Algo 1.1

Travaux dirigés - Série n°6

Les consignes données en tête de la série n° 2 restent bien sûr la règle pour ce qui suit :
Titre / Variables / Corps / Commentaires

Dans les algorithmes des applications utilisant des sous-programmes, écrire d'abord un algorithme général (programme principal) de l'application, ensuite des algorithmes détaillés (sous-programmes) des procédures et/ou fonctions.

Exercice 1. Puissance

a. Ecrire l'algorithme qui permet de calculer $\mathbf{x^n}$, **x** étant un réel et **n** un entier tous deux saisis en données.

b. Ecrire l'algorithme de l'application qui permet de répéter ce traitement, avec des valeurs de **x** et **n** différentes, passées en paramètres, à la demande de l'utilisateur (en répondant à la question « voulez vous refaire le traitement ? O/N »). Dans ce cas, le calcul de la puissance d'un nombre doit être réalisée par un sous-programme. S'il s'agit d'une fonction elle doit retourner la puissance en résultat. S'il s'agit d'une procédure elle peut soit afficher le résultat soit le retourner dans un paramètre.

Exercice 2. Puissance/Factorielle

Écrire l'algorithme de l'application qui permet de répéter le calcul de l'expression $\mathbf{x^n/n\ !}$ à la demande de l'utilisateur, avec différentes valeurs de **x** et **n** saisies en données. Les calculs de $\mathbf{x^n}$ et de **n !** seront réalisés par des fonctions qui ont comme paramètre(s) les valeurs de **x** et **n** et qui retournent des valeurs correspondant aux calculs effectués.

Note : Lors du passage sur machine, ne pas utiliser de grandes valeurs de **n** afin d'éviter des dépassements de capacité dus à la représentation des données.

Exercice 3. Représentation

a. Écrire un algorithme qui permet de calculer le nombre de chiffres nécessaire pour représenter un nombre donné en base 10 en un nombre dans une base **b** saisie en donnée (Exemple b=2, 8,16).

b. Ecrire l'algorithme de l'application qui permet de répéter ce traitement à la demande de l'utilisateur. Utiliser dans ce cas une fonction qui retourne ce nombre de chiffres et qui utilise comme paramètres le nombre en base 10 et la base **b**.

Exercice 4. Fibonacci

a. Écrire un algorithme qui permet de calculer les **n** premiers termes de la suite de Fibonacci, **n** étant saisi en donnée. Les termes de la suite de Fibonacci sont obtenus en calculant la somme des deux termes précédents : $T_0=1$, $T_1=1$, $T_n=$

$T_{n-1} + T_{n-2}$ (pour n>=2).

b. Ecrire l'algorithme de l'application qui permet de calculer différentes suites de Fibonacci, ce calcul s'arrêtera à la demande de l'utilisateur. Utiliser une procédure qui permet d'afficher les **n** termes d'une suite de Fibonacci. Le nombre de termes à afficher est saisi en donnée et passer en paramètre à la procédure.