数据结构与算法
习题课

黄进晟

2022.10.19

■ 答题规范

- 算法设计题答题规范
  - 必要的数据结构定义
  - 算法思想描述
  - 伪代码 / 代码（若题目没有明确要求，则二者均可，要保证逻辑清晰）
  - 复杂度分析（时间复杂度、空间复杂度）

■期中复习题

■ 第一章

1.　　某算法仅含顺序执行的语句1和语句2，语句1执行次数为$20n^2+3n\log n$，而语句2执行次数为$2n^3$，则该算法的时间复杂度为 ________________。

答案：$O(n^3)$

2. 以下函数的时间复杂度为________________。

```
int work(int n, int m) {
    int sum = 0;
    for (int i = 1; i <= m; i *= 2)
        for (int j = 1; j <= n; j *= 2)
            sum += i * j;
    return sum;
}
```

答案：O(log n * log m)

3. 以下函数的时间复杂度为________________。

```
void recursive(int n, int m, int k){
    if (n <= 0)
        printf("%d, %d\n", m, k);
    else {
        recursive(n-1, m+1, k);
        recursive(n-1, m, k+1);
    }
}
```

答案：$T(n)=2*T(n-1)$ $T(0)=1$推出$T(n)=2^n$

4. 以下时间复杂度从小到大的顺序是：

A、$(\log(n))!$

B、$\sqrt{2}^{\log(n)}$

C、$2^{\log(\log n)}$

D、$\log(n!)$

答案：C<B<D<A

■ 第二章

1. 请问在以下操作中，单向链表的时间复杂度高于基于数组的顺序存储结构的是:

A. 合并两个有序线性表使合成后表仍有序

B. 交换第一个元素与第二个元素的值

C. 查找值 x是否在线性表中出现

D. 输出第 i个元素的值

答案： D

2. 如何判断一个单链表是否有环？若有环，如何找出环的入口节点。

思路：

如果单链表有环，当slow指针和fast指针相遇时，slow指针还没有遍历完链表，而fast指针已经在环内循环n（n>=1）圈了，假设此时slow指针走了s步，fast指针走了2s步，r为fast在环内转了一圈的步数，a为从链表头到入口点的步数，b为从入口点到相遇点的步数，c为从相遇点再走c步到达入口点。

参考答案：

slow指针走的步数：

$s = a + b$

fast指针走的步数：

$2s = s + n*r$　　即：$s = n*r$

由上可得：

$a + b = n*r$

$a = (n - 1)*r + r - b$

$a = (n - 1)*r + c$

综上可得：在链表头和相遇点分别设置一个指针，同时出发，每次各走一步，它们必定会相遇，且第一次相遇的点就是环入口点。

■ 第三章

1. 一队列从头至尾分别为 abcabd（每个字母为一元素），现在希望将其内部元素变为 abcabc至少需要进行（）次进队或出队操作。

A. 2

B. 6

C. 8

D. 12

答案：D

- 2. 使有如下命题:设入栈序列为1,2,...,n，出栈序列不合理的充分必要条件是
  - $\exists a,b,c,\ a<b<c \wedge p_c<p_a<p_b$
- 其中p_i表示元素 i 是第几个出栈的。
- 例如，对于出栈序列 1,3,4,2，有$p_1=1$, $p_2=4$, $p_3=2$, $p_4=3$
- (1) 证明命题的充分性
- 答案：(3 分) 假设存在三元组(a,b,c)满足$a<b<c \wedge p_c<p_a<p_b$。由于c比a，b后入栈但比a，b先出栈，所以当 c 出栈时，a，b 仍在栈中;又因 b 比 a 后入栈，所以 b 一定比 a 更靠近栈顶， 因而不可能比 a 后出栈。因此，当此三元组存在时，出栈序列一定不合理。
- 关键：c出栈时，b比a靠近栈顶/栈的后进先出性质/枚举法：根据卡特兰数，三个元素有五种出栈顺序，唯独没有cab（主要看这几个关键句，写出一个基本上就八九不离十了）
- 扣一分的情况：没说明为何c比ab先出栈时，a不可能比b先出栈/说明时把p_a和a搞混/没说明枚举的方式或原则，导致枚举不具备说服力
- 扣光的情况：证明成必要性/证明存在其他原则性错误

- (2)对于下列不合理的出栈序列，找出所有的三元组(a,b,c)并给出对应的(p_a,p_b,p_c)，使之满足$a<b<c \wedge p_c <p_a <p_b$
- a) 1,4,2,3
- b) 2,5,3,4,1
- c) 4,3,6,2,1,5

- 答案：（3分）
- a) (2,3,4), (3,4,2)
- b) (3,4,5), (3,4,2)
- c) (1,5,6), (5,6,3) ; (2,5,6), (4,6,3)

- a)b)只有一种组合，c)有两种组合，abc)每条一分
- 每条多写，错写0分
- 每条没写(a,b,c)或(p_a,p_b,p_c)，扣一半
- c)只写了一种正确的组合，扣一半

没有0.5或0.75分，1.75=1

- (3) 命题的必要性是否成立?若否，请给出反例。若是，不需要证明。
- 答案：(1 分) 成立

- 写“是”即可
- 试图曲解题意的，如写“若入栈序列中存在相同元素，则不成立；若不存在相同元素，则成立”等，属于画蛇添足，0分，因为题目已经说得很明白了——入栈序列就是1到N

- 补充：可用数学归纳法证明必要性。假设命题不成立，以$p_b = n$为界将序列$\{p_1, p_2, \ldots, p_n\}$一分为二，可证$\forall a \in [1, b-1], c \in [b+1,n]$，有$p_a < p_c$成立，然后归纳假设。

■ 第四章

1. 设有两个字符串，一个为“ 142857 ”，另一为 285714 ”，可以看出，若将第一个字符串首尾相连，第二个则是从‘ 2 ’开始向后遍历一周得到的。我们将第二个字符串称为第一个字符串的旋转词。请你设计一个算法，给定两个长度为 n 的字符串，判断第一个字符串是否是第二个字符串的旋转词，要求算法的时间复杂度为 O(n) 。

评分标准:

满分11分：KMP算法或其它O(n)算法

9分：使用KMP时搞反了模式串和目标串，求了目标串的Next数组

6分：O(n^2)算法

3分：算法不能给出正确结果（实际就是错误答案，酌情给了3分）

0分：未写或写了一两句与不能体现解答的语句

此外，对于存在一些小错误的解答，酌情扣了1–2分

可能出现的问题:

1、求Next数组时求的是目标串的next数组，而不是模式串

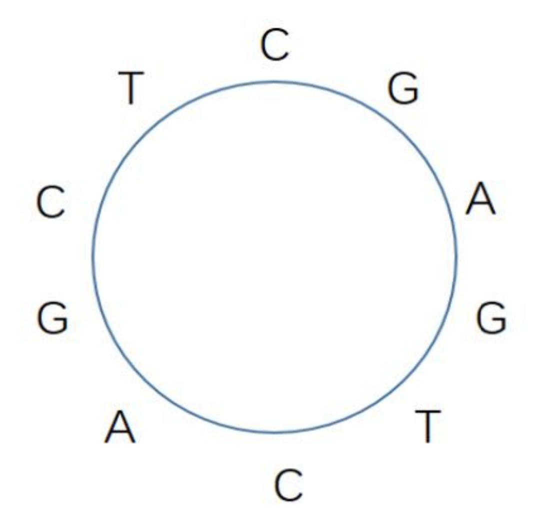
2、部分同学使用了O(n^2)的算法

参考答案

分析：假设第一个字符串为“142857”，它的长度为 6。然后将其复制一遍，放在其后，构成142857142857”，则发现，它的每一个长度为 6的字串都是原字符串的旋转词（自身也是自身的旋转词）。然后用 KMP算法，用 O(n)时间的复杂度即可解决。

答案：假设原串为N，长度为 n，要证的字符串为 M，证明 M是 N的旋转词。则复制一遍 N放在 N后，构成长度为 2n的字符串，然后用 KMP算法来判断 M是否是长度为 2n的串的子串，是，则表示 M是 N的旋转词，否则， M不是 N的旋转词。

- 2. 长度为 n 的环状串有 n 种表示法，分别为从某个位置开始顺时针得到。例如，下图的环状串有 10 种表示:CGAGTCAGCT，GAGTCAGCTC，AGTCAGCTCG 等。在这些表示法中，字典序最小的称为"最小表示"。 输入一个长度为 n 的环状 DNA 串(只包含 A、C、G、T这 4 种字符)的一种表示法，请简述如何输出该环状串的最小表示。例如，CTCC 的最小表示是 CCCT，CGAGTCAGCT 的最小表示为 AGCTCGAGTC。



答案:

法一:

可以通过一轮朴素比较字符串得到最小表示。用一个变量 ans 表示目前为止，字典序最小串在输入串中的起始位置，然后不断更新 ans。单次比较最坏情况下O(n)，最好情况下O(1)

worse case 为`aa...ac`（n – 1个a），时间复杂度O(n^2)

- 法二：原理上和kmp有点类似，利用已匹配pattern的信息
- 考虑对于一对字符串A, B, 它们在原字符串 S 中的起始位置分别为 i, j , 且它们的前 k 个字符均相同，即
  - $A := S[i \ldots i+k-1] = S[j \ldots j+k-1] =: B$
- 不妨先考虑 $S[i+k] > S[j+k]$ 的情况，我们发现起始位置下标 l 满足 $i \le l \le i+k$ 的字符串均不能成为答案。因为对于任意一个字符串 S_{i+p}（表示以 $i+p$ 为起始位置的字符串）一定存在字符串 S_{j+p} 比它更优。
- 所以我们比较时可以跳过下标 $l \in [i, i+k]$, 直接比较 S_{i+k+1} 和 S_j

- ref: https://oi-wiki.org/string/minimal-string/

■ 具体代码：

```
int k = 0, i = 0, j = 1;
while (k < n && i < n && j < n) {
  if (sec[(i + k) % n] == sec[(j + k) % n]) {
    k++;
  } else {
    sec[(i + k) % n] > sec[(j + k) % n] ? i = i + k + 1 : j = j + k + 1;
    if (i == j) i++;
    k = 0;
  }
}
i = min(i, j);
```

<3n

<2n

■ 时间复杂度O(n)。

- 扣分细则：

- – 没有时间复杂度分析：–1（好多人忘了，但设计算法这是必须的）
- – 算法复杂度高于O(nlogn)：–1（没啥办法，统一标准吧）
- – 如果使用后缀数组(https://blog.csdn.net/neweryyy/article/details/109907238)，后缀自动机(https://blog.csdn.net/dyx404514/article/details/8714165)，Lyndon分解(https://www.cnblogs.com/purinliang/p/14321322.html)，正确有复杂度的分析即给满分
- – 很多使用了字符串hash：酌情扣0–2分（视分析的程度而定）
- – 算法错误：酌情扣1–5分

■ 第五章

1. 一棵 Huffman 树的带权外部路径长度定义为所有叶结点权值与其深度的乘积之和。设根结点的深度为 0，对集合 { 2, 3, 5, 7, 8 } 构造二叉 Huffman 树，则其最小带权外部路径长度为________________。

答案：55

2. 用数组存储一个有 50 个元素的最小值堆。已知堆中没有重复元素。给出最大元素的下标可能的取值范围为__________到 __________。

答案：25~49

最小堆是完全二叉树且子节点不小于父节点，已知堆中没有重复元素，所以父节点一定小于子节点，即任何内部节点都不可能是最大值。最大值只可能出现在叶节点上，堆用数组存是分层存放的，要求最大元素的下标的可能的取值范围，也就是求叶节点的下标的范围。

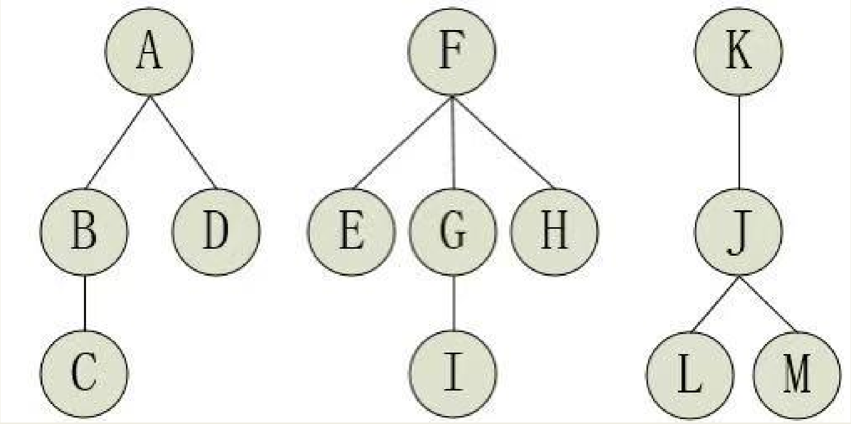
最后一个叶节点必然是存在第 49 位（下标从 0 开始），所有只要求出叶节点的个数就可以了。50 个元素的堆，从上到下每层的元素个数分别是 1,2,4,8,16,19。倒数第二层有 19/2（上取整）个内部节点，所以叶节点的个数是 19+16–10=25 个，下标范围是 25~49。

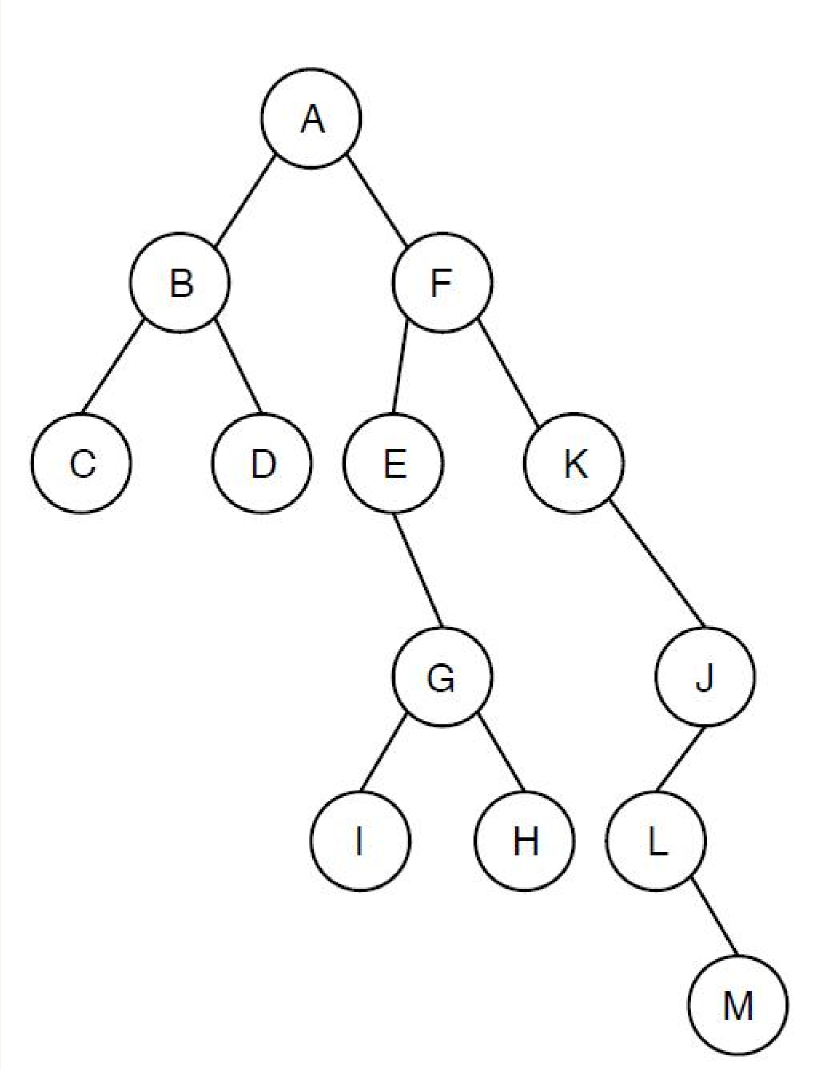
■ 第六章

1. 试证：一个具有n(n>1)个结点的m叉树有n(m−1)+1个空指针。

证明：在n个结点的m叉树一共有m*n个指针域，已经使用的为n−1，所以空指针的个数为n(m−1)+1.

2. 设如下图所示的森林T有三棵树，该森林转化为二叉树后，与结点G深度相同的结点有________________ 。





答案：J

3. 一棵树 T有 20个度为 4的结点，10个度为 3的结点，1个度为 2的结点，10个度为 1的结点，则树 T的结点个数为_____。

答案：**123**

4. 请证明非空满 K叉树的叶结点数目为 $(K-1)*n+1$，其中 n为分支结点数目。

答案：数学归纳法证明如下：

1)当 n=0 时，只有一个根结点，此时叶结点数目为 1，而 $(K-1)n+1$ 也为 1

n=1

时，$(K-1)n+1=K$，这是只有一个分支结点的情况，结果正确；

2)．假设 n=m (m>=1, m 为自然数)时叶结点数目为（K–1）m+1,

则 n=m+1 时，相当于把 n=m 时的 K叉树里的一个叶结点扩展为分支结点，叶结点增加

K–1个

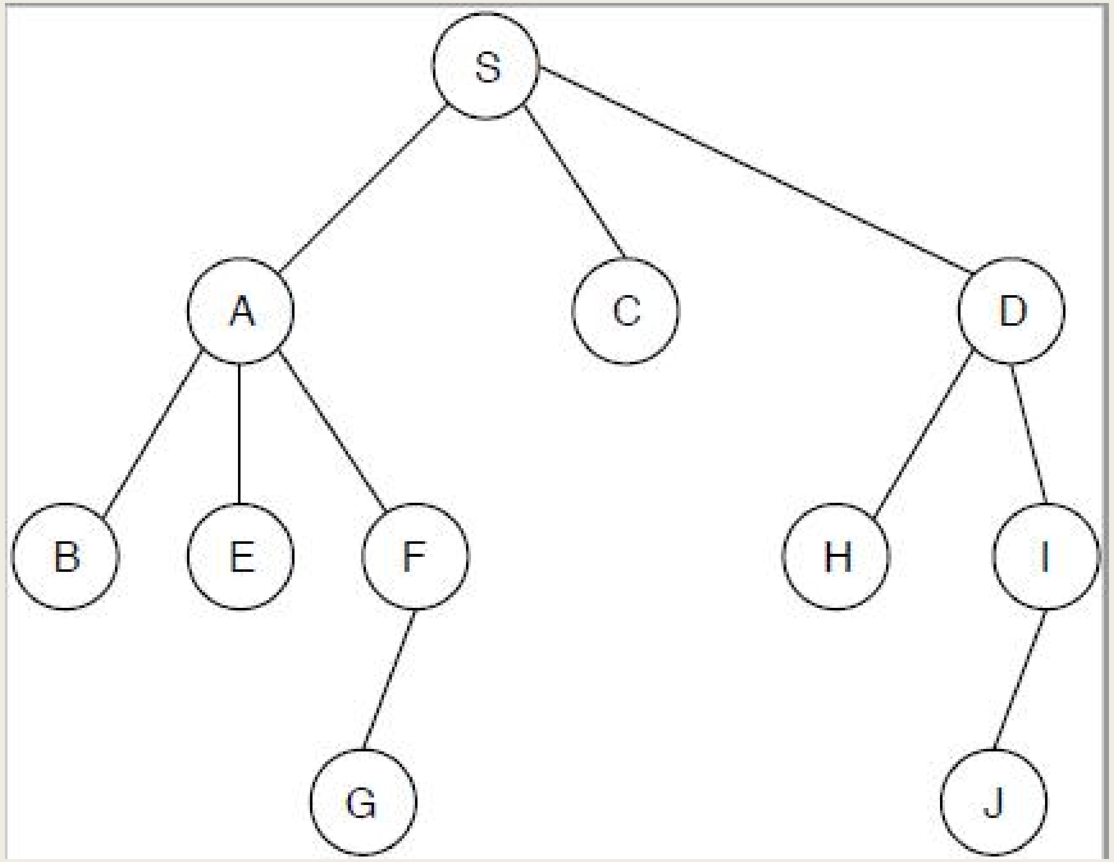
故 n=m+1 时，叶结点数目为 $(K-1)m+1+K-1= (K-1)(m+1)+1 = (K-1)n+1$

即 n=m+1 时此公式仍然成立。

综 1、2所述，该公式成立，证毕。

5. 一棵树按照先根次序遍历的结点序列为SABEFGCDHIJ，后根次序遍历该树为BEGFACHJIDS，那么E结点的兄弟结点 有 ________________ 。

答案：B，F



祝大家考试顺利！