**First Question**

T(n) = time complexity  
O(n) = asymptotic time complexity

T(n) = O(g(n)) that have means g(n) is the greater ordo in T(n)  
Reference: <https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Kompleksitas-Algoritma-2020-Bagian2.pdf>   
  
1.   
A computer code with colorful text

Description automatically generated with medium confidence  
**Answer:**

The code in the top will print n in n times with for loop  
so the time complexity will be:  
T(n) = log n  
  
and the asymptotic time complexity will be:  
O(log n) = log n

So the asymptotic complexity is:  
**O(log n)**

2.   
A screenshot of a computer program

Description automatically generated

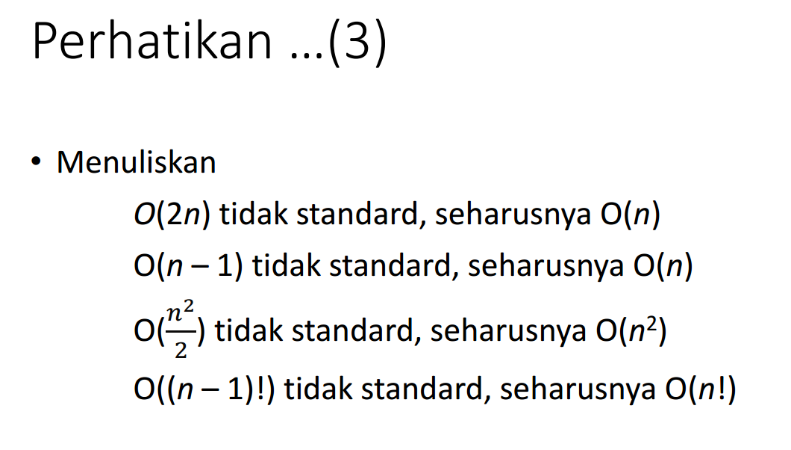
**Answer:**The code in the top will for loop n times in var i, n times in var j, n times in var k, and 10 times in var l. so the total of for loop will be n\*n\*n\*10 = 10n3. So time complexity is:  
T(n) = 10n3  
  
and the asymptotic time complexity will be:  
10n3 = O(n3)  
  
So the asymptotic time complexity is:  
**O(n3)**why the constanta isn’t included in the O(n3)? Because:  


Figure : https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Kompleksitas-Algoritma-2020-Bagian2.pdf

3.  
A computer code with colorful text

Description automatically generated

Code will be run forever because the condition is always true so:  
T(n) = ∞  
  
and asymptotic time complexity will be  
**O(∞)**