**计算机2001级数据结构试题参考答案**

一．（10分，每题1分）

1．C 2．D 3．A 4．B 5．B

6．D 7．D 8．A 9．B 10。C

二．（20分，每题2分）

1. 先（前）序 、中序
2. Q->next->prior=s
3. 6 1
4. EF-G\*ABC-D\*/+
5. 1140
6. MID – 1 ,R[MID].KEY==K或l>h
7. 第I列非零元素之和（∑aji）、第I行非零元素之和（∑aij）
8. Ls==NULL ls=ls->link或ls=p->link
9. Lq->front==lq->rear
10. 动态查找表

三、应用题（共40分）

1．（4分）

先：A BCD EF GH

中：CB DAE GHF

后：C DBHG FEA

2．（6分）其连通分量为：

其广度优先搜索序列为：

（1）V2

（2）V1，V4，V3，V5

1. （6分）其哈夫曼树为：

2（C）：00000

3（F）：00001

6（D）：0001

7（A）：0010

10（H）：0011

32（E）：01

19（B）：10

21（G）：11

所以其带权路径长为：

wpl=2\*5+3\*5+6\*4+7\*4+10\*4+

32\*2+19\*2+21\*2=261

权值为10的结点层次为5

4．（6分）

设该树有n个结点，树有 n0个结点。

则 n=n0+n1+n2+….. +nm (1)

又因为树中的结点数等于其分支数加1。

所以，n-1=n1+2n2+3n3+…+mnm (2)

由（1）、（2）得： n0=n2+2n3+3n4+…(m-1)nm

5．（6分）最小生成树为：

1

5

3 2

4

6．（6分）

7．（6分）

push(‘A’)、push(5)、push(‘B’)、pop(‘A’)、pop(5)、pop(‘B’) 得：B 5 A

push(‘A’)、pop(‘A’)、push(5)、pop(5)、push(‘B’)、pop(‘B’) 得：A 5 B

push(‘A’)、pop(‘A’)、push(5)、push(‘B’)、pop(‘B’)、pop(5) 得：A B 5

四．（20分）

1．（8分）

假设此单链表是带头结点的

typedef struct node{

datatype data;

struct node \* next;

}node, \*link;

int find\_same(link head, datatype k)

{ link p;

int x;

p=head->next;

while(p!=NULL)

{ if(p->data==k)x++;

p=p->next;

}

return x;

}

2．（6分）

typedef struct node {

datatype data;

struct node \* **lchild,\*** rchild;

}node , \* bitreptr;

int count\_leaf(bitreptr root)

{ int x, top;

bitreptr s[100];

bitreptr p;

top=0; x=0;

top++;

s[top]=root;

do{

p=s[top];top--;

if(p->lchild==NULL&& p->rchild==NULL)

x++;

else{

if(p->lchild!=NULL){top++; s[top]=p; }

if(p->rchild!=NULL){ top++;s[top]=p; }

}

}while(top>0);

return x;

}

3．（6分）

#define VexNum 20

typedef struct node {

int adjvex;

struct node \*nextarc;

}arcnode;

typedef struct node1{

vertextype data;

arcnode \* firstarc;

} vnode,adjlist[VexNum];

typedef struct {

adjlist vertices;

int vexnum,arcnum;

}graph;

typedef struct{int in;

int out;

}dd;

void qcdrd(graph g, dd d[ ])

{ int I,j; arcnode \*p;

for(I=0;I<g.vexnum;I++)

{d[I].in=0;d[I].out=0;

}

for( I=0;I<g.vexnum;I++)

{ p=g.vertices[I].firstarc;

while(p!=NULL)

{j=p->adjvex;

d[I].out=d[I].out+1;

d[j].in=d[j].in+1;

p=p->nextarc;

} } }