复旦大学计算机科学技术学院

《数据结构》期末考试答案及评分标准

A 卷　共 7 页

课程代码：INFO130051.02-03　　考试形式：☐开卷　☑闭卷　　2010 年 1 月

（本试卷答卷时间为 120 分钟，答案必须写在试卷上，做在草稿纸上无效）

（装订线内不要答题）

一、填空题（20%）

1、 答：$O(n^{1/2})$

2、 答：$O(n^2)$

3、 答：$2^i-1,2^{(i-1)},2^{(i-2)}+1$

4、 答：11

5、 答：n^2-e

6、 答：O(n)

7、 答：14　5　(15?　4?)

二、选择题（10%）

1、 答：d

2、 答：d

3、 答：a

4、 答：d

5、 答：b

三、问答题（35%）

1、 已知一个有向网络的邻接矩阵 A 如下，现在要在该网络中的某个结点建立超市，要求该结点距离其它各结点的最长往返路程最短，在最长往返路程相等的情况下，则选择到其它所有结

点的往返路程总和最短的结点。问应该选择哪个结点，给出解题过程（结点编号为 1-6）。（10 分）

0	2	-	-	-	3
-	0	3	2	-	-
4	-	0	-	4	-
1	-	-	0	1	-
-	1	-	-	0	3
-	-	2	5	-	0

答：先用 floyd 算法算出所有结点之间的最短路径，然后对每个结点 i 算出到其它各结点的往返距离总和 sum(i)，以及 i 到其它结点的最长距离 long(i)，最后选择结点 5。

0	2	-	-	-	3
-	0	3	2	-	-
4	6	0	-	4	7
1	3	-	0	1	4
-	1	-	-	0	3
-	-	2	5	-	0

0	2	5	4	-	3
-	0	3	2	-	-
4	6	0	8	4	7
1	3	6	0	1	4
-	1	4	3	0	3
-	-	2	5	-	0

0	2	5	4	9	3
7	0	3	2	7	10
4	6	0	8	4	7
1	3	6	0	1	4
8	1	4	3	0	3
6	8	2	5	6	0

0	2	5	4	5	3
3	0	3	2	3	6-
4	6	0	8	4	7
1	3	6	0	1	4
4	1	4	3	0	3
6	8	2	5	6	0

0	2	5	4	5	3
3	0	3	2	3	6
4	5	0	7	4	7
1	2	5	0	1	4
4	1	4	3	0	3
6	7	2	5	6	0

0	2	5	4	5	3
3	0	3	2	3	6
4	5	0	7	4	7

1	2	5	0	1	4
4	1	4	3	0	3
6	7	2	5	6	0

Sum(1)=2+5+4+5+3+3+4+1+4+6=37
Sum(2)=3+3+2+3+6+2+5+2+1+7=34
Sum(3)=4+5+7+4+7+5+3+5+4+2=46
Sum(4)=4+2+7+3+5+1+2+5+1+4=34
Sum(5)=5+3+4+1+0+6+4+1+4+3+0+3=34
Sum(6)=3+6+7+4+3+0+6+7+2+5+6=49

Long(1)=9; long(2)=13, long(3)=9, long(4)=12, long(5)=9; long(6)=13

评分标准：总分 10 分，思路正确得 5 分，计算正确得 5 分，其它情况酌情扣分。

2、 在 AVL 树中插入一个新结点后，怎么判断需要调整 AVL 树恢复平衡？怎样判断这种调整可以结束，不再需要进一步的调整？（5 分）

答：从新结点开始向上追溯，看新结点的双亲结点的平衡因子，如果变为 0，不需继续向上调整，如果变为 1，则需要继续向上追溯，如果变为 2，则需要做调整，因为调整后局部高度恢复到原来的值，所以不需要继续调整。

评分标准：叙述正确得 5 分，其它情况酌情扣分。

3、 在 n 个不同元素的顺序表中找出最大和最小值需要至少进行多少次比较？算法的思路是怎样的？（6 分）

答：3/2*n-2
把表分为两部分，分别找出各部分的最大最小值，递归进行。
也可每次向前循环两个元素，对这两个元素比较一次，得到最大最小值，然后与前面得到的元素比较最大值和最小值。

评分标准：叙述正确得 4 分，计算正确得 2 分，其它情况酌情扣分。

4、 对数据元素 1、13、29、18、75、60、43、54、90、46、21、34，采用散列函数 hash(key)=key%13 和线性探测再散列方法构造散列表 HT[13]。（1）请计算出各元素散列地址。（2）求搜索成功和不成功的平均搜索长度。（6 分）

答：各元素的散列地址为：1（1）、13（0）、29（3）、18（5）、75（10）、60（8）、43（4）、54（2）、90（12）、46（7）、21(9)、34(11)
搜索成功的平均搜索长度为：（1*10+2*1+4*1）/12=4/3
搜索不成功的平均搜索长度为：（0+1+2+3+4+5+6+7+8+9+10+11+12）/13=6

评分标准：散列地址计算正确、两种搜索长度计算正确，各得 2 分，其它情况酌情扣分。

5、 若存于顺序存储结构中的 n 个元素的序列近似有序，即去掉其中少数 k 个元素的序列是有序序列，试对这种数据文件讨论用直接插入排序法、起泡排序法和直接选择排序法的渐进时间复杂度。（8 分）

答：对于基本有序的 n 个元素，用直接插入排序法其渐进时间复杂度为 O(k*n)，其中 k 为排列无序的元素个数，n 为排列有序的元素个数。显然，用直接插入法时，对有序元素的插入通常只需要比较 1 次即可确定其位置，而对无序元素的插入至多需要比较 n 次；也即，该方法的渐进时间复杂度为 O(k*n)。在用冒泡排序法、直接选择排序法对基本有序的 n 个元素排序时，其渐进时间复杂度都为 O(n2)。

评分标准：总分 8 分，各种简单内排序方法分析比较正确或者部分正确为 1-8 分，否则为 0 分。

四、算法题（35%）

（除写出相应算法之外，还需要说明算法和数据结构的设计思路，算法用 c 或者 C++语法描述）

1、 设 T 是一棵满二叉树，写一个递归算法，把 T 的前序遍历序列转换成后序遍历序列。（10 分）

答：由于是满二叉树，结点总数为 2n+1，先序的第一个结点为根结点，第 2 到 n+1 为左子树的结点，第 n+2 到 2n+1 为右子树的结点。假设已经有了一个先序数组 pre[a1,…,b1]，求后序数组 post[a2,…,b2]的算法如下：

```
convert(int[] pre, int[] post, int preStart, int postStart, int k) {
    post[postStart+k] = pre[preStart];
    if(k==0) return;
    convert(pre, post, preStart+1, postStart+k/2+1,k/2);
    convert(pre, post, preStart+k/2+1, postStart,k/2);
    return;
}
```

评分标准：总共 10 分。

思路 3 分——思路清晰正确为 3 分，方法不限于参考答案这一种，思路不完全正确最多为 1 分，没写为 0 分；

程序 7 分——程序结构基本正确为 3 分，部分正确为 5 分，全部正确为 7 分。

注：由于算法题可能会存在多种求解方法，因此评分过程可依据实际情况进行动态调整。

2、 设 A、B 是长度为 n 的两个非降序数组。如果将这 2n 个数全体排序，处于位置 n 的数称为中位数。设计一个最坏情况下 O(logn)的算法，求这个中位数。并且证明你的算法的时间复杂性。（15 分）

答：可以递归地完成。首先对于两个长度为 k 的有序表，可以比较它们各自的中位数 A[k/2]和 B[k/2]，比较结果分为 3 种情况：

（1） 如果 A[k/2]>B[k/2]，则 A[k/2]..A[k]都大于 B[0]..B[k/2-1]，再加上 B[k/2]..B[k]本来就都大于 B[0]..B[k/2-]，多于 k 个数比 B[0]..B[k/2-1]大，所以，第 k 大的数不可能在 B[0]..B[k/2-1]中，因此可以将 B[0]..B[k/2-1]这部分丢掉；类似地，第 k 大的数也不可能在 A[k/2+1]..A[k]中，所以也可以将这部分丢掉，因此得到了两个新的序列：A[0]..A[k/2]和 B[k/2]..B[k]，然后继续递归。

（2） 如果 A[k/2]<B[k/2]，这种情况和第（1）种情况类似，只不过丢掉的是 A[0]..A[k/2-1]和 B[k/2+1]..B[k]，留下的是 A[k/2]..A[k]和 B[0]..B[k/2]。

（3） 如果 A[k/2]==B[k/2]，因为 A[k/2]大于 A[0]..A[k/2-1]和 B[0]..B[k/2-1]，并且 A[k/2]==B[k/2]，所以第 k 大的数就是 A[k/2]。

因为每次比较都能减少一半的元素，所以最坏情况下递归深度为 logn,所以算法的时间复杂度为 logn。

算法如下，假设 A 和 B 为整型数组：

```
int findMid(int[] a, int[] b，int startA, int startB, int k) {
   if(a[startA+k/2] == b[startB+k/2]) return a[startA+k/2];
   if(a[startA+k/2]>b[startB+k/2]) findMid(a,b,startA+k/2,startB,k/2);
   else (a[startA+k/2]<b[startB+k/2]) findMid(a,b,startA,startB+k/2,k/2);
}
```

评分标准：总共 15 分。

思路 5 分——思路清晰正确为 5 分，方法不限于参考答案这一种，思路不完全正确最多为 2 分，没写为 0 分；

程序 10 分——程序结构基本正确为 5 分，部分正确为 7 分，全部正确为 10 分。

注：由于算法题可能会存在多种求解方法，因此评分过程可依据实际情况进行动态调整。

3、 设计一个算法，判断一个有向无环图中是否存在这样的顶点，该顶点到其它任意顶点都有一条有向路。（10 分）

答：如果存在这样的顶点，由于是有向无环图，则该顶点到其它顶点的路径形成一棵树，树上的结点数量等于图上的结点数量。因此依次从每个结点开始，用深度优先或者广度优先，如果遍历到的结点数量等于图上的结点数量，则存在，否则不存在。

如果用广度优先，图用邻接表表示，其数据结构描述如下：

```
struct node {
   int adjvex;
   node *next;
}

struct g {
   vertexttype data;
   node *firstarc;
}typedef ADJLIST;

int rbfs(ADJLIST[] g, int v, int n) {
```

```
  int i, count, yes;
  yes = 0;
  count = 1;
  Queue Q;
  for(i=0; i<n; i++) visited[i] = 0;
  ENQUEUE(v, Q);
  visited[v] = 1;

  while(!EMPTY(Q)&&!yes) {
      w=Q.front;
      p=g[w].firstarc;
    while((p!=NULL) &&!yes) {
      w=p->adjvex;
      if(visited[w] = 0) {
        visited[w] = 1;
        count++;
        ENQUEUE(w,Q);
      }
      else p=p->next;
    }
  }
  if(count == n) yes=1;
  return yes;
}

Bool   mainLoop(ADJLIST[] g, int n) {
  for(int i=0;i<n;i++)   if( rbfs(g, i, n) == 0) return true;
  return false;
}
```

评分标准：总共 10 分。

思路 2 分——思路清晰正确为 2 分，方法不限于参考答案这一种，思路不完全正确最多为 1 分，没写为 0 分；

关键数据结构 2 分——如邻接表中顶点结点与边结点；

程序 6 分——程序结构基本正确为 2 分，部分正确为 4 分，全部正确为 6 分。

注：由于算法题可能会存在多种求解方法，因此评分过程可依据实际情况进行动态调整。