Consider the following functions:

$f(n) = n \log \log n$

$g(n) = n(\log n)^2$

Which of the following is true?

Options :

6406532239756. ✖ $f(n)$ is $O(g(n))$ and $g(n)$ is $O(f(n))$

6406532239757. ✔ $f(n)$ is $O(g(n))$, but $g(n)$ is not $O(f(n))$

6406532239758. ✖ $f(n)$ is not $O(g(n))$ and $g(n)$ is not $O(f(n))$

Consider the following functions:

- $f(n) = 102n^4 + 26n^3$
- $g(n) = 103n^3 + 20n^2$
- $h(n) = 110n^3 \log n + 36n^2$

Which of the following is/are true?

Options :

6406531929589. ✖ $f(n) = O(g(n))$

6406531929590. ✔ $g(n) = O(h(n))$

6406531929591. ✖ $f(n) = O(h(n))$

6406531929592. ✖ $h(n) = O(g(n))$

6406531929593. ✔ $h(n) = O(f(n))$

```python
def insertionsort(L):
    n = len(L)
    if n < 1:
        return(L)
    for i in range(n):
        j = i
        while(j > 0 and L[j] < L[j-1]):
            (L[j],L[j-1]) = (L[j-1],L[j])
            j = j-1
    return(L)
```

Suppose L is a list of distinct integer elements. Let x, y and z be the largest, second largest, and third largest elements in the list L. Suppose z appears before x in the list. Which of the following is true, with respect to the implementation above?

Options :

6406531929602. ✖ x and z are always compared in a run of insertion sort, regardless of the position of y.

6406531929603. ✖ x and z are compared in a run of insertion sort if and only if y appears before z in the list L.

6406531929604. ✔ x and z are compared in a run of insertion sort if and only if y appears after x in the list L.

6406531929605. ✖ x and z are compared in a run of insertion sort if and only if y appears after z but before x in the list L.

4 sorted lists each of length $n/2$ are merged into a single sorted list of $2n$ elements using two way merging. What will be the minimum number of element comparisons needed for this process ?

Options :

6406531561926. ✖ $n-1$

6406531561927. ✖ $2n-1$

6406531561928. ✔ $4n-3$