**苏州大学 数据结构 课8卷参考答案（共 6 页）**

院系 专业 \_\_\_\_\_\_\_\_\_\_

一、填空题（每题3分，共30分）

1、 物理存储位置；链域的指针值

2、队尾；队头 ；先进先出

3、SXSSXSXX

4、i\*(i+1)/2+j+1

5、27-1+1=128； 7

6、零个字符；0 ；由一个或多个空格字符组成的串；其包含的空格个数

7、(a,b,c,d)； ( )

8、树的结点数至少为1，二叉树可以为0 树中结点度数没有限制，而二叉树结点的度数最多为2

9、索引表；块

10、v1,v2,v3,v6,v5,v4 ； v1,v2,v5,v4,v3,v6

二、应用题（每题8分，共40分）

1、

A

B E

C D F

G

（4分）

0 A 0

0 B 0 1 E 0

1 C 1 1 D 1 0 F 1

NIL 1 G 1 NIL

（4分）

2、

15 17 18 22 35 51 60 88 93

↑ ↑ ↑

LOW=0 mid=4 high=8

18<35,在左子表中查找 （3分）

15 17 18 22 35 51 60 88 93

↑ ↑ ↑

LOW=0 mid=1 high=3

18〉17,在右子表中查找 （3分）

15 17 18 22 35 51 60 88 93

↑↑ ↑

LOW=2 mid=2 high=3

18=18查找成功

（4分）

3、

86

50 90

23 78 100

59

55 64

（4分）

ASL= (1+2\*2+3\*3+4\*1+5\*2)/9=28/9 （4分）

86

23 90

78 100

59

55 64

（3分）

4、

10

2 23

15 12 14 6

5 3

初态

（2分）

2

3 6

5 12 14 23

10 15

初始堆

（4分）

3

5 6

10 12 14 23

15 2

输出元素2后调整的堆

（4分）

5、

(1) a b c d e f g

a 0 15 2 12 0 0 0

b 0 0 0 0 6 0 0

c 0 0 0 0 8 4 0

d 0 0 0 0 0 0 3

e 0 0 0 0 0 0 9

f 0 0 0 5 0 0 10

g 0 4 0 0 0 0 0

（3分）

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 终点  Dist | b | c | d | e | f | g | S  (终点集) |
| K=1 | 15  (a,b) | **2**  **(a,c)** | 12  (a,d) |  |  |  | {a,c} |
| K=2 | 15  (a,b) |  | 12  (a,d) | 10  (a,c,e) | **6**  **(a,c,f)** |  | {a,c,f} |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| K=3 | 15  (a,b) |  | 11  (a,c,f,d) | **10**  **(a,c,e)** |  | 16  (a,c,f,g) | {a,c,f,e} |
| K=4 | 15  (a,b) |  | **11**  **(a,c,f,d)** |  |  | 16  (a,c,f,g) | {a,c,f,e,d} |
| K=5 | 15  (a,b) |  |  |  |  | **14**  **(a,c,f,g)** | {a,c,f,e,d,g} |
| K=6 | **15**  **(a,b)** |  |  |  |  |  | {a,c,f,e,d,g,b} |

(2) 从源点a到其它各顶点的最短路径的求解过程如上：

（5分）

1. 算法设计题（每题10分，共30分）

1、

可引入一个指针q，当q->next=p时，说明q所指结点为p所指结点的前趋结点，算法如下：

delete(p) /\*在链表中删除p所指结点的前趋结点\*/

linklist \*p；

{linklist \*q；

if（p->next = =p）

p=Null；

else （3分）

{ q=p；

while（q->next->next！=p）

q=q->next； （4分）

q->next=p；

free（q->next）；

}

} /\*delete\*/ （3分）

2、

Status EnCyQueue(CyQueue &Q,int x)//带length域的循环队列入队算法  
{  
  if(Q.length==MAXSIZE) return OVERFLOW;  
  Q.rear=(Q.rear+1) % MAXSIZE; （3分）  
  Q.base[Q.rear]=x; （3分）  
  Q.length++;  
  return OK;  
}//EnCyQueue

Status DeCyQueue(CyQueue &Q,int &x)//带length域的循环队列出队算法  
{  
  if(Q.length==0) return INFEASIBLE;  
  head=(Q.rear-Q.length+1) % MAXSIZE; //详见书后注释 （2分）  
  x=Q.base[head];  
  Q.length--; （2分）  
}//DeCyQueue

3、

char Pred,Ind; //假设前序序列和中序序列已经分别储存在数组Pre和In中

Bitree Build\_Sub(int Pre\_Start,int Pre\_End,int In\_Start,int In\_End)//由子树的前序和中序序列建立其二叉链表  
{  
  sroot=(BTNode\*)malloc(sizeof(BTNode)); //建根 （3分）  
  sroot->data=Pre[Pre\_Start];  
  for(i=In\_Start;In[i]!=sroot->data;i++); //在中序序列中查找子树根  
  leftlen=i-In\_Start;  
  rightlen=In\_End-i; //计算左右子树的大小 （2分）  
  if(leftlen)  
  {  
    lroot=Build\_Sub(Pre\_Start+1,Pre\_Start+leftlen,In\_Start,In\_Start+leftlen-1);  
    sroot->lchild=lroot; （2分）  
  } //建左子树,注意参数表的计算  
  if(rightlen)  
  {  
    rroot=Build\_Sub(Pre\_End-rightlen+1,Pre\_End,In\_End-rightlen+1,In\_End);  
    sroot->rchild=rroot;  
  } //建右子树,注意参数表的计算  
  return sroot; //返回子树根 （2分）  
}//Build\_Sub

main()  
{  
  ...  
  Build\_Sub(1,n,1,n); //初始调用参数,n为树结点总数 （1分）  
  ...  
}  
分析：本算法利用了这样一个性质,即一棵子树在前序和中序序列中所占的位置总是连续的。因此，就可以用起始下标和终止下标来确定一棵子树Pre\_Start，Pre\_End，In\_Start和In\_End分别指示子树在前序子序列里的起始下标，终止下标和在中序子序列里的起始和终止下标。