**2022春数据结构（数学班）期中作业**

**2050035 徐恒**

**一、简答题**

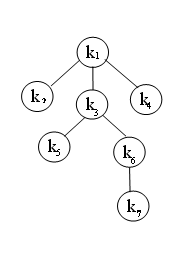
1. 简述数据的逻辑结构和存储结构。

答：逻辑结构是数据与数据之间的联系。存储结构是数据结构在计算机中的存储表示，它包括数据元素的表示和关系的表示。

2. 简述顺序存储结构和链式存储结构的特点和适用场景。

答：顺序存储结构在计算机内存中用一组地址连续的存储单元依次存储线性表中的各个数据元素，链式存储结构的内存地址不一定是连续的。链式存储适用于在较频繁地插入、删除、更新元素时，而顺序存储结构适用于频繁查询时使用。

1. 有一棵树如下图所示，回答下面的问题：

  
⑴ 这棵树的根结点是\_ k1 \_；

⑵ 这棵树的叶子结点是\_\_ k2 k4 k5 k7 \_\_；

⑶ 结点k3的度是\_\_ 2 \_\_；

⑷ 这棵树的度是\_\_ 3 \_\_；

⑸ 这棵树的深度是\_\_ 4 \_\_；

⑹ 结点k3的子女是\_\_ k5 k6 \_\_；

1. 结点k3的父结点是\_\_ k1 \_\_；

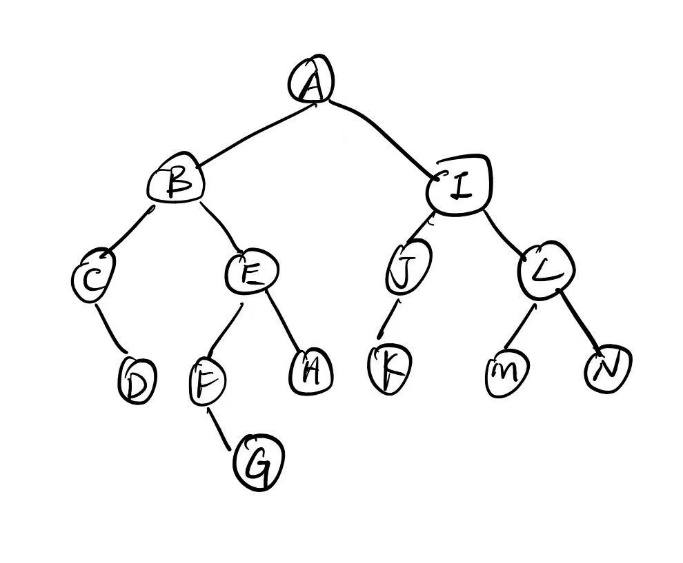
4. 已知二叉树的

先序遍历序列是A B C D E F G H I J K L M N

中序遍历序列是 C D B F G E H A K J I M L N

请画出该二叉树，并给出它的后序遍历序列。

答：二叉树如图



后序遍历：D C G F H E B K J M N L I A

**二、代码题（给出代码描述）**

1. 写出二叉树找后序遍历第一个结点并打印输出的非递归算法。

答：

BiTree P=T;

while (P->lchild | | P->rchild){

if（P->lchild）P=P->lchild;

else P=P->rchild;

}

printf(P->data);

2. 在顺序有序表La中插入某个元素x，要求元素x插入以后依然保持顺序表的有序性，假设表中元素为非减序。

答：

Status ListInsert\_sq(SqList &La, ElemType x){

for(int i=0;i<La.length&&La.elem[i]<x;i++) ;

newbase=(ElemType\*)realloc(La.elem,

(La.listsize+LISTINCREMENT)\*sizeof(ElemType));

if (!newbase) exit(OVERFLOW);

La.elem = newbase;

La.listsize+=LISTINCREMENT;

ElemType \*p,\*q;

q=&(La.elem[i]);

for(p=&(La.elem[La.length-1]);p>=q;--p){\*(p+1) = \*p;

\*q=e;

++L.length;

return OK;

}

**三、应用题**

1.内存中有一片连续的空间，假设地址从1到n，现在这片空间将提供给两个栈S1和S2使用，如何把这块存储空间分配给两个栈，使得对于任何一个栈，仅当这片空间全满时才会发生上溢？并给出相应的栈满栈空条件。

答：S1.base=1；S1.top初始值指向1，进栈时++S1.top；

S2.base=n；S2.top初始值指向n，进栈时--S2.top；

判断S1栈空：S1.base=S1.top；

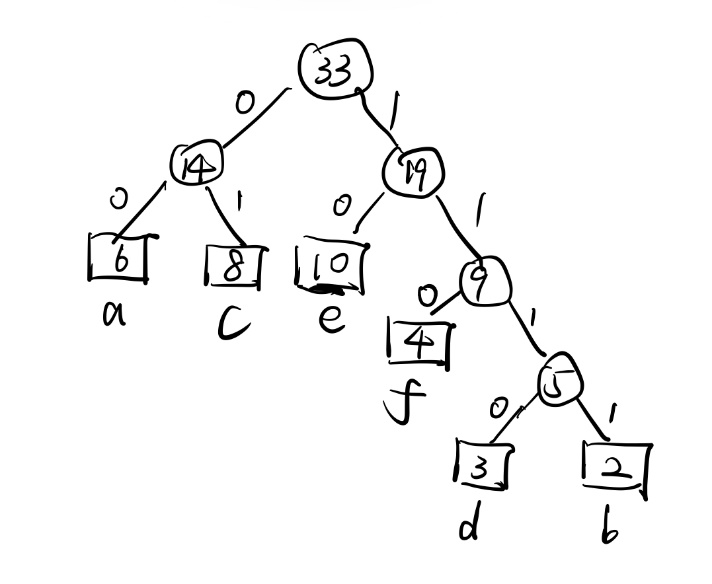
判断S1栈满：S1.top=S2.top+1；

判断S2栈空：S2.base=S2.top；

判断S2栈满：S2.top=S1.top+1；

2. 已知字符集{a,b,c,d,e,f}，若各字符出现的次数分别是{6,3,8,2,10,4}，请构建赫夫曼树并给出各字符对应的赫夫曼编码。

答：



赫夫曼编码 a:00; b:1111; c:01; d:1110; e:10; f:110;

3. 已知两个长度分别为m 和n 的升序单链表，请将它们合并为一个长度为m+n 的降序单链表，并给出时间复杂度。（写出所需的数据结构及合并函数）

答：

数据结构：

typedef struct LNode{

ElemType data;

Struct LNode \*next;

} LNode, \*LinkList;

合并函数：

Typedef struct

void MergeList\_L(LinkList La,LinkList Lb,LinkList &Lc)

{

LinkList pa,pb,pc;

pa = La->next; pb= Lb->next;

Lc = pc = La;

while (pa || pb) {

p=(LinkList)malloc(sizeof(LNode));

if (pa->data >= pb->data || pa){

p->next=pc; p=pa; pa=pa->next;}

else if (pa->data < pb->data || pb){

p->next=pb;p=pb;pb=pb->next;}

}

pc->next=pa?pa:pb; free(Lb);

}

时间复杂度：O(Max(m,n))