Université Paris 7 — Année 2003–04
Licence d'informatique — Algorithmique

TD 1

Exercice 1. Logarithmes et exponentielles

Rappels de cours

La fonction *logarithme népérien*

$$\ln : \begin{array}{ccc} \mathbf{R}_+^* & \to & \mathbf{R} \\ x & \mapsto & \ln x \end{array}$$

est par définition la primitive de $x \mapsto \frac{1}{x}$ qui s'annule en 1. C'est une bijection croissante. Pour tous x et y strictement positifs, on a : $\ln(xy) = \ln x + \ln y$.

Par définition, la fonction réciproque de ln est la fonction *exponentielle*

$$\exp : \begin{array}{ccc} \mathbf{R} & \to & \mathbf{R}_+^* \\ x & \mapsto & \exp(x) = e^x \end{array}.$$

C'est aussi une bijection croissante. On a : $\forall (x, y) \in \mathbf{R}^2$, $e^{x+y} = e^x e^y$.

Logarithmes et exponentielles en base quelconque

On définit, pour tout $x \in \mathbf{R}$, $10^x = e^{x \ln 10}$ et $\log_{10}(x) = \frac{\ln x}{\ln 10}$. On démontre les propriétés suivantes :

- les fonctions $\begin{array}{ccc} \mathbf{R} & \to & \mathbf{R}_+^* \\ x & \mapsto & 10^x \end{array}$ et $\begin{array}{ccc} \mathbf{R}_+^* & \to & \mathbf{R} \\ x & \mapsto & \log_{10}(x) \end{array}$ sont réciproques l'une de l'autre ;
- pour tous réels x et y strictement positifs, $\log_{10}(xy) = \log_{10}(x) + \log_{10}(y)$;
- pour tous réels x et y, $10^{x+y} = 10^x 10^y$.

Questions

1. Montrer que, pour tout entier positif ou nul n, 10^n vaut $\underbrace{10 \times 10 \times \ldots \times 10}_{n \text{ fois}}$. On procédera par récurrence sur n.
2. Si $\log_{10}(n) = 5$, que vaut n ?
3. Comparer 10^{100} et 100^{10}.
4. Comparer $10^{(10^{10})}$ et $(10^{10})^{10}$.

Plus généralement, pour tout $p > 0$ et pour tout $x \in \mathbf{R}$, on définit $p^x = e^{x \ln p}$ et $\log_p(x) = \frac{\ln x}{\ln p}$; les propriétés ci-dessus dans le cas $p = 10$ s'étendent naturellement.

Exercice 2. Partie entière supérieure et inférieure

Soit $x \in \mathbf{R}$.

- La *partie entière inférieure* de x, notée $\lfloor x \rfloor$, est le plus grand entier qui est inférieur ou égal à x ;
- la *partie entière supérieure* de x, notée $\lceil x \rceil$, est le plus petit entier qui est supérieur ou égal à x.

1. Calculer $\lfloor 17,43 \rfloor$; $\lceil 17,43 \rceil$; $\lfloor -3,4 \rfloor$ et $\lceil -10 \rceil$.
2. Soit x un réel ; calculer $\lfloor x \rfloor - 2\lfloor \frac{x}{2} \rfloor$. On pourra distinguer selon la parité de $\lfloor x \rfloor$.
3. Soit n un entier strictement positif. Déterminer le nombre exact de chiffres de n, en écriture décimale.

Exercice 3. Raisonnement par récurrence

1. Trouver une formule simple pour la somme
$$\sum_{i=1}^{n} i = 1 + 2 + \cdots + n$$
et prouver qu'elle est correcte.
2. Prouver par récurrence que
$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}.$$

Exercice 4. Calcul de complexité

1. Calculer $\sum_{i=0}^{n} 2^i$.
2. Calculer $\sum_{i=0}^{n} (n-i)2^i$. On pourra regrouper les puissances de 2 différemment, pour faire apparaître dans cette expression des sommes du type $\sum_{i=0}^{k} 2^i$.
3. On considère la méthode `test` qui prend en argument une suite finie $u = (u_1, u_2, \ldots, u_n)$ de 0 et de 1, et qui renvoie l'indice du premier 0 de cette suite s'il existe ; par exemple, `test(1101101)` renvoie `3`. Si la suite ne contient pas de zéro, la méthode retournera la valeur -1.

   On suppose que le temps mis pour calculer `test(u)` est :
   - la longueur n de `u`, si `u` ne contient que des 1 ;
   - l'indice du premier 0 dans `u`, sinon.

   Calculer la complexité en moyenne de `test`, sur toutes les entrées u de longueur n fixée. (Pour cela vous diviserez le total des temps de calcul pour chaque entrées, par leur nombre).