张老师班

复旦大学计算机科学技术学院

《数据结构》期中考试试卷

共 8 页

课程代码：COMP130004.04, 05　考试形式：□开卷 ■闭卷　2023 年 11 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）

专业＿＿＿＿＿＿学号＿＿＿＿＿＿姓名＿＿＿＿＿＿成绩 80

题号	一	二	三	总分
得分				

-6

一、 填空题（本大题共十个填空，每空 2 分，共 20 分）

1、算法的每一步都应该确切、无歧义地定义。这被称为算法的＿＿＿＿性。

2、一个网站估计用户量可以达到 1 亿，为每个用户分配 1 个位(bit)的空间来记录该用户当天是否登录网站，规定 1 个字节 8 个位，则每天记录 1 亿用户是否登录所需的存储空间为（精确到整数）＿＿＿＿ MB。

3、用数组 q[m...n]表示的顺序存储队列，用 front 表示队头指针，指向队头元素的前一位置，初值为 m-1，当队列发生假溢出时，队列中的元素个数为＿＿＿＿＿＿＿＿.

4、用一维数组 B 按列优先存放三对角矩阵 A 中的非零元素 A[i, j] (1≤i≤n, i-2≤j≤i+2 并且 1≤j≤n)，B 中的第 8 个元素在 A 中的行列号为＿＿＿＿

5、后缀表达式 abcd-*+ef/-对应的中缀表示为＿＿＿＿＿＿＿＿.

6、高度为 h 的满二叉树，根结点的高度为 0，则其结点总数 n 可以用 h 表示为＿＿＿＿＿＿＿＿.

7、一棵度为 2 的普通树用左子女-右兄弟法转为二叉树后，右子树的结点数为＿＿＿＿

8、一棵度为 k 的树上有 n_1 个度为 1 的结点、n_2 个度为 2 的结点、…、n_k 个度为 k 的结点，则树中叶结点的个数为＿＿＿＿＿＿＿＿

9、广义表 A=((a), b, (c, d, (e, f)))的深度是＿＿＿＿，表尾是＿＿＿＿＿＿＿＿

二、问答题（本大题共六小题，共 40 分）

1、请分析教材中利用队列打印杨辉三角形前 n 行的算法空间复杂度。（5 分）

4

答：逐行进队，在打印下一行时前一行退队。

所用空间最多为第 n 行元素个数（+2冗余）。即 n+1+2=n+3（认为每一行为 0,1,0）。

∴空间复杂度为 O(n)

-1

O(n+3) = O(n)

6

2、已知一个算法的运行时间可表示为 T(N)=3*T(N/4)+N，T(0)=1，请推导出 T(N)的大 O 表示法。（6 分）

可以归纳，不能直接找规律

解：令 $N=4^k$.

$\therefore T(4^k) = 3T(4^{k-1}) + 4^k$

$= 3(3T(4^{k-2}) + 4^{k-1}) + 4^k = 3^2 T(4^{k-2}) + 3\times 4^{k-1} + 4^k$

$= 3^3 T(4^{k-3}) + 3^2\times 4^{k-2} + 3\times 4^{k-1} + 4^k$

$= \cdots$

$= 3^{k+1} T(4^0) + 3^k \times 4^1 + 3^{k-1}\times 4^2 + \cdots + 4^k$

$= 3^k T(1) + 3^{k-1}\times 4 + \cdots + 4^k$.

若认为 $T(1) = 3\times T(0) + 1 = 4$.

$\therefore T(4^k) = \sum_{i=0}^{k} 3^i 4^{k-i} + 3^{k+1}$

$\therefore$ O(N).

3、字符串 S1 由某个字符串 S2 不断自我连接形成，但是字符串 S2 未知。给出 S1 的一个长度为 n 的片段 S，问怎样根据 S 的 Next[ ]数组计算出可能的 S2 的最短长度。例如，给出 S1 的一个长度为 8 的片段 S="cabcabca"，则最短的 S2 长度是 3，S2 可能是"abc"、"cab"、"bca"。（8 分）

答：需当 S 至少包含 2 遍 S_2。

则 S 的 Next[] 数组中最大元素，即为最短 S_2 长度。

eg. c a b c a b c a
-1 0 0 1 2 3 1 2

∴ min(len(S_2)) = 3.

c a b c a b c a
-1 0 0 0 1 2 3 4

1) S由完整的k个S_2拼接而成

Next[n-1] = (k-1)×L-1，n=k×L，L=n-Next[n-1]-1

c a b c a b c a b
-1 0 0 0 1 2 3 4 5　　9-6=3

2) S由k个完整的S_2和一个不完整的S_2拼成，设S_2长为L，不完整部分长为Z

Next[n-1] = (k-1)L+Z-1，n=k×L+Z，⇒ L = n-Next[n-1]-1

4、请分析并回答以下问题：（8 分）

(1) 在一棵二叉树的前序遍历序列、中序遍历序列、后序遍历序列和层次序遍历序列中，任意两种序列的组合可以唯一地确定这棵二叉树吗？若不可以，有哪些组合可以。请说明理由。

(2) 已知一个森林的先根次序遍历序列和后根次序遍历序列分别为 ABCDEFGHIJKLMNO 和 CDEBFHIJGAMLONK，请构造出该森林。若写树会扣分!

答：(1) 不可以。

前中&后中：递归，依靠前/后确定根，利用中序分开左、右子树，分别递归。

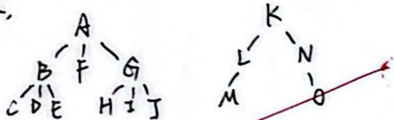
前后不行：eg. 前序同为 ABCD.

前层不行

后层不行

中层可以：层确认根，中区分左右子树。

(2) 第一棵树根为 A，第二棵树根为 K，共二棵树。



(采用直接用树表示，非左孩子右兄弟！)

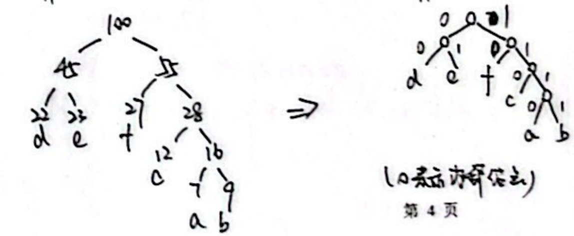
5、一棵二叉树的数组存储表示为[a, b, c, d, #, e, f, #, g, #, #, #, h, #]，其中#表示空结点，请画出这棵二叉树的图形表示，并且加上中序线索。（5 分）

解：

中序遍历：d g b a h e c f.

6、假设字符 a、b、c、d、e、f 的使用概率分别是 0.07、0.09、0.12、0.22、0.23、0.27，画出对应的 Huffman 编码树，计算总编码长度，并给出 a、b、c、d、e、f 对应的 Huffman 编码。（8 分）

解：Huffman 树（权值×100）： Huffman 编码树：



(0表示左子节点)

∴ Huffman 编码：

a	1110
b	1111
c	110
d	00
e	01
f	10

平均编码长度 = WPL
= 0.07×4+0.09×4
+0.12×3+0.22×2
+0.23×2+0.27×2
= 2.44

总编码长度：个数×2.44

三、算法设计题（本大题共四小题，共 40 分）

（以下题目，首先用简明的文字写出算法基本思路，然后给出数据结构和算法的 C 或 C++描述。另，有关堆栈和队列的相关操作如 Push、Pop、GetTop、EnQueue、DelQueue、GetFront 等可直接调用，不需要给出具体函数实现。）

1、给定一个矩阵，其中取值为 0 或者 1，请编写程序求其中最大的全是 1 的矩阵中 1 的个数。例如，输入矩阵[1, 1, 1, 0]，返回 3；输入矩阵{{1, 0, 1, 1}, {1, 1, 1, 1}, {1, 1, 1, 0}}，返回 6。函数声明为 int CountMaxRec(int m[ ][ ], int i, int j)。m 为输入的矩阵，i 为矩阵行数，j 为矩阵列数。分析程序的时间复杂度。（10 分）

-5

解：输入元素为矩阵最左上角，检测从这个1开始的最大全是1矩阵(右下 下一个)。先检测行最大连续1，再检测下一行是否能构成"1"的矩阵，减小最大行长度。int n 记录当前矩阵最大"1"数，int num 记录目前为止的最大"1"数。

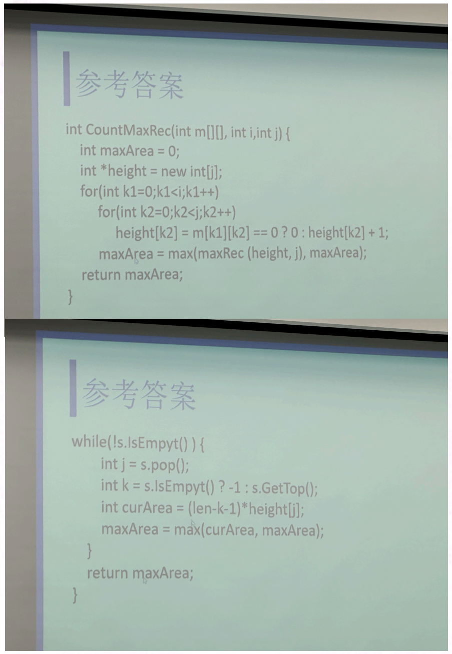
时间复杂度：$O(i^2j^2)$，遍历矩阵，对每个元素，遍历其右&下所有元素。

最坏：矩阵全1.

5

```
int CountMaxRec(int m[][], int i, int j){
    int n=0; int num=0;
    for(int row=0; row<i; ++row){
      for(int col=0; col<j; ++col){ //int n=0, r=0, c=0;
        int i0=row, j0=col; //遍历指针，遍历右下
        while(m[i0][j0]){
            n++; j0++;
        } // while(m[i0][t+j])
        i0++; j0=col; //指向下一行
        while(m[i0][j0] && j0-col<n)  //当前行1下
            j0++;

        }
        c--;
      num= max(c*n, num);
    }
    return num;
};
```



9

2、规定 12 进制数的各位用字符 0-9, A, B 表示，同时规定 15 进制数的各位用字符 0-9, A-E 表示。请编写函数"char *convert(char *x)"根据输入的表示一个 12 进制数的字符串 x，转换为一个 15 进制数的字符串 y 返回。例如，convert("A2")返回的结果为"82"。已知 x 对应的十进制数的大小不会超过 C 语言的 int 类型的范围，n 为 x 的字符串长度，要求算法时间复杂度为 O(n)，空间复杂度为 O(1)。（10 分）

-1

解：先12进制转10进制：读取一个字符，判断为'0'~'9' or 'A'/'B'，num*=12，num+=相应值。(从高位到低位)

再10进制转15进制：考虑到输出为字符串，最大长度为 $\log_{15}(2^{31})<8$. ∴ 用char[]，用个指针指向后一个未赋值的位置，指针不断前移，最后留一个'\0'. 利用长除法，从低位到高位构建输出的15进制数。

```
char* convert(char* x){
    //先转12->10.
    int num = 0;
    while(*x != '\0'){ //在结尾停止处理
        num *=12;
        if(*x == 'A')
            num +=10;
        else if (*x == 'B')
            num +=11;
        else
            num += (*x - '0');
    } //while结束。得到10进制数 num.
    char res[10] = {'\0'}; //初始化字符串，最大：log15 2^31 <8. 10位足够.
    int t=8; // res的指针，指向最后一个还未赋值的字符。最后留一个'\0'
    while (num != 0){
        int digit = num % 15 ;
        if(digit == 10)   res[t--] = 'A';
        else if (digit == 11)  res[t--] = 'B';
        else if(digit == 12)  res[t--] = 'C';
        else if (digit == 13)  res[t--] = 'D';
        else if (digit == 14)  res[t--] = 'E';
        else  res[t--] = '0' + digit;
        num /= 15;
    } //end of while. 10->15 complete.
    return res+t+1;  // eg. {'\0','\0','\0','A','B','C','D','A','B','C'}. t=2. return res+3;
}
```

'A'+(digit-10)

10

3、为二叉树 BTree 编写一个方法"void ancestor(BTreeNode *x)"打印 x 结点在树中的所有祖先的 name 属性值，并分析算法的渐进时间复杂度（不加分析直接写出结果得零分）。（BTreeNode 中有 3 个属性，字符串型属性 name、左孩子指针 left、右孩子指针 right；BTree 中有 1 个属性"BTreeNode *root"指向树根）（10 分）

解：用一个stack存放当前路径（用vector<BTreeNode*>实现，便于打印）.

递归：当前结点若非空则进栈，若左、右孩子有x结点则返回打印，否则递归左子树，再递归右子树，若均搜索失败则当前结点退栈.

时间复杂度：用DFS深度优先搜索，每个结点访问一次，共n个结点，最坏访问n-1次最好访问1次.

$$\therefore \frac{1}{n-1}\sum_{i=1}^{n-1} i = \frac{1}{n-1} \times \frac{n(n-1)}{2} = \frac{n}{2}.$$ 平均访问一半. ∴ O(n).

```
void ancestor(BTreeNode* x) {
    if (x == root) {cout << "根结点无祖先" << endl; return;}.
    ga(root, x, path);  vector<BTreeNode*> path;
    for (int i=0; i<path.size(); i++) {
        cout << path[i]->name << endl; }.
};
bool ga(BTreeNode* r, BTreeNode* x, vector<BTreeNode*> &path) {
// 递归函数，path需要引用传递，返回true表示搜索到x }.
    if (r == NULL) return false;
    path.push_back(r);
    if (r->left == x || r->right == x) return true;
    bool c1, c2; // 记录递归情况.
    c1 = ga(r->left, x, path);
    if (c1) return true;
    c2 = ga(r->right, x, path);
    if (c2) return true;
    path.pop(); // 搜索失败，r不为x的祖先，退栈.
    return false;
}
```


9

4、给定指向二叉树根结点的指针 root，二叉树为二叉链表存储方式（每个结点中有 left 和 right 两个指针指向左右孩子，整型的 data 属性中保存结点关键码），对于包含 n 个结点的二叉树，假设其中各个结点的关键码互相不重复，请编写非递归算法找到结点关键码的最大值，试对算法性能进行分析。（10 分）

-1

解：通过层序遍历访问每一个结点，记录最大data. 依靠队列实现：根结点入队；后循环，队头出队，队头的两个child入队，直到队空，访问结束.

时间复杂度：O(n). (访问每个结点，共n个).

空间复杂度：即为某一层的结点个数最大值. 考虑完全二叉树. 为 $\frac{n+1}{2}$ 个. ∴ O(n)为空间复杂度

```
int getMax(BTreeNode* root) {
    queue<BTreeNode*> q;   int maxData = root->data;
    q.push(root);
    while (!q.empty()) { // 队非空，循环入队.
        BTreeNode* curr = q.top(); // 取队头.
        q.pop(); // 队头出队.
        if (curr->left). q.push(curr->left); // 左子 非空则入队.
        if (curr->right). q.push(curr->right);
        maxData = max(maxData, curr->data);
    } // end of while.
    return maxData;
}
```

int maxData = 0;
if (!root). return maxData;