DIU « ENSEIGNER L'INFORMATIQUE AU LYCEE »

NOTIONS DE COMPLEXITE – EXERCICES SUPPLEMENTAIRES

1. Trouver le plus petit $n_0 \in \mathbb{N}$ de telle façon que pour tout $n \geq n_0$:
   a. $8n\log(n)$ est plus petit que $2n^2$
   b. 2^n est plus grand que n^4

2. Donner une constante $x > 0$ et un entier $n \geq 1$ tels que pour tout $n \geq n_0$:
   a. $16\, n \log(n^2) \leq c \times n^2$
   b. $\frac{1}{10} 2^n \geq c \times n^4$

3. Montrer que
   a. $2n^3 + 9n^2$ est $O(n^3)$
   b. $(n \log n)/8$ est $\Omega(n \log n)$
   c. $2^{\mathrm{n}+2} - \mathrm{n}$ est $\Theta(2^{\mathrm{n}})$

4. Prouver ou réfuter que « si $d(n)$ est $O(f(n))$, alors $a \times d(n)$ est $O(f(n))$ pour toute constante $a > 0$ ».

5. Prouver ou réfuter que « si d(n) est O(f(n)) et e(n) est O(g(n)), alors d(n)-e(n) est O(f(n)-g(n))».

6. On suppose l'algorithme suivant qui calcule et retourne la multiplication de deux entiers positifs m et n.

```
def multiplier(m,n) :
   x = 0
   while n > 0 :
      if n%2 == 1:
         x = x + m
      m = 2 x m
      n = n // 2
   return x
```

   a. Compter le nombre d'opérations primitives exécutées à chaque ligne dans le pire des cas. En conclure sur la complexité en notation O.
   b. Quel est la borne supérieure de la complexité dans le pire des cas en notation Ω ? En conclure sur la complexité en notation Θ.

7. Ecrire une fonction de test de primalité d'un nombre entier plus grand que 1 passé en paramètre où le nombre de tests de multiplicité est n-2. Donner au moins deux variantes de votre fonction produisant un nombre différent d'opérations (plus petit).