

Partie B

Question 1)

Fonction	Le pire cas	Le meilleur cas
a)	$O(n \log_2 n)$	$O(1)$
b)	$O(n^2)$	$O(1)$
c)	$O(\infty)$	$O(\infty)$
d)	$O(n * \sqrt{n} * \log_3 \sqrt{n})$	$O(1)$
e)	$O(n)$	$O(1)$

Question 2)

a)

$$\lim_{n \rightarrow \infty} \frac{\log^2 n}{\log n} = \lim_{n \rightarrow \infty} \log n * \frac{\log n}{\log n} = \lim_{n \rightarrow \infty} \log n = +\infty$$

D'après la limite **a) est vraie**

b)

$$\lim_{n \rightarrow \infty} \frac{(7n^5 + 2n^2 + 7)}{n^3} = \lim_{n \rightarrow \infty} \frac{7n^5}{n^3} = \lim_{n \rightarrow \infty} 7n^2 = +\infty$$

D'après la limite **b) est vraie**

c)

$$\lim_{n \rightarrow \infty} \frac{n(\log n)^2}{n\sqrt{n}} = \lim_{n \rightarrow \infty} \left(\frac{\log n}{\sqrt{n}} \right) * \log n \text{ or la } \lim_{n \rightarrow \infty} \frac{\log n}{n^r} = 0$$

si r est positif par conséquent la $\lim_{n \rightarrow \infty} \left(\frac{\log n}{\sqrt{n}} \right) * \log n = 0$

D'après la limite **c) est fausse**

d)

$(n^2 - n + 2)^3$ le terme qui a le degré le plus haut est n^6

$$\text{donc } \lim_{n \rightarrow \infty} \frac{(n^2 - n + 2)^3}{n^7} = \lim_{n \rightarrow \infty} \frac{n^6}{n^7} = \lim_{n \rightarrow \infty} \frac{1}{n} = 0$$

D'après la limite **d) est vraie**

Question 3)

Dans la méthode récursive notre complexité vaut $O(n^2 \log n)$ vu que nous avons utilisé la méthode « sort » de java et sa complexité valait d'après la documentation $n \log n$. Lorsque nous déroulons l'algorithme à chaque itération la complexité était de $2n \log n$. Si notre algorithme est appelé n fois dans la récursivité alors sa complexité va être égale $2n^2 * \log n$ d'où $O(n^2 \log n)$.

En comparaison avec la méthode itérative nous avons constaté que la complexité était la même. Si nous mettons une boucle « while » qui exécute n fois et que chaque exécution nous avons $2n \log n$ alors au final nous aurons $O(n^2 \log n)$.