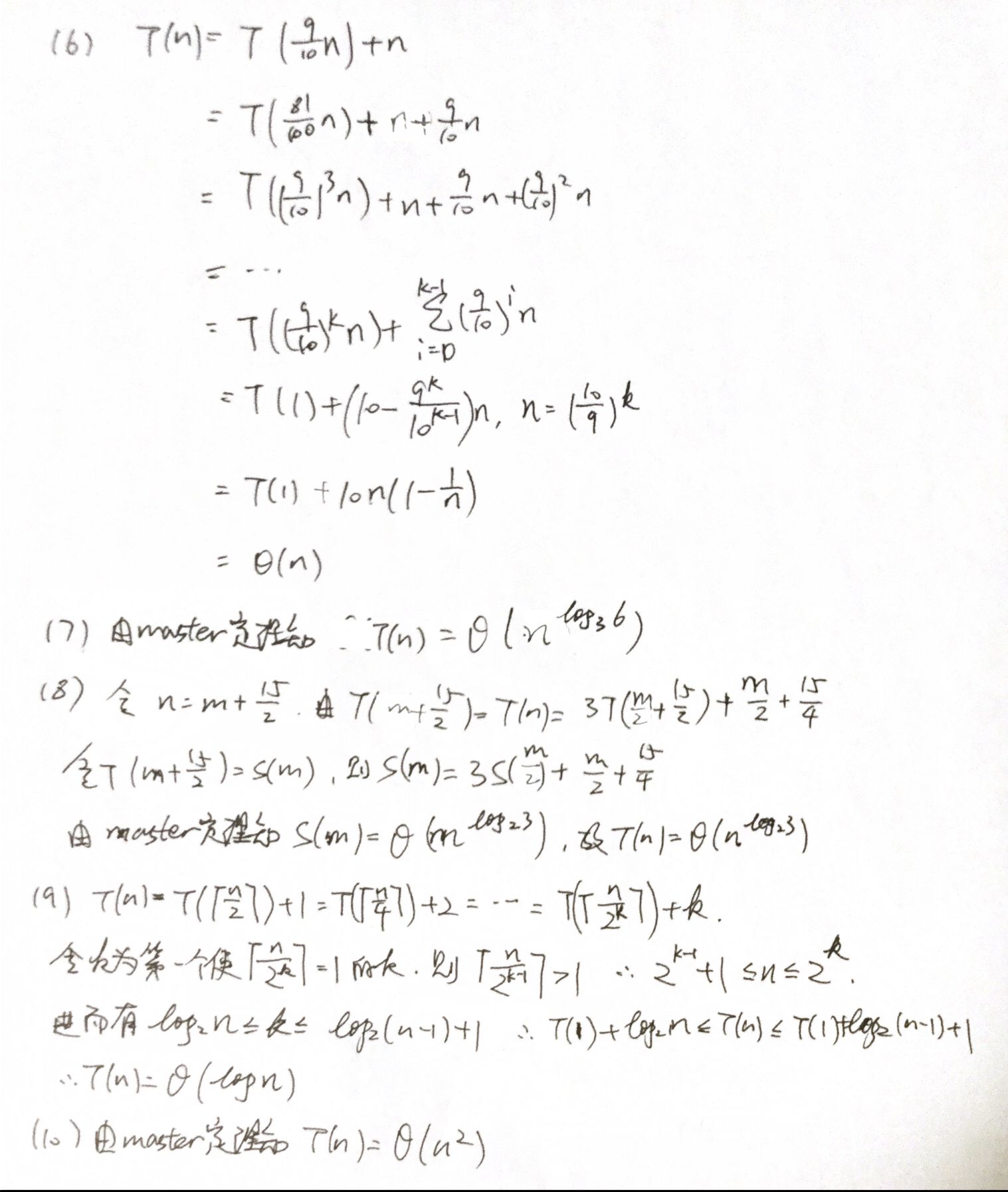
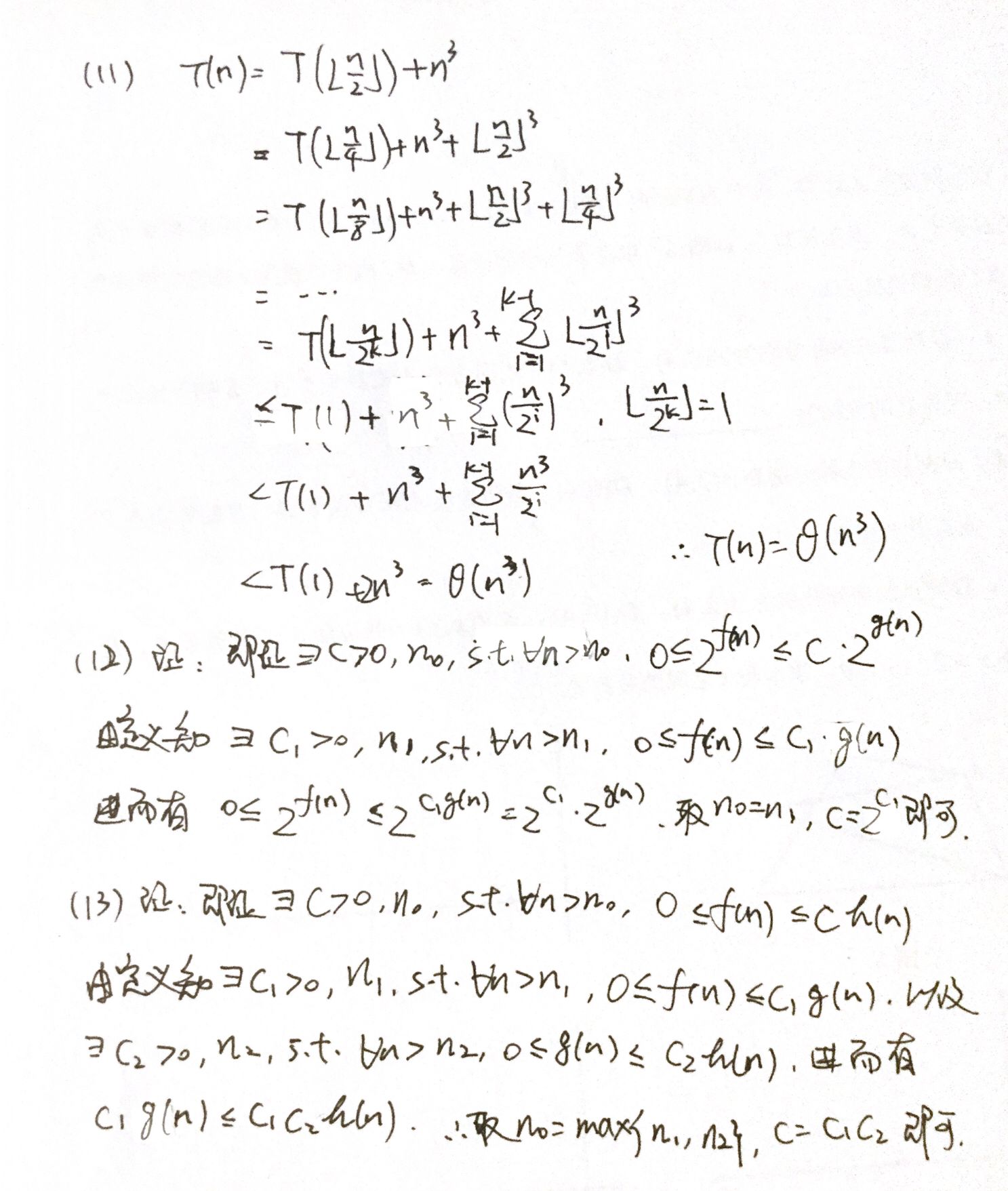
1. 证明或否证：*f* (*n*)+o(*f*(*n*))=Θ(*f*(*n*))
2. 证明：Θ(*f*(*x*)) + O(*g*(*x*)) = O( max(*f*(*x*), *g*(*x*))).
3. 证明或给出反例：Θ(*f*(*n*))∩o(*f*(*n*))=∅
4. 证明：设*k*是任意常数正整数，则*logkn* = o(*n*)
5. 证明：*logn*!=Θ(*nlogn*)

前5题解答用LaTeX写的，但是太费时间了所以后面还是手写扫描吧。

1. 用迭代法解方程 T(*n*)＝T(9*n*/10)+*n*
2. 解方程 T(*n*)=6T(*n*/3)+*logn*
3. 解方程 T(*n*) =3T(*n*/3 +5) + *n*/2
4. 解方程 T(*n*)= T (⎡*n*/2⎤ )+1
5. 解方程 T(*n*)=9T(*n*/3)+*n*



1. 解方程 T(*n*)=T( ⎣*n*/2⎦ )+*n*3
2. 证明或否证：如果*f*(*n*)=O(*g*(*n*))，那么2*f*(*n*)=O(2*g*(*n*))
3. 令*f*, *g*和*h*为定义在正整数上的正实数函数。假设*f*(*n*)=*O*(*g*(*n*))，*g*(*n*)=*O*(*h*(*n*))，证明*f*(*n*)=*O*(*h*(*n*)).



1. 将下列函数按它们在n→∞时的无穷大阶数，从小到大排序。

n, nlg7, n2.5, n3lgn, n2lgn, nlgn

从小到大依次为：nlg7、n、nlgn、n2lgn、n2.5、n3lgn

1. 一个时间复杂性为O(*n*2)的算法的运行时间一定大于一个时间复杂性为O(n)的算法么？为什么？

当然不了，通常来讲前者规模远小于后者的话运行时间就会短于后者