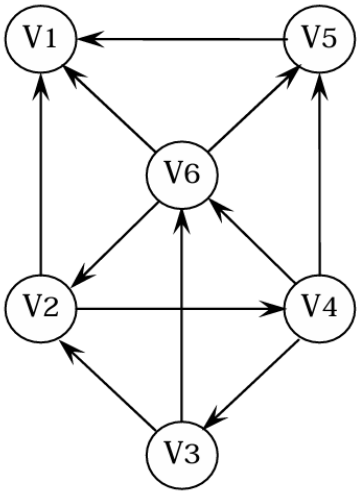
请给出该图的

(1) 邻接矩阵示意图

(2) 邻接表示意图

(3) 逆邻接表

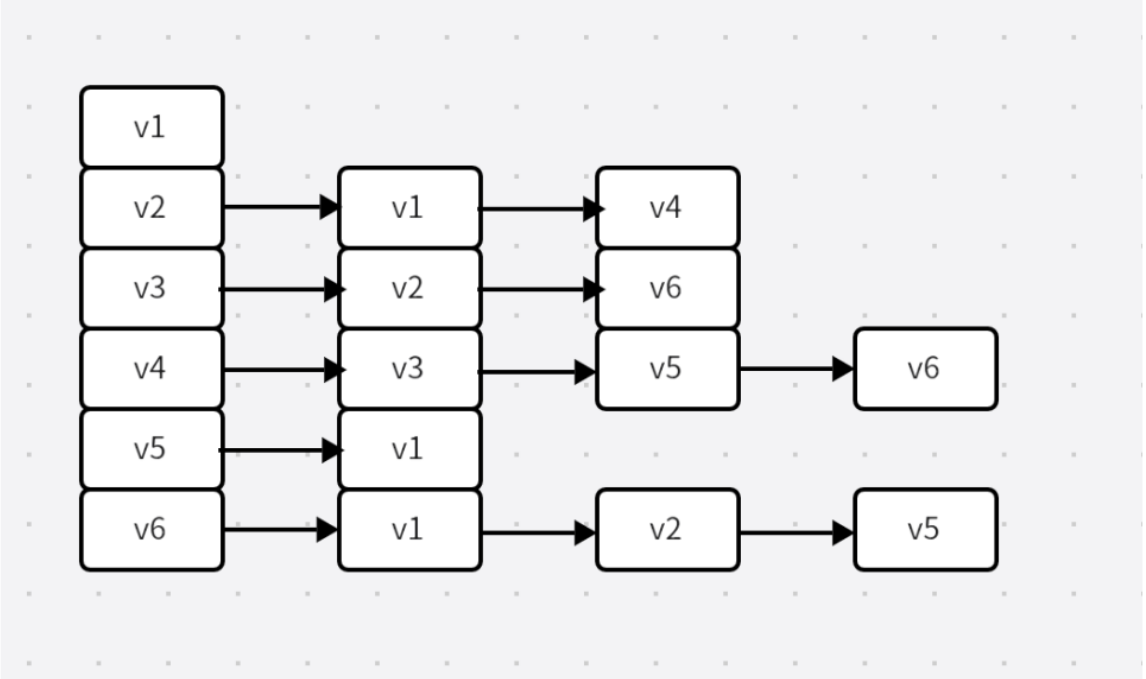
(4) 所有强连通分量



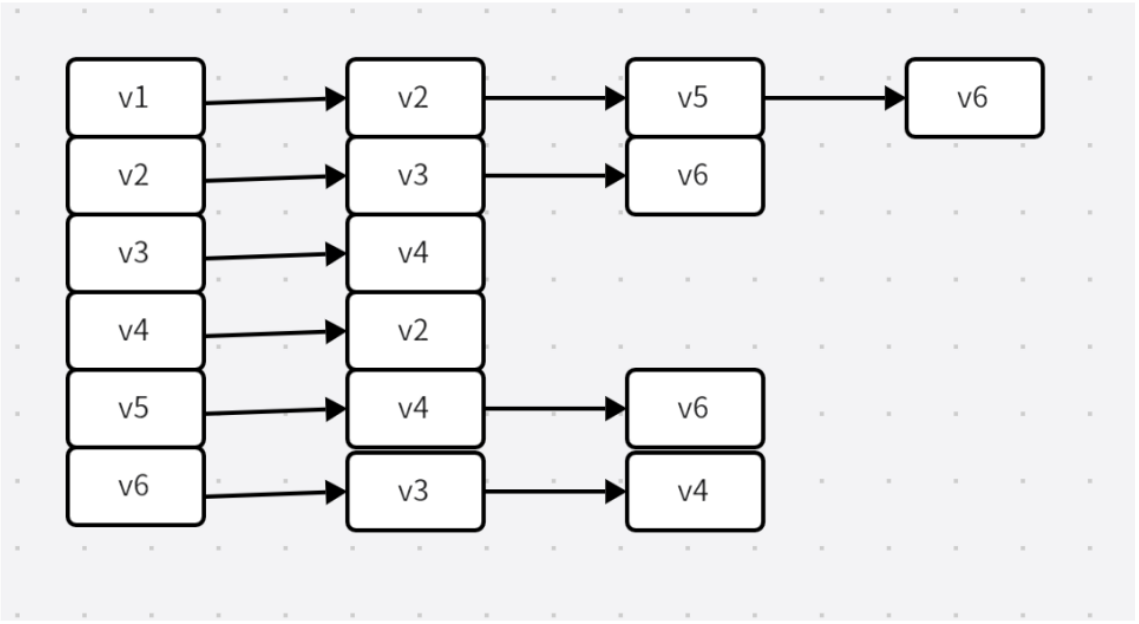
1、

0	0	0	0	0	0
1	0	0	1	0	0
0	1	0	0	0	1
0	0	1	0	1	1
1	0	0	0	0	0
1	1	0	0	1	0

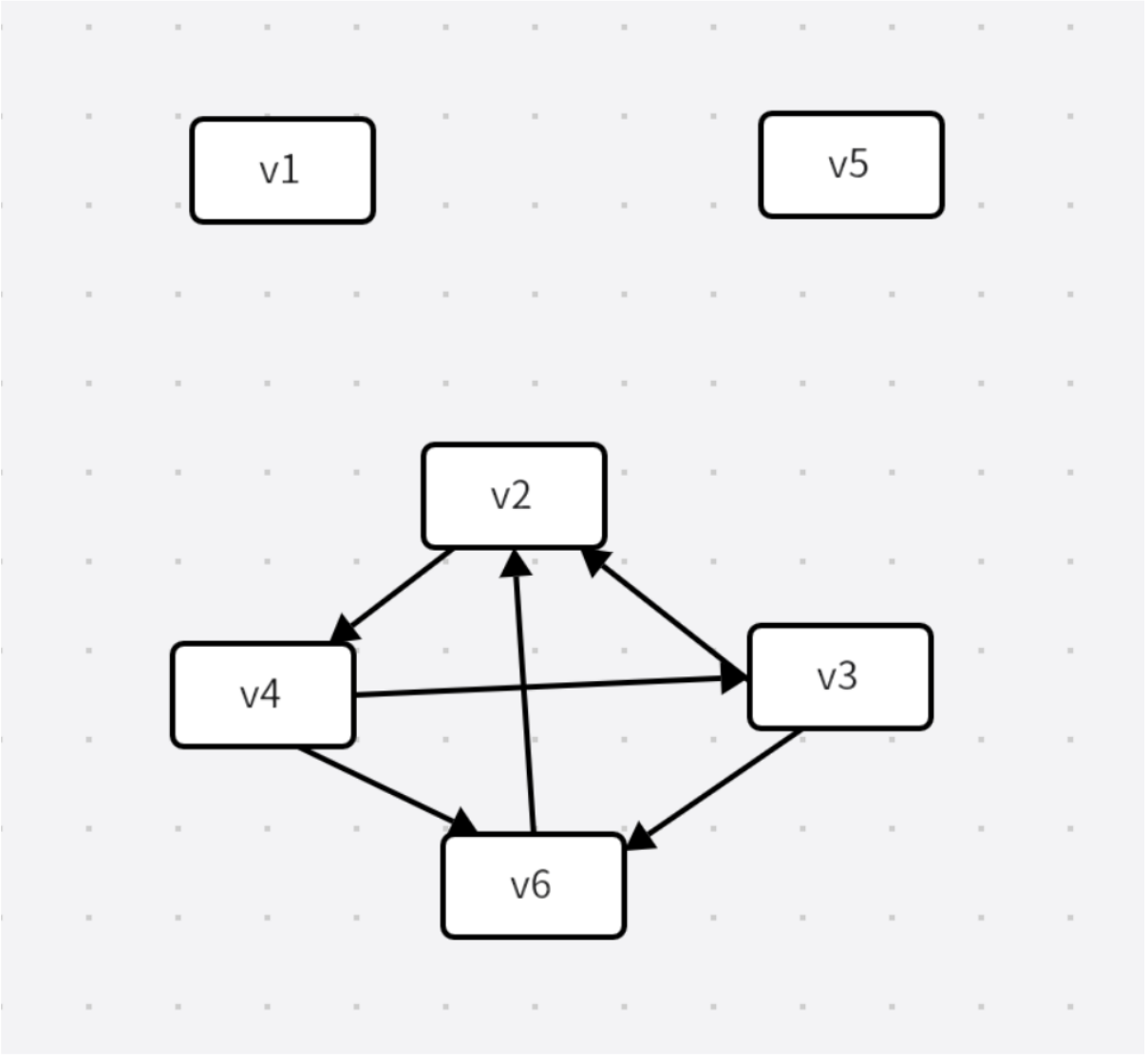
2、



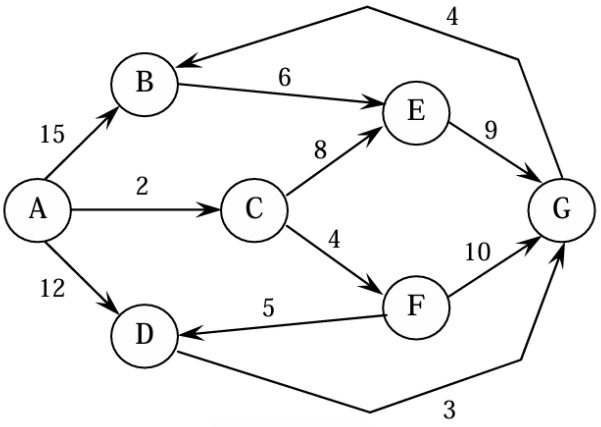
3、



4、



7.5 试利用 Dijkstra 算法求图 7-5 中顶点 A 到其他各顶点之间的最短路径。要求写出执行算法过程中，数组 D、P 和 S 各步的状态。



1

	S	D	P		
A	1	0			
B	0	15	0	1	
C	0	2	0	2	
D	0	12	0	3	
E	0	∞			
F	0	∞			
G	0	∞			

2

	S	D	P		
A	1	0			
B	0	15	0	1	
C	1	2	0	2	
D	0	12	0	3	
E	0	10	0	2	4
F	0	6	0	2	5
G	0	∞			

3

	S	D	P				
A	1	0					
B	0	15	0	1			
C	1	2	0	2			
D	0	11	0	2	5	3	
E	0	10	0	2	4		
F	1	6	0	2	5		
G	0	16	0	2	5	6	

4

	S	D	P				
A	1	0					
B	0	15	0	1			
C	1	2	0	2			
D	0	11	0	2	5	3	
E	1	10	0	2	4		
F	1	6	0	2	5		
G	0	16	0	2	5	6	

5

	S	D	P				
A	1	0					
B	0	15	0	1			
C	1	2	0	2			
D	1	11	0	2	5	3	
E	1	10	0	2	4		
F	1	6	0	2	5		
G	0	14	0	2	5	3	6

6

	S	D	P				
A	1	0					

	S	D	P				
B	0	15	0	1			
C	1	2	0	2			
D	1	11	0	2	5	3	
E	1	10	0	2	4		
F	1	6	0	2	5		
G	1	14	0	2	5	3	6

7

	S	D	P				
A	1	0					
B	1	15	0	1			
C	1	2	0	2			
D	1	11	0	2	5	3	
E	1	10	0	2	4		
F	1	6	0	2	5		
G	1	14	0	2	5	3	6

第七章

本章主要是搜索。线索比较明确，主要内容比如顺序查找，二分查找，二叉排序树，平衡二叉排序树，哈希等。大多数要求手动运行，但涉及到二叉树的算法题也值得注意。

9.4 已知如下所示长度为 12 的表：

(Jan, Feb, Mar, Apr, May, Jun, July, Aug, Sep, Oct, Nov, Dec)

表中，每个元素的查找概率分别为：

(0.1, 0.25, 0.05, 0.13, 0.01, 0.06, 0.11, 0.07, 0.02, 0.03, 0.1, 0.07)

(1)若对该表进行顺序查找，求查找成功的平均查找长度；

(2)画出从初态为空开始，依次插入结点，生成的二叉排序树；

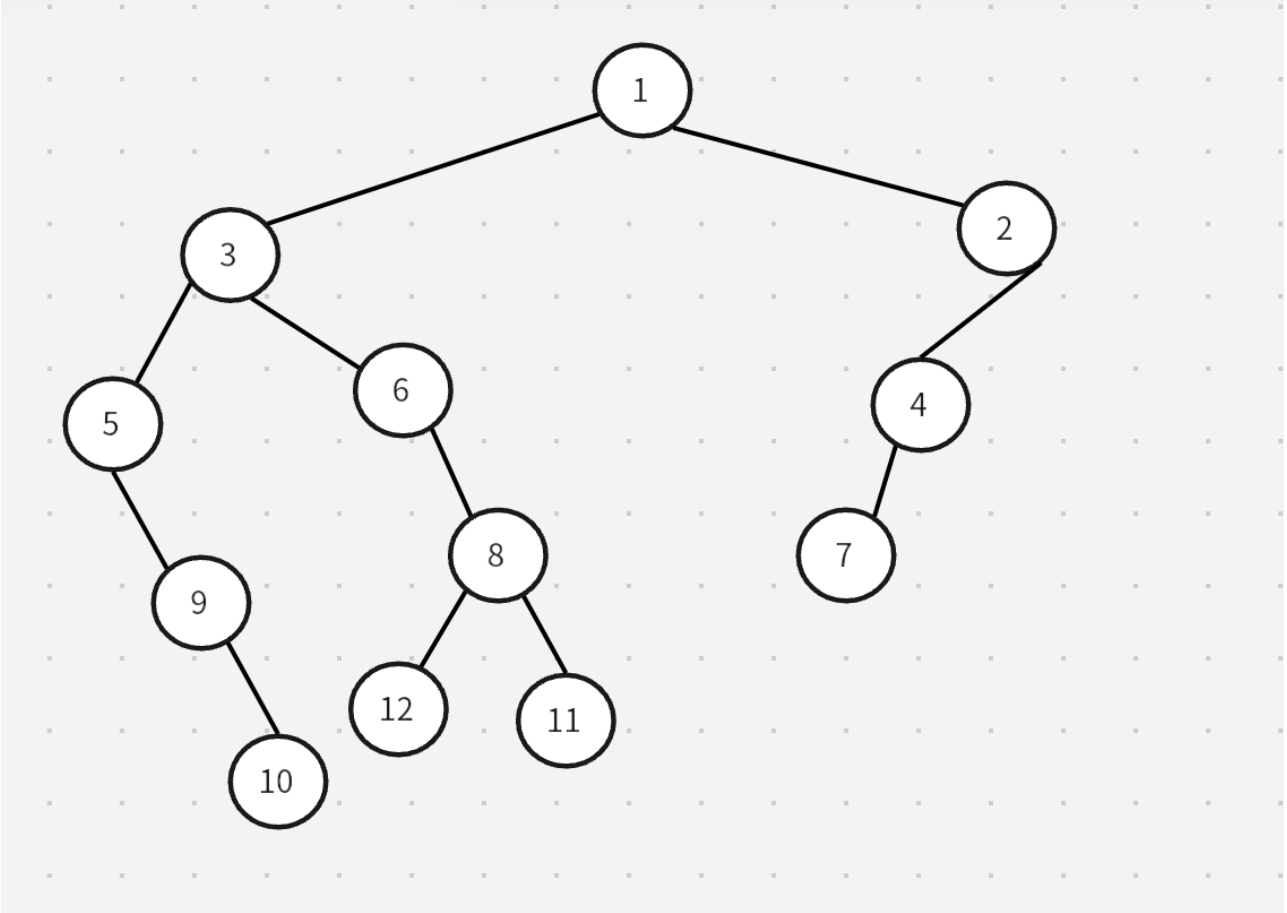
(3)计算该二叉排序树查找成功的平均查找长度；

(4)将二叉排序树中的结点 Mar 删除，画出经过删除处理后的二叉排序树。

(1)

ASL = 0.1x1+0.25x2 +
0.05x3+0.13x4+0.01x5+0.06x6+0.11x7+0.07x8+0.02x9+0.03x10+0.1x11+ 0.07x12 = 5.43

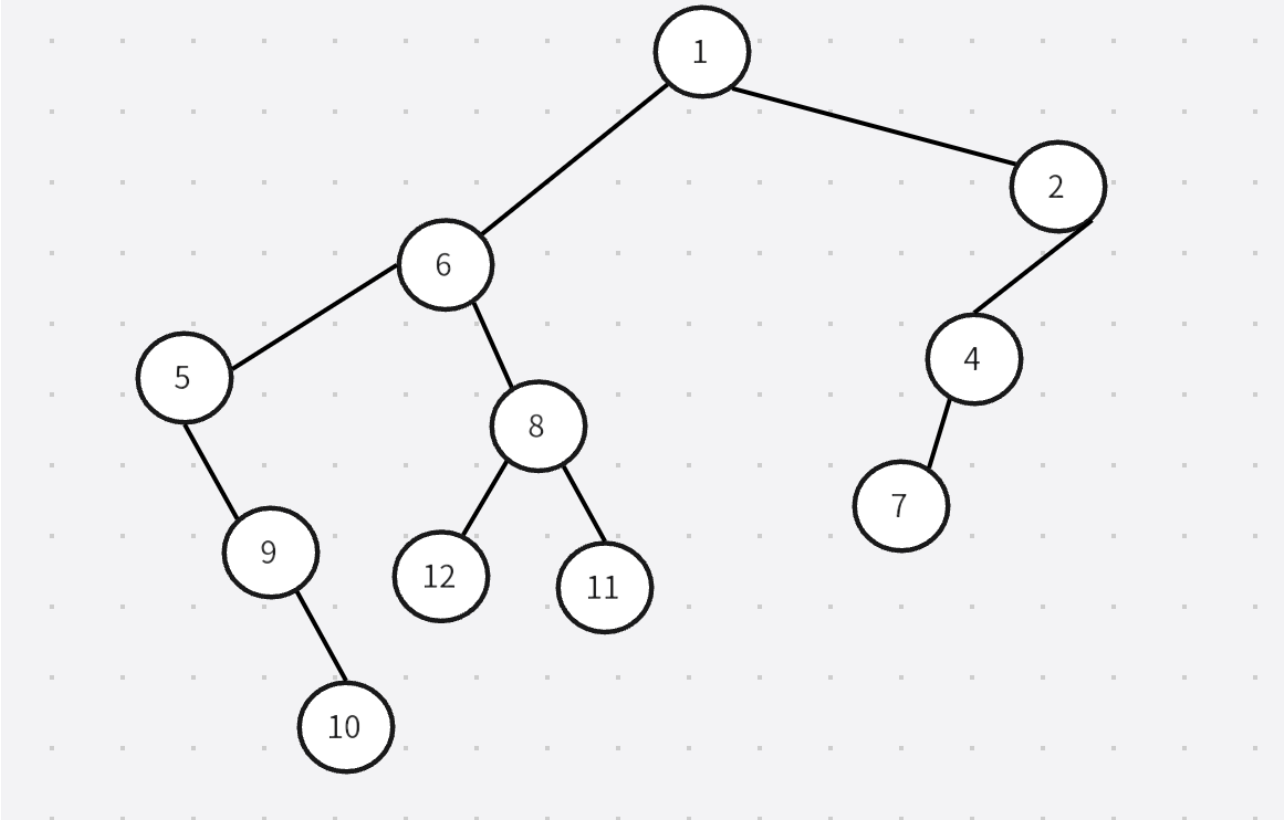
(2)



(3)

ASL = (1x0.1+2x0.3+3x0.2+4x0.2+5x0.2)=3.1

(4)



9.5 已知关键字序列{10, 25, 33, 19, 06, 49, 37, 76, 60}，哈希地址空间为 0～10，哈希函数为 H (Key)=Key % 11，求：

（1） 用开放定址线性探测法处理冲突，构造哈希表 HT1，分别计算在等概率情况下 HT1 查找成功和查找失败的 ASL；

（2） 用开放定址二次探测法处理冲突，构造哈希表 HT2，计算在等概率情况下 HT2 查找成功的 ASL；

（3） 用拉链法解决冲突，构造哈希表 HT3，计算 HT3 在等概率情况查找成功的 ASL。

(1)

0	1	2	3	4	5	6	7	8	9	10
33	76		25	37	49	06	60	19		10

成功ASL = (1x7+3x2)/9 = 13/9

失败ASL = (3+2+1+7+6+5+4+3+2+1+4)/11 = 38/11

(2)

0	1	2	3	4	5	6	7	8	9	10
33	60		25	37	49	06		18	76	10

ASL = (7 + 3+5)/9 = 5/3

(3)

这个问需要把图画出来，画邻接表那样的

0	1	2	3	4	5	6	7	8	9	10
33	60		25	37	49，60	06		18		10,76

ASL = (7x1+2x2)/9 = 11/9

9.7 写出判别一棵二叉树是否为二叉排序树的算法，设二叉排序树中不存在关键字值相同的结点。

```
ing judge(BiTree T)
{
    int b1,b2;
    if(!T)
        return 1;
    b1 = judge(T->lchild);
    if(b1==0||T->data<pre)
        return 0;
    pre = T->data;
    b2 = judge(T->rchild);
    return b2;
}
```

第八章

这一部分比较简单，也不怎么单独考察大题，但是思想非常重要，特别是快排，堆排和归并。快排在涉及到顺序表的算法题中比较常见，或对顺序表排序，或利用其思想进行查找或者找第k大的数之类的。堆排也是一样，我们可以利用它的性质实现优先队列，将一些操作的时间复杂度从n降到logn。手动运行了解即可。注意掌握后文中的排序圣经。

10.3 判别以下序列是否为堆（小顶堆或大顶堆），若不是，则吧它调整为堆

(1) (96, 86, 48, 73, 35, 39, 42, 57, 66, 21)；

(2) (12, 70, 33, 65, 24, 56, 48, 92, 86, 33)；

补充题：

1、下表给出了某工程各工序之间的优先关系和各工序所需的时间，其中空值表示无先驱工序，请完成以下各题：

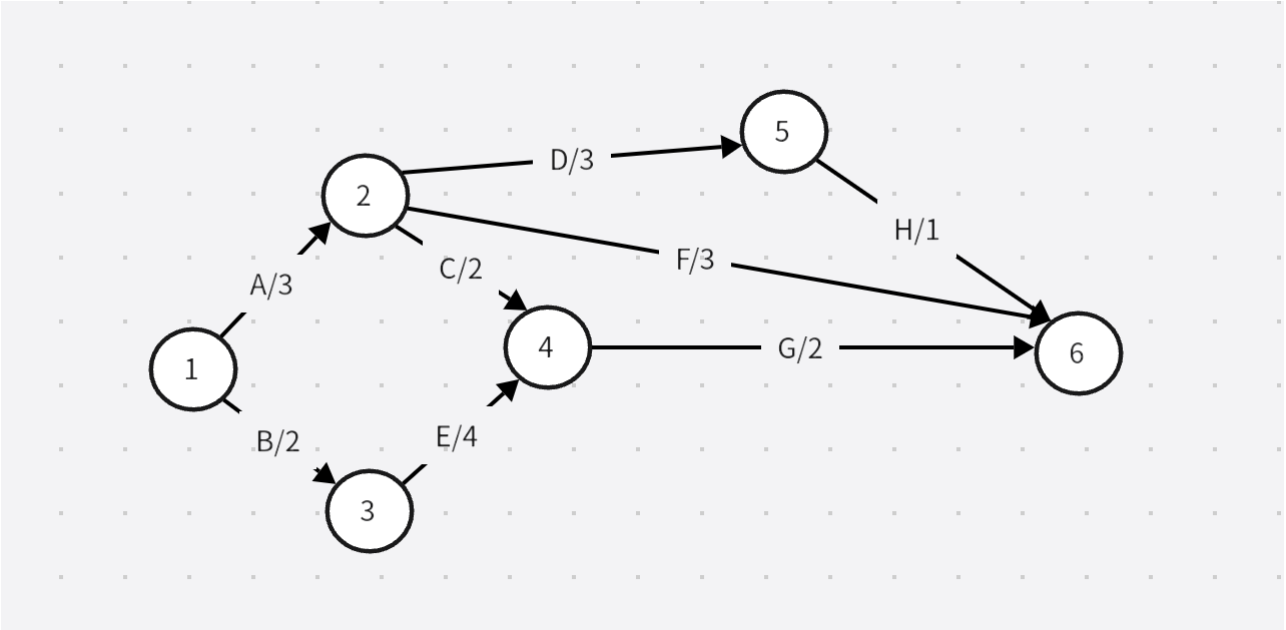
（1）根据表格画出AOE网

（2）列出各事件的最早发生时间和最迟发生时间

（3）求出关键路径并指明完成该工程所需的最短时间

工序代号	A	B	C	D	E	F	G	H
所需时间	3	2	2	3	4	3	2	1
先驱工序			A	A	B	A	C、E	D

2、依次把结点（34,23,15,98,115,28,107）插入初始状态为空的平衡二叉排序树，使得在每次插入后保持该树仍然是平衡二叉树，请依次画出每次插入后所形成的平衡二叉树。



各事件的最早发生时间和最迟发生事件

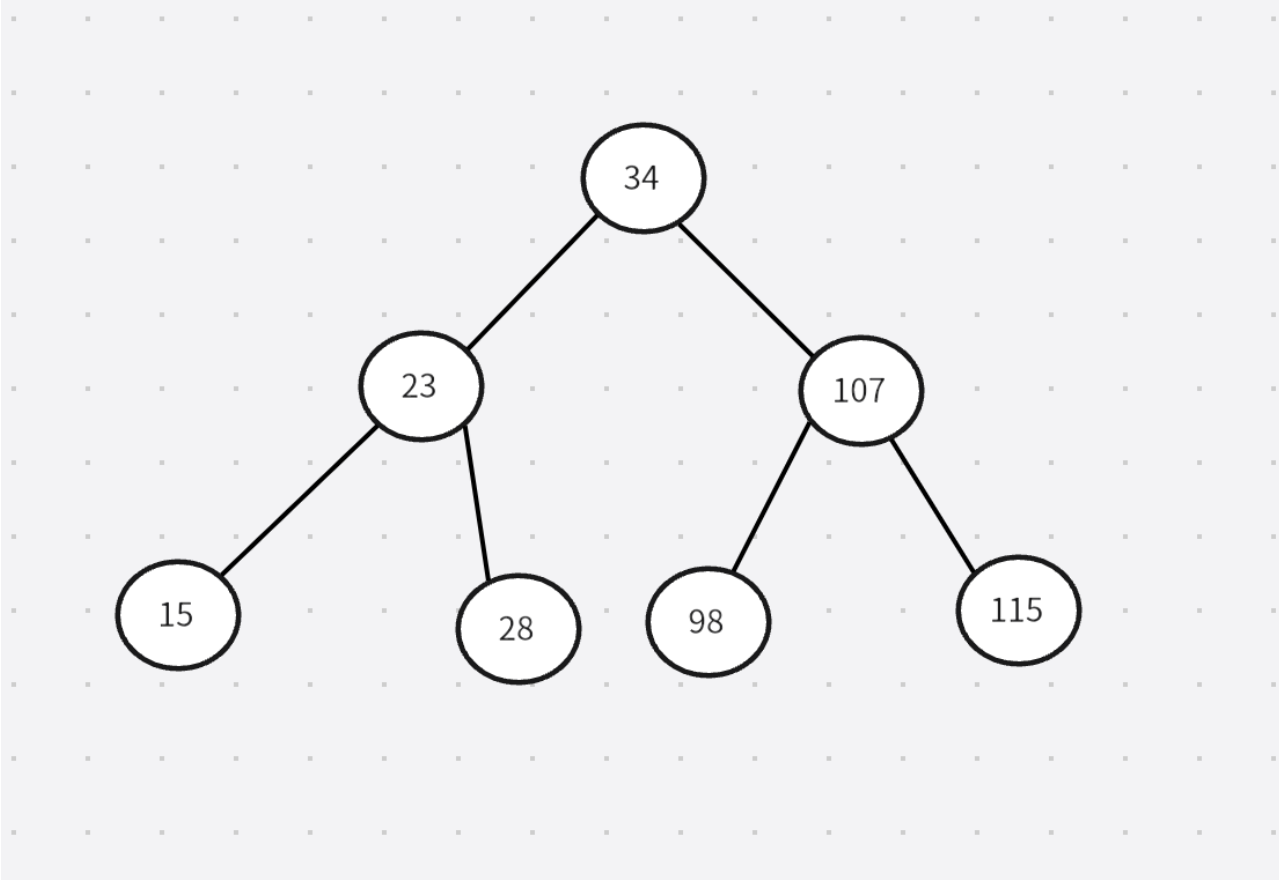
	1	2	3	4	5	6
Ve(i)	0	3	2	6	6	8
Vi(i)	0	4	2	6	7	8

由此，各活动的最早发生时间和最迟发生时间

	A	B	C	D	E	F	G	H
e(i)	0	0	3	3	2	3	6	6
l(i)	1	0	4	4	2	5	6	7
l(i)-e(i)	1	0	1	1	0	2	0	1

因此，关键路径为B,E,G，至少要8

补充二



排序：

排序算法复杂度及稳定性分析

排序方法		时间复杂度			空间复杂度	稳定性
		平均情况	最好情况	最坏情况		
插入排序	直接插入	$O(n^2)$	$O(n)$	$O(n^2)$	$O(1)$	稳定
	Shell排序	$O(n^{1\sim2})$	$O(n\log^2 n)$	$O(n^2)$	$O(1)$	不稳定
选择排序	直接选择	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$	不稳定
	堆排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(1)$	不稳定
交换排序	冒泡排序	$O(n^2)$	$O(n)$	$O(n^2)$	$O(1)$	稳定
	快速排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n^2)$	$O(\log_2 n)$	不稳定
归并排序		$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n)$	稳定
基数排序		$O(d(n+r))$	$O(d(n+r))$	$O(d(n+r))$	$O(n+rd)$	稳定
注：1.希尔排序的时间复杂度和增量的选择有关。 2.基数排序的复杂度中，r代表关键字的基数，d代表长度，n表示关键字的个数。						

建议：

要仔细阅读教材，牢记各种概念、定义，各种数据结构的基本操作及其时间复杂度，各种算法的内容及其时间复杂度以及书上提到过的其他内容。往年考过相当基础的内容，比如：处理哈希冲突的几种方法；逻辑上可以把数据结构分成哪几类等等内容。算法的运行题务必要重视，比如：图算法、哈希、平衡二叉树、哈夫曼树、堆、栈、队列等等，这种题分值的占比起码在一半以上。本课程内容较多，考试虽然肯定有重点（线性表，树，图，搜索），但整体上看考察内容非常多，非常繁杂，因此需要一定的功夫才能得高分。

关于练习，可以参考王道的考研数据结构复习书，上面有比较丰富的习题包括考研真题。真题质量比较高，而且很经典，值得一做。其他题目可以适当选择部分大题当做练习，选择题视情况选做即可，看书背书可能更重要。算法题的练习可以在LeetCode平台上，选择相关数据结构的习题练习，洛谷等算法竞赛的平台题目相对综合，难度更大一些，如果仅仅为了备考本课程考试，是没有必要做的。算法题的重点一般在线性表（顺序表、链表）及其应用（栈、队列）和树，几乎年年都考，但图也需要掌握。