Langages de scripts (LS4)

TP 4 – Premiers pas en PYTHON

Exercice 1 – *Fonction factorielle*

À l'aide de l'interpréteur, afficher les valeurs de $n!$ pour $n \leqslant 15$, ainsi que le type de la variable contenant la valeur affichée. Écrire ensuite la version récursive de la fonction factorielle, puis sa version itérative. Modifier cette dernière fonction pour ne réaliser l'affichage que si le résultat est inférieur à 16384.

Exercice 2 – *Division euclidienne*

La division euclidienne de a par b renvoie le couple (q, r) défini par l'égalité $a = bq + r$ avec $0 \leqslant r < b$. Écrire une fonction récursive calculant le couple (q, r) **sans utiliser** `/` et `%` mais seulement $+$, $-$ et $<$.

Exercice 3 – *Table d'addition*

Écrire une fonction affichant la table d'addition de $0+0$ à $10+10$. Modifier ensuite cette fonction pour qu'elle conserve ce comportement par défaut, mais permette également d'afficher la table d'addition de $0+0$ à $m+n$. On pourra utiliser l'instruction `print "%5d" % i` qui affiche l'entier `i` dans une fenêtre de 5 caractères.

Exercice 4 – *Boucle `while` et classes*

On considère deux entiers $a = 2$ et $b = 3$ et la suite $(x_n)_n$ définie par $x_0 = 1$, $x_1 = 3$ et pour tout $n \geqslant 0$, $x_{n+2} = ax_{n+1} + bx_n$.

1. À l'aide de l'interpréteur PYTHON, affichez toutes les valeurs de $(x_n)_n$ pour $n \leqslant 20$ en utilisant une boucle `while`.
2. Écrire une fonction qui prend en entrée 5 arguments correspondant à a, b, x_0, x_1 et au rang n du terme à calculer, et renvoie la valeur de x_n.
3. Définir une classe `Suite` ayant 4 attributs correspondant à a, b, x_0 et x_1, et contenant une méthode qui prend en argument le rang n et renvoie la valeur de x_n.

Exercice 5 – *Boucle `for` et tableaux*

La fonction `r nge` utilisée avec un seul argument entier positif renvoie une liste d'entiers de la taille de cet argument et commençant en 0. On peut alors utiliser cette liste pour écrire une boucle `for`[1].

Le but de cet exercice est d'écrire une fonction renvoyant les nombres premiers inférieurs à un entier donné. Nous allons résoudre ce problème de deux façons différentes.

[1] Cela rappelle le `foreach` de PHP.

1. Écrire une fonction `est_premier()` à un argument qui vérifie si cet argument est un nombre premier en regardant s'il est divisible par un entier quelconque inférieur ou égal à sa racine carrée (fonction `sqrt` dans le module `m th`). Écrire ensuite une fonction qui affiche les nombres premiers inférieurs à un entier donné (qui sera paramètre de cette fonction).

2. On explore maintenant une autre méthode, appelée *crible d'Eratosthène*. Son principe algorithmique est joliment illustré sur le site `perso.orange.fr/therese. eveilleau/ pages/truc_mat/pratique/textes/crible_an.htm` (première référence donnée par Google). Pour programmer cet algorithme, on considère une liste de taille $n+1$ contenant des 0 (pour cela, utilisez `[0]*(n+1)`), la case `i` correspondant à l'entier `i`, la première case (case `0`) étant ignorée. Quand un nombre est retenu, on place un 1 dans la case correspondante, quand il est éliminé, un -1. À la fin, il suffit de regarder quelles cases sont à 1. En suivant cette méthode, écrire une fonction qui affiche les nombres premiers inférieurs à un entier donné (qui sera paramètre de cette fonction). On pourra écrire une fonction intermédiaire pour la phase d'élimination des multiples d'un entier.