P1:

Yes, we can do it in $O(\log n)$.

Since for array $A[1, \dots n]$, we have $A[i] \le A[i+1]$,

we can use binary search to get sum of array.

As we all know, the time complexity is $O(\log n)$;

for example:

we have an ordered array:

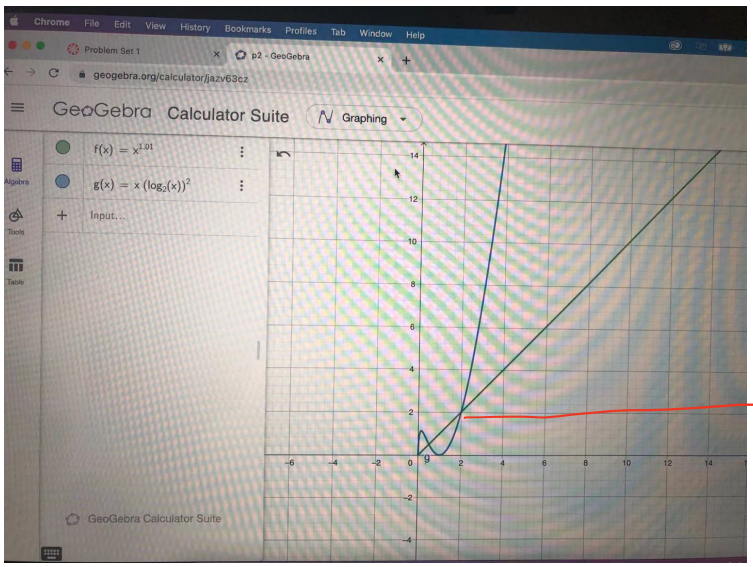
index :	0	1	2	3	4	5	6	7	8	9	10	11	12
number:	0	0	1	2	2	3	4	5	6	7	8	8	9

So, we can have $9 \cdot \log n + 10 \cdot (1)$ [adding num is $O(1)$, binary search is $O(\log n)$]

Finally, we do it $O(\log n)$.

P2.

(a):



→ n_0

From the question, we have

$f(n) = n^{|2|}$, $g(n) = n(\log n)^2$.

From the chart, we can get:

For all $n \geq n_0$, $n > 0$, we have n_0 that

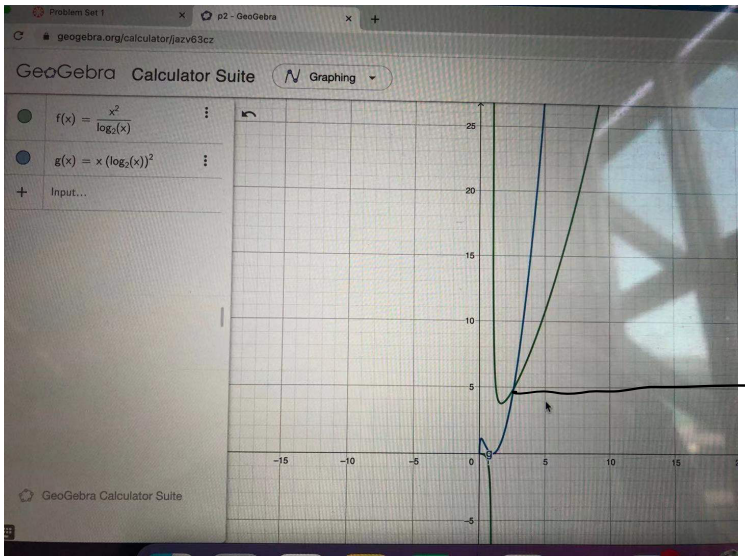
$f(n_0) = g(n_0)$,

for all $n > n_0$, we have $g(n) > f(n)$.

so, we can get $f(n) = O(g(n))$; $f = O(g)$.

(b): From the question, we know

$$f(n) = \frac{n^2}{(\log n)}, \quad g(n) = n(\log n)^2$$



$\rightarrow n_0$

From this chart, we can get:

For all $n > n_0$, we have $g(n) > f(n)$.

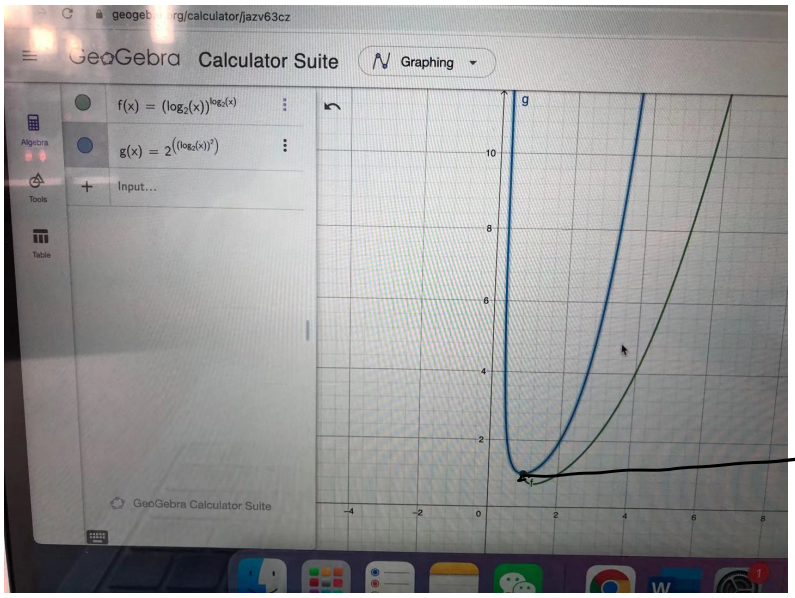
and we can also get $f(n_0) = g(n_0)$.

so, we can conclude $f = O(g)$.

P3:

(a). From the question, we have:

$$f(n) = (\log n)^{\log n}, \quad g(n) = 2^{(\log n)^2}.$$



$\rightarrow n_0$

From the chart, we can get:

for all $n \geq n_0$ or $0 < n < n_0$, we all have $g(n) \geq f(n)$.

for n_0, we have $f(n_0) = g(n_0)$.

we can conclude: $f = O(g)$;

(b). From the question, we have:

$$f(n) = \sum_{i=1}^{n} i^k$$

$$= 1^k + 2^k + 3^k + \cdots + (n-1)^k + n^k$$

$$g(n) = n^{(k+1)}$$

$$= n \cdot n^k$$

$$= (n^k + n^k + n^k + \cdots + n^k).$$

From above, for all $n > n_0$ and $k > 1$, we can have

$$g(n) > f(n); \quad f(n) = O(g(n)).$$

For example: $k = 2$, $n = 5$.

$$f(5) = 1^2 + 2^2 + 3^2 + 4^2 + 5^2 = 55$$

$$g(5) = 5^2 + 5^2 + \cdots + 5^2 = 125$$

We also have:

$$f(n) = 1^k + 2^k + \dots + n^k$$

$$\geq \left(\frac{n}{2}\right)^k + \left(\frac{n}{2}\right)^k + \dots + \left(\frac{n}{2}\right)^k$$

$$= \left(\frac{n}{2}\right) \cdot \left(\frac{n}{2}\right)^k$$

$$= \left(\frac{n}{2}\right)^{k+1}$$

$$= \left(\frac{1}{2}\right)^{k+1} \cdot n^{k+1}$$

So, we can conclude, there is a constant c_2 and $f(n) = \Omega(g(n))$

Therefore, $f(n) = \Theta(g(n))$, or $f = \Theta(g)$.