Factorielle : une fonction en Python

Voici une version avec la fonction factorielle() <sxh python; title : factorielle04-fonction_1.py> #! /usr/bin/env python # -*- coding: utf-8 -*- “”“ Calcul de la factorielle d'un nombre Référence : http://fr.wikipedia.org/wiki/Factorielle ”“” def factorielle(arg_n):

```
  # structure de répétition pour appliquer la définition de la factorielle
  reponse=1  # la réponse sera dans la variable reponse
  i=1  # on va commencer par 1
  while i <= arg_n:  # répétition "while" avec une condition à préciser
      reponse = reponse*i    #actualisation de reponse
      i=i+1                 #incrémenter i
  return reponse
```

on demande le nombre : print “Calcul de la factorielle de n” chainelue=raw_input(“Que vaut n ? ”) n= int(chainelue) print n

on affiche la réponse print “La factorielle vaut ”,factorielle(n) </sxh>

On aurait pu utiliser une autre structure de répétition que “while”, comme le “for” en utilisant une liste de facteurs à l'aide de “range”. Cette dernière fonction doit utiliser comme première valeur incluse (start) 1, et comme dernière valeur exclue (stop) l'argument de la factorielle augmenté de 1. Voici cette variante de la fonction :

<sxh python> def factorielle2(arg_n):

```
  # structure de répétition pour appliquer la définition de la factorielle
  reponse=1  # la réponse sera dans la variable reponse
  for i in range(1, arg_n + 1):  # répétition "for" avec "range"
      reponse = reponse*i    # actualisation de reponse
  return reponse
```

</sxh>

Il est aussi possible d'utiliser une variante de “while” en progressant du facteur le plus grand jusque “1” (en fait même “2” puisque 1 est l'élément neutre de la multiplication. Voici cette variante de la fonction :

<sxh python> def factorielle3(arg_n):

```
  # structure de répétition pour appliquer la définition de la factorielle
  reponse=1  # la réponse sera dans la variable reponse
  while arg_n > 1:  # répétition "while"
      reponse = reponse*arg_n    # actualisation de reponse
      arg_n = arg_n -1              # décrémenter arg_n
  return reponse
```

</sxh>

La factorielle étant une fonction courante en mathématique, elle est bien sûr intégrée au module

“math”, appelable par l'instruction “import math”. On peut vérifier en mode console de Python via la commande “help(math)” que la fonction “factorial” est effectivement présente. On pourra donc facilement calculer la factorielle d'un nombre naturel, et vous pourrez tester que sur un argument négatif ou non-entier, math.factorial() renvoie un message d'erreur.

Il existe une autre approche de la programmation de la factorielle, basée sur sa définition récursive : $n!=(n-1)! n$. On peut définir la fonction de telle sorte qu'elle s'appelle elle-même avec un argument décrémenté de une unité. Reste à gérer la définition de la factorielle de zéro afin que l'invocation de la factorielle par elle-même ne se fasse pas de manière infinie. Du fait que Python est un langage interprété, une limitation apparaît cependant dans l'implémentation récursive, du fait de la gestion nécessaire d'une pile d'informations des appels successifs. Voici cette variante de la fonction :

<sxh python> def factorielle4(arg_n):

```
# écriture récursive pour appliquer la définition de la factorielle
if arg_n == 0:
    return 1
else:
    return factorielle4(arg_n - 1) * arg_n
```

</sxh>

> Exercice : rassembler toutes les fonctions dans un même programme (elles doivent figurer en tête, avant les instructions “principales”), et prévoir l'importation du module math. Imprimer les factorielles calculées à l'aide des différentes possibilités.

Pour des travaux additionnels sur le sujet, allez à la page suivante !