中南大学考试试卷

参考答案与评分标准

2020--2021 学年 2 学期 时间 100 分钟 2021 年 7 月 2 日

数据结构 课程 56 学时 3.5 学分 考试形式：闭 卷

专业年级：计算机信息类 20 级 总分 100 分， 占总评成绩 60 %

注：此页不作答题纸，请将答案写在答题纸上

一、填空题：（每题 1 分，共 10 分）

1. **试确定下列各程序段中前置以记号@的语句的频度表达式为_____。**

```
for(i=1; i<=n; i++)
        for(j=1; j<=i; j++)
            for(k=1; k<=j; k++)
                @ x += 1;
```

$$1+(1+2)+\cdots(1+2+\cdots+n)$$
$$=\sum_{i=1}^{n}\frac{i(i+1)}{2}=\frac{1}{6}n(n+1)(n+2)$$

2. **数组Q［n］用来表示一个循环队列，f 为当前队列头元素的前一位置，r 为队尾元素的位置，假定队列中元素的个数小于 n，计算队列中元素的公式为 (n + r - f) % n 。**

3. **一个广义表为 D=(a,b,D)，则该广义表的长度为 3 。**

4. **假设有二维数组 $A_{6\times8}$，每个元素用相邻的 6 个字节存储，存储器按字节编址，已知 A 的起始存储位置（基地址）为 1000，按列存储时，元素 A_{47} 的第一个字节的地址为 1276 。**

5. **GetHead【GetTail【GetHead【((a, b), (c, d))】】】= b .**

6. **设一棵完全二叉树有 700 个结点，则共有 350 个叶子结点。**

7. **用 5 个权值{3, 2, 4, 5, 1}构造的哈夫曼（Huffman）树的带权路径长度是 33 。**

8. **设有一稀疏图 *G*，则 *G* 采用__邻接表__存储较省空间。**

9. **已知散列表的地址空间为 A[0..11]，散列函数 H（k）=kmod11，采用线性探测法处理冲突。请问在下列数据{25,16,38,47,79,82,51,39,89,151,231}中查找元素 82 时，需要比较__3__次。**

10. **设要将序列（Q, H, C, Y, P, A, M, S, R, D, F, X）中的关键码按字母序的升序重新排列，则：快速排序一趟扫描的结果是__F H C D P A M Q R S Y X__。**

二、选择题：（每题 2 分，共 20 分）

1. **在表长为 n 的顺序表中，算法的时间复杂度为 O(1)的操作是_____。**

A．在第 n 个结点之后插入一个结点　　B．在第 i 个结点前插入一个新结点

C．删除第 i 个结点　　D．求表长

2. 已知 L 是无表头结点的单链表，且 P 结点既不是首元结点，也不是尾元结点，在 P 结点后插入 S 结点的语句序列是：_____

A. P ->next= S ->next;　S->next= P;　B. S->next=P->next; P->next=S;

C. P->next=S; S->next=P->next;　　D. S->next= P; P ->next= S ->next;

3. 双向循环链表的每个结点中包括两个指针 next 和 previous，分别指向该结点的后继和前驱结点。现要删除指针 p 所指向的结点，下面的操作序列中_____是正确的？

A. *p*->next-〉previous = *p*->previous; *p*->previous-〉next = *p*->next；

B. *p*->next-〉previous = *p*->next; *p*->previous-〉next = *p*->previous；

C. *p*->previous-〉next = *p*->previous; *p*->next-〉previous = *p*->next；

D. *p*->priou-〉next-〉next = *p*->next; *p*->next-〉previous = *p*->previous；

4. **若已知一个栈的入栈序列是 1，2，3，…，*n*，其输出序列为 p_1，p_2，p_3，…，p_n，若 $p_1=n$，则 p_i 为：_____**

A.　*i*　　B.　*n-i*　　C.　*n-i*+1　　D.　不确定

5. **若用一个大小为 6 的数组来实现循环队列，且当前 front 和 rear 的值分别为 3 和 0，当**

从队列中删除一个元素，再加入两个元素后，front 和 rear 的值分别为_____。

A．5 和 1　　B．4 和 2　　C．2 和 4　　D．1 和 5

6. **若 REPLACE（S，S1，S2）表示用字符串 S2 替换字符串 S 中的子串 S1 的操作，则对于 S="Beijing＆Nanjing"，S1="Beijing"，S2="Shanghai"，REPLACE（S，S1，S2）=_____。**

A. "Nanjing＆Shanghai"　　B. "Nanjing＆Nanjing"

C. "ShanghaiNanjing"　　D. "Shanghai＆Nanjing"

7. 设哈希表长度为 11，哈希函数为 H（key）=key mod 11。表中已有 4 个元素 H（15）=4；H（38）=5；H（61）=6；H（84）=7 其余地址为空，若用二次探测再散列处理冲突，关键字为 49 的元素的地址是_____。

A. 3　　B. 5　　C. 8　　D. 9

8. 在一个有向图中，所有顶点的入度之和等于所有顶点的出度之和的 _____倍。

A．1/2　　B．1　　C．2　　D．4

9. 将 5 个不同的数据进行排序，至多需要比较_____次。

A．8　　B．9　　C．10　　D．25

10. 若一组记录的排序码为（46，79，56，38，40，84），则利用堆排序的方法建立的初始堆为_____.

A．79, 46, 56, 38, 40, 84　　B．84, 79, 56, 38, 40, 46

C．84, 79, 56, 46, 40, 38　　D．84, 56, 79, 40, 46, 38

三．计算题（共 5 小题，每小题 8 分，共 40 分）

1. **利用两个栈 sl,s2 模拟一个队列时，如何用栈的运算实现队列的插入，删除以及判队空运算。请简述这些运算的算法思想。**

解答：

使用两个栈，分别依元素加入的顺序和其反序保存元素，在适当的时机将元素在两个栈中进行转移，从而模拟队列的操作。3 分

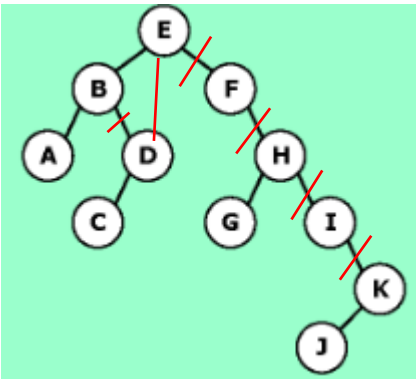
令 S1 中元素的顺序为自底向上与元素添加顺序一致，S2 与其相反，则：加入队列时，若 S2 不空，则将 S2 中的元素依次出栈，每出栈一个向 S1 中入栈一个；将入队元素入 S1 栈；

从队列中取出时，若 S1 不空，则将 S1 中元素依次出栈，每出栈一个向 S2 中入栈一个；从 S2 栈顶出栈一个即队列中取出的元素。5 分

2. **假设一棵二叉树的先序序列为 EBADCFHGIKJ 和中序序列为 ABCDEFGHIJK。请**

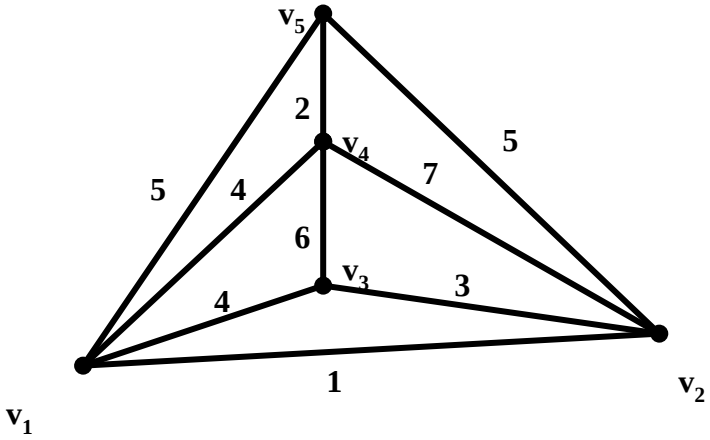
（1）画出该树。6 分

（2）在该二叉树对应的森林中有几棵树。2 分



该二叉树对应的森林中有 5 棵树

3. 假设要在某地建造 5 个工厂，拟修筑道路连接这 5 处。经勘探，其道路可按下图的无向边铺设。现在每条边的长度已经测出并标记在图的对应边上，如果我们要求铺设的道路总长度最短，这样既能节省费用 ，又能缩短工期 。请你为该地设计一个方案，并按方案给出每步迭代的结果（假设开始顶点就选为顶点 V1）。



解答：基于图求解最小生成树。3 分

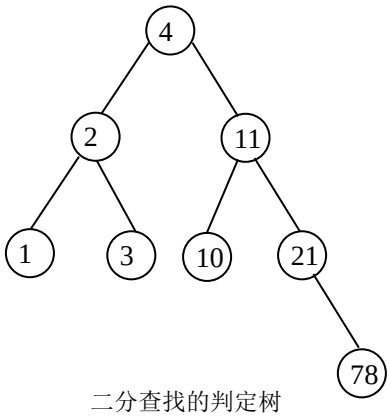
比如使用 Prim 算法求解过程如下：5 分

	V2	V3	V4	V5	U	V-U	k
adiver lowcost	V1 1	V1 4	V1 4	V1 5	{v1}	{v2,v3,v4,v5}	V2
adiver lowcost	 0	V2 3	V1 4	V1 5	{v1,v2}	{v3,v4,v5}	V3
adiver lowcost	 0	 0	V1 4	V1 5	{v1,v2,v3}	{v4,v5}	V4
adiver lowcost	 0	 0	 0	V4 2	{v1,v2,v3,v4}	{v5}	V5
adiver lowcost	 0	 0	 0	 0	{V1,V2,V3,V4,V5}	{ }	

4. 给定关键字序列 11，78，10，1，3，2，4，21，试分别用二分查找、二叉排序树查找来实现查找，试画出它们的对应存储形式(二分查找的判定树，二叉排序树)，并求出每一种查找的成功平均查找长度。

解答：

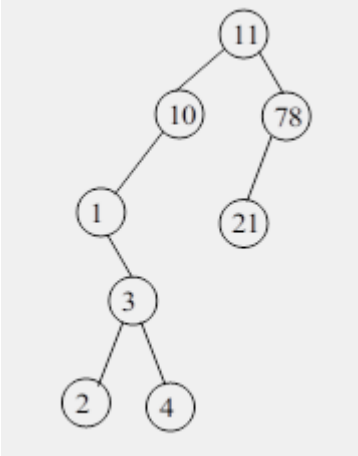
二分查找的判定树（中序序列为从小到大排列的有序序列）3+1 分



二分查找的判定树

二分查找的成功平均查找长度为：ASL=(1+2*2+3*4+4)/8=2.625；

二叉排序树（关键字顺序已确定，该二叉排序树应唯一）3+1 分



从图可以得到二叉排序树查找的成功平均查找长度为：

ASL=(1+2*2+3*2+4+5*2)/8=3.125；

5. 算法填空题 8 分

如下为 Dijkstra 算法的部分代码，试将其填写完整。

```
1.      #define SIZE 110
2.      #define INF 1000000
3.      int arcs[SIZE][SIZE];  //邻接矩阵存储
```

```
4.      int Dist[SIZE];     //dist[i]表示源点到 i 这个点的距离
5.      int visit[SIZE];  //节点是否被访问
6.      int NumOfVertices ;
7.
8.      void Dijkstra(int from ){
9.
10.         int i,j;
11.
12.         for(i = 1 ; i <= NumOfVertices ; i ++){  //初始化
13.             visit[i] = 0;   //一开始每个点都没被访问
14.             Dist[i] = arcs[from][i];   //先假设源点到其他点的距离
15.         }
16.
17.         for(i = 1 ; i < NumOfVertices ; ++i){ //对除源点的每一个点计算
18.             int min = INF;  //记录最小 Dist[i]
19.             int pos;  //记录最小 Dist[i] 的点
20.
21.             for(j = 1 ; j <= NumOfVertices ; ++j){
22.                 if(!visit[j] && min > Dist[j]){
23.                     pos = j;
24.                     min = Dist[j];
25.                 }
26.             }
27.             visit[pos] = 1;
28.
29.             for(j = 1 ; j <= NumOfVertices ; ++j){
30.                 if(!visit[j] && (Dist[j] > (Dist[pos] +arcs[pos][j]))){
31.                     Dist[j] = Dist[pos] + arcs[pos][j];
32.                 }
33.             }
34.         }
35.
36.     }
```

每空 2 分

四. 算法设计题（共 3 小题，每题 10 分，共 30 分）酌情给分。

1. 已知指针 ha 和 hb 分别指向两个单链表的头结点，试写一算法将这两个链表连接在一起（即令其中一个表的首元结点连在另一个表的最后一个结点之后），假设指针 hc 指向连接后的链表的头结点，并要求算法以尽可能短的时间完成连接运算。请分析你的算法和时间

复杂度。

参考代码：

```
void MergeList_L(LinkList &ha, LinkList &hb, LinkList &hc)//合并链表
{
  LinkList pa, pb;
  pa = ha;
  pb = hb;
  while(pa->next && pb->next)
  {
    pa = pa->next;
    pb = pb->next;
  }
  if(!pa->next)
  {
    hc = ha;
    pa->next = hb->next;
    free hb;
  }
  else
  {
    hc = hb;
    pb->next = ha->next;
    free ha;
  }
}
```

8 分

假设两个表长度分别为 m,n，则时间复杂度：O(min(m,n)) 2 分

2. 假设二叉树采用二叉链存储结构，设计一个算法把一棵含有 n 个节点的二叉树 b 复制到另一棵二叉树 t 中。给出你所设计算法的时间和空间复杂度。

```
typedef struct BTNode {
    char data;
    struct BTNode *left;
    struct BTNode *right;
}BTNode;

//由二叉树 BT 复制产生另一棵二叉树 BT1
void copyBTree(BTNode*BT, BTNode*&BT1) {
    if (BT==NULL) {
        BT1 = NULL;
```

```
    }else{                                          2分
        BT1 = (BTNode*)malloc(sizeof(BTNode));
        BT1->data = BT->data;
        copyBTree(BT->left, BT1->left);
        copyBTree(BT->right, BT1->right);
    }                                               8分
```

3. 编写算法求出连通图中一个广度优先生成树。

参考代码：类型定义可不写

```
#define    MAXV 100//最大顶点个数

int visited[MAXV];//全局数组

typedef struct ANode
{    int adjvex;            //该弧的终点位置

     struct ANode *nextarc; //指向下一条弧的指针

     InfoType info;         //可选。该弧的相关信息,这里用于存放权值 n

} ArcNode;//弧的结点结构类型

typedef struct Vnode
{   int data;           //顶点信息

    ArcNode *firstarc;//指向第一条弧

} VNode;//邻接表头结点的类型

typedef struct
{
    VNode adjlist[MAXV];//邻接表

    int n,e;//图中顶点数 n 和边数 e

} ALGraph;//图的邻接表类型


void BFS(ALGraph *G,int v)
{
```

```
    int queue[MAXV],front=0,rear=0;     //定义循环队列并初始化
    int visited[MAXV]; //定义存放结点的访问标志的数组
    for (int i=0;i< G->n;i++)  visited[i]=0;     //访问标志数组初始化

    visited[v]=1; //置已访问标记
    queue[rear]=v;//已访问过的顶点 v 进队
    rear=(rear+1)%MAXV;
    while (front!=rear)//若队列不空时循环
    {
        int w=queue[front];//出队并赋给 w
        front=(front+1)%MAXV;
        ArcNode *p=G->adjlist[w].firstarc; //找与顶点 w 邻接的第一个顶点
        while (p!=NULL)
        {
            if (visited[p->adjvex]==0) //若当前邻接顶点未被访问
            {
                printf("<%d,%d> ",w,p->adjvex);//输出生成树的一条边

                visited[p->adjvex]=1;//置该顶点已被访问的标志
                queue[rear]=p->adjvex;//该顶点 p 的终点进队
                rear=(rear+1)%MAXV;
            }
            p=p->nextarc;//找下一个邻接顶点
        }
    }
    printf("\n");
}
```