

RICM1 - 2004/05

Langage et Programmation 2

TP 1

Préliminaires : récupérer les trois fichiers sous
<http://www-verimag.imag.fr/~monin/EnseignementPublic/RICM/LP2> et
suivre les instructions du 00readme.

Exercice 1

Écrire la fonction permettant de calculer a^b .

Exercice 2

La suite de Fibonacci : Écrire la fonction qui à un entier positif associe le n^e terme de la suite de Fibonacci.

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ... jusqu'à l'infini.

Exercice 2-1

Il existe trois versions pour cette fonction (dont deux ont été expliquées en TD). Exécuter les trois fonctions avec une grande valeur de n .

Exercice 2-2

La fonction `System.time` lorsqu'elle est appliquée à la valeur `()` rend le temps total consommé par le processus. Écrire une fonction qui calcule le temps consommé par l'exécution de la fonction `fib` lorsqu'elle est appliquée à un élément donné.

Exercice 3

Compter les 'a' : Écrire une fonction qui permet de compter le nombre d'occurrences du caractère 'a' dans une chaîne de caractères.

Il faudra définir auparavant les fonctions :

- *estVide* : teste si une chaîne est vide.
- *premierCar* : renvoie le 1^{er} caractère d'une chaîne supposée non vide.
- *chaîneSansPremCar* : renvoie la chaîne sans le 1^{er} caractère.

Exercice 4

Une phrase peut cacher, dans l'ordre, les lettres d'un mot. Par exemple, la chaîne : « Examens de programmation fonctionnelle à Grenoble » contient le mot « espoir ».

Exercice 4-3

Écrire une fonction *motCache* qui teste si un mot donné est caché dans une chaîne de caractères. (On pourra utiliser les fonctions définies dans l'exercice

précédent).

Exercice 4-4

Écrire une fonction *nbMotCache* qui compte le nombre d'occurrences d'un mot caché dans une chaîne de caractères.