1. 假定 $f(n)$ 与 $g(n)$ 都是渐进非负函数，判断下列等式或陈述是否一定是正确的，并简要解释你的答案.

a) $f(n) = O(f(n)^2)$.

b) $f(n) + g(n) = \Theta(max(f(n), g(n)))$.

c) $f(n) + O(f(n)) = \Theta(f(n))$.

d) if $f(n) = \Omega(g(n))$, then $g(n) = o(f(n))$. (注意是小 o)

a) 不一定正确

假设 $f(n)$ 为单调递减函数且存在正常量 n_1，使得对所有 $n > n_1$，有 $f(n) < 1$

由于 $f(n)$ 是渐进非负函数，则存在正常量 n_0，使得对所有 $n > n_0$，有 $f(n) > 0$

取 $N = \{ n_0 , n_1 \}$，则 $n > N$， $0 < f(n) < 1$，故 $f(n)^2 \leqslant f(n)$

进而 $f(n) \neq O(f(n)^2)$

b) 正确

取 $h(n) = \max\{ f(n) , g(n) \}$，则有

$h(n) \leqslant f(n) + g(n) \leqslant 2h(n)$

故 $f(n) + g(n) = \Theta(h(n)) = \Theta(max\{ f(n) , g(n) \})$

c) 正确

令 $g(n) = O(f(n))$, 则存在正常量 c、n_0，使得对所有 $n > n_0$，有 $0 \leqslant g(n) \leqslant cf(n)$

则有 $f(n) \leqslant f(n) + g(n) \leqslant (c+1)f(n)$，故 $f(n) + O(f(n)) =f(n) + g(n) = \Theta(f(n))$

d) 正确

$f(n) = \Omega(g(n)) \Rightarrow$ 存在正常量 c、n_0，$n > n_0$，有 $0 \leqslant cg(n) \leqslant f(n)$

故 $0 \leqslant g(n) \leqslant \frac{1}{c} f(n)$，进而有 $0 \leqslant g(n) < \frac{1}{c}f(n)$

令 $c_0=\frac{1}{c}$， 存在正常量 c_0、n_0，$n > n_0$，有 $0 \leqslant g(n) < c_0f(n)$

$\Rightarrow g(n) = o(f(n))$

2. 时间复杂度

a) 证明 $lg(n!) = \Theta(nlg(n))$(课本等式 3.19), 并证明 $n! = \omega(2n)$ 且 $n! = o(n^n)$.

lg(n!) = Θ(nlg(n)) 的证明

$$
\begin{aligned}
&lg(n!) = lg(n) + lg(n-1) + \cdots + lg(1) \leqslant lg(n) + lg(n) + \cdots + lg(n) = nlg(n) \\
&lg(n!) = lg(n) + lg(n-1) + \cdots + lg(\frac{n}{2}+1) + lg(\frac{n}{2}) + \cdots + lg(1) \\
&\geqslant lg(n) + lg(n-1) + \cdots + lg(\frac{n}{2}+1) + lg(\frac{n}{2}) \\
&\geqslant lg(\frac{n}{2}) + lg(\frac{n}{2}) + \cdots + lg(\frac{n}{2}) + lg(\frac{n}{2}) \\
&= \frac{n}{2}lg(\frac{n}{2}) = \frac{n}{2}(lg(n) - lg(2)) \geqslant \frac{1}{2}n(lg(n) \\
&\Rightarrow \frac{1}{2}n(lg(n) \leqslant lg(n!) \leqslant nlg(n) \\
&\Rightarrow lg(n!) = \Theta(nlg(n))
\end{aligned}
$$

n! = ω(2n) 的证明

$$\lim_{n\to+\infty}\frac{n!}{2n}=\lim_{n\to+\infty}\frac{n\cdot(n-1)\cdots1}{2n}=\lim_{n\to+\infty}\frac{(n-1)\cdot(n-2)\cdots1}{2}=+\infty$$

故n! = ω(2n)

n! = *o*(n^n) 的证明

$$0\leqslant\frac{n!}{n^n}=\frac{n}{n}\cdot\frac{n-1}{n}\cdots\frac{1}{n}\leqslant\frac{1}{n}$$

由两边夹定理，$\lim_{n\to+\infty}\frac{1}{n}=0$，得 $\lim_{n\to+\infty}\frac{n!}{n^n}=0$，故n! = $o(n^n)$

b) 使用代入法证明 *T*(*n*) = *T*(⌈n/*2*⌉) +1 的解为 *O*(*lgn*).

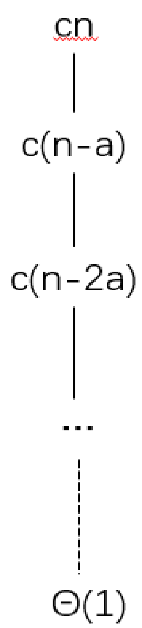
设 n > n_0 时，存在正常量 c，使得 T(n) ⩽ clg(n)

则当 c ⩾ 2 时，

$$T(n)=T(\lceil\frac{n}{2}\rceil)+1\leqslant clg(\lceil\frac{n}{2}\rceil)+1\leqslant clg(\frac{n}{\sqrt{2}})+1=clg(n)-clg(\sqrt{(2)})+1=clg(n)+1-\frac{c}{2}\leqslant clg(n)$$

故 T(n) = O(lgn)

c) 对递归式 *T*(*n*) = *T*(*n* - *a*)+*T*(*a*)+*cn*, 利用递归树给出一个渐进紧确解, 其中 *a* ≥ 1 和 *c* > 0 为常数.



每一层都只有一个结点，深度为 j 的叶结点的代价为 c(n-(j-1)a)，设为为叶结点时的深度为 k，则 T(n-ka) = T(1)，故 $k=\frac{n-1}{a}$，即树的深度为$\frac{n-1}{a}$

总时间：

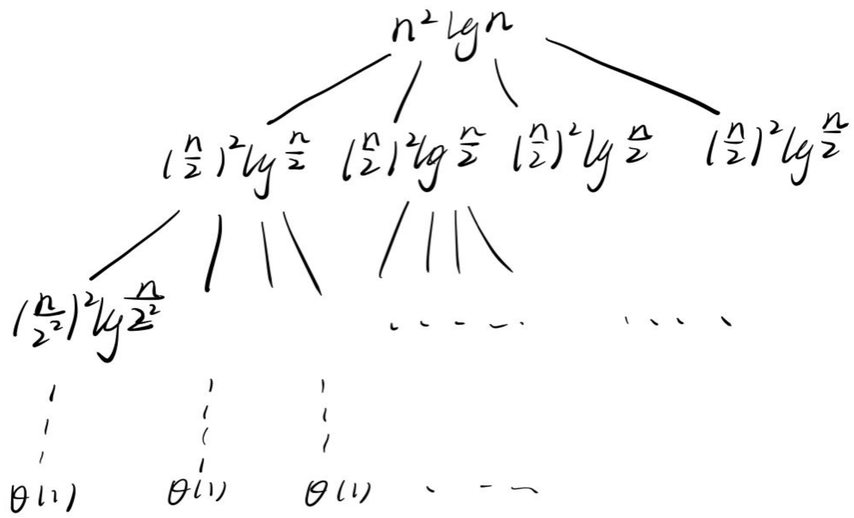
$$
\begin{aligned}
total &= \Theta(1) + kT(a) + \sum_{i=0}^{k-1} c(n-ia) \\
&= \Theta(1) + \frac{n-1}{a}T(a) + \sum_{i=0}^{\frac{n-1}{a}-1} c(n-ia) \\
&= \Theta(1) + (n-1)T(1) + c\frac{(n+a+1)(n-1)}{2a} \\
&= \Theta(n) + \frac{c}{2a}(n^2+(n-1)a-1) \\
&= \Theta(n^2)
\end{aligned}
$$

d) 主方法能应用于递归式 *T(n)* = 4*T(n/2)* +n^2 *lgn* 吗？请说明为什么可以或者为什么不可以. 给出这个递归式的一个渐进上界.

不可以，a = 4 , b = 2 , f(n) = n^{2}lg(n) , $n^{log_b a} = n^2$ ，对任意 ε，

$\frac{f(n)}{n^{log_b a}} = log(n)$ 的渐进小于 n^{ε}，f(n) = n^{2}lg(n) 不是多项式意义的大于$n^{log_b a} = n^2$，故不可以用主方法

其递归树如下



$$
\begin{aligned}
total &= \Theta(n^{log_2 4}) + \sum_{i=0}^{\lg n-1} 4^i(\frac{n}{2^i})^2 \lg\frac{n}{2^i} \\
&= \Theta(n^2) + \sum_{i=0}^{\lg n-1} n^2(\lg n - i) \\
&= \Theta(n^2) + n^2\sum_{i=0}^{\lg n-1} (\lg n - i) \\
&= \Theta(n^2) + n^2\frac{(1+\lg n)\lg n}{2} \\
&= O(n^2(\lg n)^2)
\end{aligned}
$$

$\Rightarrow$ 渐进上界为 $n^2(\lg n)^2$

3. 对下列递归式, 使用主方法求出渐进紧确解:

a) $T(n) = 2T(n/4) + \sqrt{n}$

b) $T(n) = 2T(n/4) + n^2$

a) a = 2 , b = 4 , f(n) = $\sqrt{n}$, $n^{log_b a} = \sqrt{n}$, f(n) = Θ($n^{log_b a}$) = O($n^{\frac{1}{2}}$)，故 T(n) = Θ($\sqrt{n}\lg n$)

b) a = 2 , b = 4 , f(n) = n^2 , $n^{log_b a} = \sqrt{n}$, f(n) = n^2 = Ω($n^{log_b a+\varepsilon}$) = Ω($n^{\frac{1}{2}+\varepsilon}$)，其中，$\varepsilon = \frac{3}{2}$，故 T(n) = Θ(n^2)

4. 考虑以下查找问题：

输入: *n* 个数的一个序列 *A* = a1,a2,...,an 和一个值 *v*.

输出: 下标 *i* 使得 *v* = *A*[*i*] 或者当 *v* 不在 *A* 中出现时, *v* 为特殊值 *NIL*.

a) 写出线性查找的伪代码, 它扫描整个序列来查找 *v*. 使用一个 Loop Invariant (循环不变式) 来证明你的算法是正确的.

b) 假定 *v* 等可能的为数组中的任意元素，平均需要检查序列的多少元素？最坏情况又如何呢？用 Θ记号给出线性查找的平均情况和最坏运行时间.

a)

```
j=1
while j ≤ A.length
    if A[j] == v
        i = j
        break
    j = j + 1
if j == A.length + 1 then
    i = NIL
```

初始化：j 为 1时，此时数组内下标比 1 小的元素（为空）中不含 v，循环不变式成立

保持：每次循环 j 迭代 1，由于 A[1 ... j-1] 没有找到 v，故此时应判断 A[j] 是否为 v，即可知道 A[1 ... j] 中是否能找到 v

终止：当找到了 v 或 j = A.length 时算法终止，不管是否能找到 v，由于每次迭代 j 都会加 1，故算法最终一定会终止。若找到了 v，则 j 即为对应的下标，否则 j = A.length + 1，此时将 i 设为 NIL

由此可知算法是正确的

b)

$$
\begin{aligned}
ave &= \frac{1}{n}\sum_{i=1}^{n} i = \frac{n+1}{2} \\
wrost &= n
\end{aligned}
$$

故平均要检查 $\frac{n+1}{2}$ 个元素，最坏情况下要检查 n 个元素，avetime = Θ(n) , wrosttime = Θ(n)

5. 堆排序：

对于一个按升序排列的包含 *n* 个元素的有序数组 *A* 来说，Heapsort 的时间复杂度是多少？如果 *A*是降序的呢？请简要分析并给出结果.

升序排序：O(nlgn)，此时为最坏情况，由于最大的结点不是 i 而是 i 的左右孩子中的一个，故MAX-HEAPIFY内部会递归调用，高度为 h 的结点调用MAX-HEAPIFY的代价为 O(h)，且高度为 h 的堆最多包含 $\lceil \frac{n}{2^{h+1}} \rceil$个结点，故BUILD-MAX-HEAP总代价为

$$
\sum_{h=0}^{\lfloor \lg n \rfloor} \lceil \frac{n}{2^{h+1}} \rceil O(h) = O(n \sum_{h=0}^{\lfloor \lg n \rfloor} \frac{h}{2^h}) = O(n)
$$

在HEAPSORT中，n-1 次调用了 MAX-HEAPIFY，每次的时间复杂度为 O(lgn)，故总时间复杂度为 O(n) + O(nlgn) = O(nlgn)

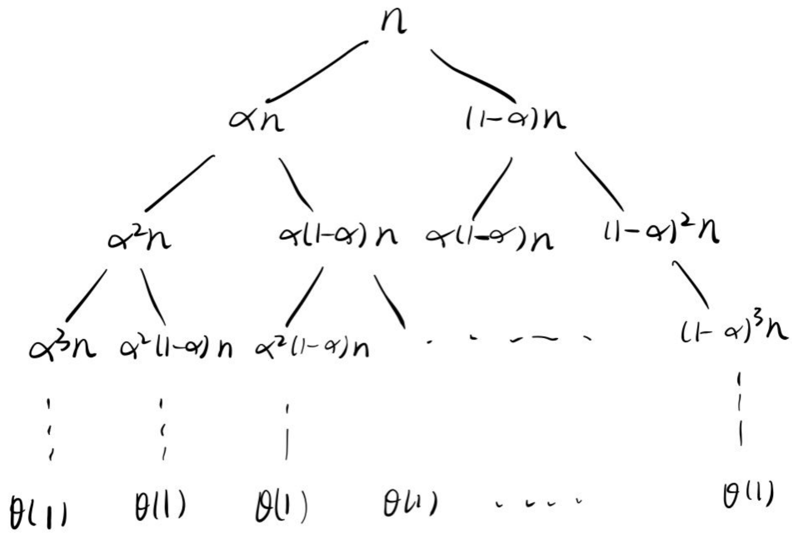
降序排列：O(n)，此时为最好情况，由于最大的结点就为 i 而不是 i 的左右孩子，故每次调用MAX-HEAPIFY的时间复杂度为 O(1)，　　 BUILD-MAX-HEAP共需 $\lfloor \frac{n}{2} \rfloor$ 次这样的调用，故BUILD-MAX-HEAP时间复杂度为 O(n)

6. 快速排序：

1. **假设快速排序的每一层所做的划分比例都是 1 α : α，其中 0 < α ≤ 1/2 且是一个常数.试证明：在相应的递归树中，叶结点的最小深度大约是 lg*n*/lgα，最大深度大约是 lg*n*/lg(1- α) (无需考虑舍入问题).**

2. **试证明：在一个随机输入数组上，对于任何常数 $0 < \alpha \leq 1/2$，Partition 产生比 $1 - \alpha : \alpha$ 更平衡的划分的概率约为 $1-2\alpha$.**

    1.判断树为：

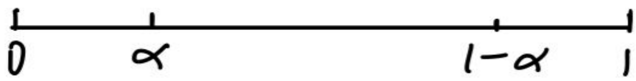


由于$0 < \alpha \leq 1/2$，系数 α^k 使 $\alpha^k n$ 下降得最快，系数$(1-\alpha)^k$ 使 $(1-\alpha)^k n$ 下降得最慢，分别令 $\alpha^{k1} n = 1$, $(1-\alpha)^{k2} n = 1$，得

最小深度 $k1 = \frac{lgn}{lg\alpha}$

最大深度 $k2 = \frac{lgn}{lg(1-\alpha)}$

2.



划分比列落在[α , 1-α]时，Partition 产生比 1 - α : α 更平衡的划分，其概率 $p = \frac{1-\alpha-\alpha}{1} = 1 - 2\alpha$