Algorithmique
TD n° 1

Exercice 1 : Fonction de complexité

Calculer aussi précisément que possible la classe de complexité pour les quatre fonctions ci-dessous.

a)

```
Fonction f1(n: entier): entier;
  i,j,res: entier; res=0;
  pour i de 1 à n faire
    pour j de 1 à i faire
       res = res+i*j;
  retourne(res);
```

b)

```
Fonction f2(n: entier): entier;
  i: entier; i=0;
  si (n<=0) alors retourne(-1);
  tant que (n>1) faire
     i++; n=n/2;
  retourne(i);
```

c)

```
Fonction f3(n: entier): entier;
  si (n<=0) alors retourne(-1);
  si (n=1) alors retourne(0);
  sinon retourne(1+f3(n/2));
```

d)

```
Fonction f4(n: entier): entier;
  i,j,res: entier; res=0;
  pour i de 1 à n faire
    j=i;
    tant que (j>1) faire
        res=res+1; j=j/2;
  retourne(res);
```

Exercice 2 : Ordre de grandeur

- Dans un calcul de complexité d'une fonction f, on note souvent une expression du type $f(n) \in O(g(n))$ ou $f(n) \in \Theta(g(n))$ Que représente n dans cette expression ? Que veut dire O ? Que veut dire Θ ?
- On utilise aussi la notation Ω. Que veut dire $\Omega(g(n))$?
- Ordonner les fonctions suivantes par ordre asymptotique de grandeur. (Si $f(n) \in O(g(n))$, on notera $f << g$).
  n^2, $log_2(n)$, $n \cdot \sqrt{n}$, 3^n, $log_2(log_2(n))$, n, n^3, 2^n, $n!$, $n \cdot log_2(n)$, $\sqrt{n}$, n^n, $log_{10}(n)$
- Peut-on dire :
  - Si $f(n) \in O(g(n))$, alors $g(n) \in O(f(n))$
  - Si $f(n) \in \Omega(g(n))$, alors $g(n) \in \Omega(f(n))$
  - $n \in O(n)$, $n \in \Theta(n)$, $n^2 \in O(2^n)$
  - $n \in O(n^2)$, $n \in \Theta(n^2)$, $n^2 \in \Theta(2^n)$,
  - $n^2 \in O(n)$, $n^2 \in \Theta(n)$, $\sqrt{n} + n \in \Theta(n)$
  - $3 \cdot log_2(n) \in \Theta(log_2(n))$, $n \cdot log_2(n) \in O(n^2)$, $n + log(n) \in O(n)$
  - Si $f(n) \in \Theta(g(n))$, alors $g(n) \in \Theta(f(n))$

— Si $f(n) \in O(g(n))$, alors $g(n) \in O(f(n))$
— Si $f(n) \in O(h(n))$ et $g(n) \in O(h(n))$, alors $f(n) + g(n) \in O(h(n))$
— Si $f(n) + g(n) \in O(h(n))$, alors $f(n) \in O(h(n))$ et $g(n) \in O(h(n))$
— Si $f(n) + g(n) \in \Theta(h(n))$, alors $f(n) \in \Theta(h(n))$ et $g(n) \in \Theta(h(n))$

Exercice 3 : Découpe de chocolat

Un tablette de chocolat est composée de $n * m$ carreaux. On souhaite séparer tous les carreaux, et pour cela la seule opération dont on dispose est de prendre **un** morceau de chocolat et de le couper en deux verticalement ou horizontalement (en suivant une ligne ou une colonne).

– Proposer un algorithme pour séparer tous les carreaux. Quelle est sa complexité ? Appliquer le sur une tablette $2 * 5$.
– Quel est le nombre minimal d'opérations permettant de séparer tous les carreaux (quel que soit l'algorithme) ? Votre algorithme est-il optimal ?

Exercice 4 : Tri fusion

Donnez un algorithme de tri de deux listes (tableaux) basé sur la fusion de deux listes (tableaux) triées. L'appliquer sur la liste `[10,3,5,25,50,1]`. Est-ce qu'on peut facilement se passer de mémoire auxiliaire ?

Exercice 5 : Attrapez les tous

Il y a quelques années, le jeu Pokémon Go sur smartphone a connu un gros succès. L'un des principes de ce jeu est de se rendre physiquement à certains endroits pour aller y capturer des pokémons à l'aide de son smartphone. Imaginons que l'on puisse connaître à l'avance la position géographique (coordonnées x et y sur une carte plane) de chacun des pokémons qui nous intéressent (uniquement les 42 premiers bien évidemment !). Nous connaissons également notre position de départ x_0, y_0.

– pouvez-vous écrire un algorithme qui calcule le plus court chemin à parcourir pour tous les capturer (l'algorithme n'a pas besoin d'être efficace) et de revenir à la position de départ ? Quelle est sa complexité ?
– Pouvez-vous créer un algorithme significativement mieux ?