

5. (1)

(a) Let  $C=1, N=0$

$$\ln n! = \ln n + \ln(n-1) + \ln(n-2) \dots \ln 1 \leq \ln n \times n = \ln n^n \quad \text{for } n \geq N$$

(b) Let  $C=1, N=0$

$$n^{\ln n} = e^{\ln n \ln n} = e^{\ln n^{\ln n}} = c^{\ln n} \Rightarrow c^{\ln n} \leq n^{\ln n} \text{ and } n^{\ln n} \leq c^{\ln n} \\ \Rightarrow n^{\ln n} = \Omega(c^{\ln n}) \text{ and } n^{\ln n} = O(c^{\ln n})$$

(c) False,  $\sin n = 0$  for  $n = (2k + \frac{3}{2})\pi$  for  $k=1, 2, \dots$

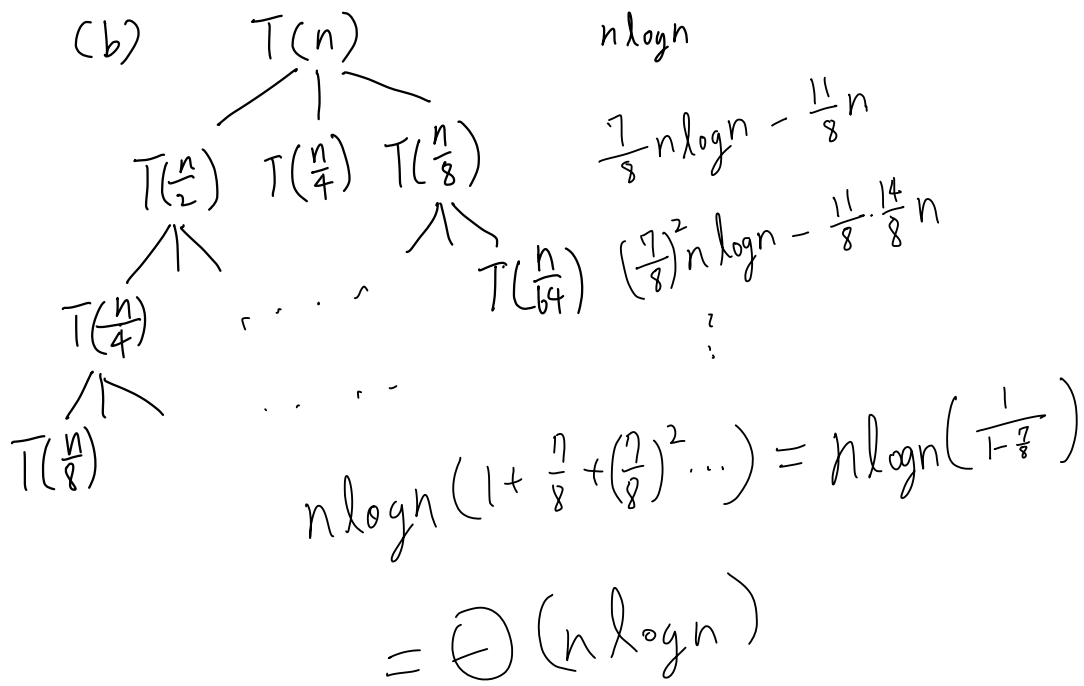
$$\Rightarrow \forall C, N, \exists n \geq \max(\frac{1}{C^2}, N)$$

$$\text{s.t. } \sin n = 0 \Rightarrow \underline{C \cdot n^{\frac{1}{2}}} \geq C \cdot (C^2)^{\frac{1}{2}} = 1 \geq 1 = \underline{n^{\sin n}}$$

$$(d) \lim_{n \rightarrow \infty} \frac{(\ln n)^3}{n} = \lim_{n \rightarrow \infty} \frac{3(\ln n)^2 \cdot \frac{1}{n}}{1} = \lim_{n \rightarrow \infty} \frac{3 \cdot 2 \cdot \ln n \cdot \frac{1}{n}}{1} \\ = \lim_{n \rightarrow \infty} \frac{6 \cdot \frac{1}{n}}{1} = 0$$

$$\begin{aligned}
 (2) \quad (a) \quad T(n) &= 2T(n-1) + 1 \\
 &= 2(2T(n-2) + 1) + 1 \\
 &= 2^2 T(n-2) + 2 + 1 \\
 &= 2^3 T(n-3) + 4 + 2 + 1 \\
 &= 2^n T(0) + 2^n - 1 \\
 &= 2 \cdot 2^n - 1 = \Theta(2^n)
 \end{aligned}$$

(b)



$$\begin{aligned}
 &n \log n \\
 &\frac{7}{8} n \log n - \frac{11}{8} n \\
 &\left(\frac{7}{8}\right)^2 n \log n - \frac{11}{8} \cdot \frac{14}{8} n \\
 &\vdots \\
 &n \log n \left(1 + \frac{7}{8} + \left(\frac{7}{8}\right)^2 + \dots\right) = n \log n \left(\frac{1}{1-\frac{7}{8}}\right) \\
 &= \Theta(n \log n)
 \end{aligned}$$

$$(c) \quad n^{\log_2 4 - 0.1} = \Omega(n \log n)$$

$$T(n) = \Theta(n^2)$$

$$(d) \quad T(n) = \sqrt{n} T(\sqrt{n}) + n$$

$$\Rightarrow \frac{T(n)}{n} = \frac{T(\sqrt{n})}{\sqrt{n}} + 1$$

$$\text{Let } G(k) = \frac{T(2^k)}{2^k}$$

$$\Rightarrow G(k) = G\left(\frac{k}{2}\right) + 1$$

$$G(k) = \Theta(\log k), \text{ by master thm.}$$

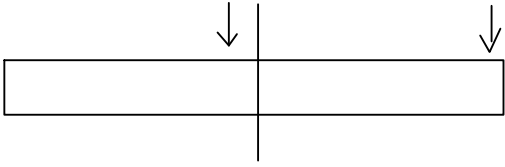
$$k = \log n$$

$$T(n) = n \cdot G(\log n) = \Theta(n \log n \log n)$$

# 6.

(1) Divide and Conquer

&(2)



if the number of elements is less than 3 just do it brute force  
first we cut the array into two half

and recursive call the two array

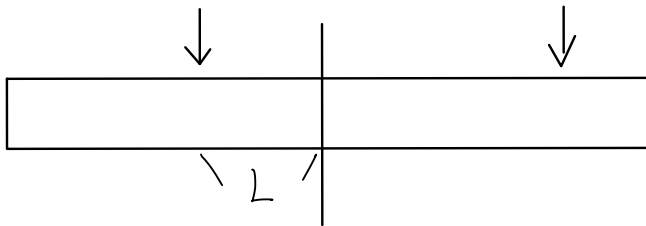
we will sort the array at the end of function

so we can know that the two array is sorted

now, we use two pointer from the back of two array

and while the element of first pointer is greater than the second

we decrease the first pointer, for every element in the second half



every iteration we will add  $L$  to our result

The two pointer totally move  $\leq \frac{n}{2}$  each

finally we sort the entire array

(using the approach of merge sort,  $O(n)$ )

return the result + result of first half + result of second half

$T(n) = 2T(\frac{n}{2}) + n$ . since every recursive call take  
 $O(n)$  to combine. And  $T(n) = O(n \log n)$  by master thm.

(3) Because every exchange will decrease number of inversions by 1.

(4) We can transform the constraint ( $A_i$  should go to  $X_i$ )

into an array of  $X$ s in the order of  $A$

and the rest of the player just insert in the original order

Then just use the algorithm

(5) Every leg exchange two element so the minimum legs is corresponding to the number of exchanges of the bubble sort.

7.

$$(1) f = [-3, 0, 6, 4, 0, -1, -2, 3, -3, 9]$$

$9 + (-3) + 0 + 6 + 4 = 16$  is one of the answer

(2) Reference to The Kadane's Algorithm

Case 1 : 沒經過  $n, 1$

Variables: start index, end index, max sum end at  $i$ , maximum sum

我們掃過整個 array 並在每次比較 (第  $i$  個 element) 與 (第  $i$  個 element + 結尾在第  $i-1$  個的 max sum) 來更新 max sum end at  $i$ , 然後在與目前看過的最大值比較, 若更大, 更新 start index, end index, maximum sum  $O(N)$  time

Case 2 : 有經過  $n, 1$

用上面的演算法找出 minimum sum in  $2 \sim n-1$   $O(N)$  time  
再用全部人的加總減去他

Time complexity  $O(N)$ , Space complexity  $O(1)$

(3)

Case 1 : 沒經過  $n, 1$

用兩個 array (leftmax, rightmax) 分別裝在結尾在第  $i$  個的 max sum (從左跟從右)  $O(N)$  space  
再掃一次從  $i = 2 \sim n-1$ , 找出最大的  $\text{leftmax}[i-1] + \text{rightmax}[i+1]$   $O(N)$  time

Case 2 : 有經過  $n, 1$

用兩個 array 分別裝第  $i$  個左邊/右邊最小的元素  $O(N)$  space  
再用(2)的演算法, 但是在每一輪判斷時多考慮左右最小的元素 來決定要不要更新 min sum end at  $i$

Time & Space complexity  $O(N)$

$O(N)$  time