Contrôle d'informatique

Durée : 1 heure

Ce contrôle est constitué de trois exercices indépendants.

Exercice 1 Autour de la suite de Fibonacci

La suite de Fibonacci est définie par les relations $f_0 = 0$, $f_1 = 1$ et pour tout $n \in \mathbb{N}$: $f_{n+2} = f_n + f_{n+1}$.

Question 1. Écrire en Python la définition d'une fonction `f(n)` réalisant le calcul de f_n de façon itérative, en calculant de proche en proche les f_k pour $k \leqslant n$, et en ne stockant que « quelques » termes de la suite.

On considère maintenant la suite g définie par $g_0 = 0$, $g_1 = g_2 = 1$ et pour tout $n \in \mathbb{N}$: $g_{n+3} = g_n + g_{n+2}$.

Question 2. Écrire en Python une fonction `g(n)` qui prend en argument un entier n et qui retourne la valeur de g_n. On écrira un programme itératif calculant les différents termes de proche en proche.

On considère maintenant, pour $m \geqslant 3$ fixé, la suite h de premiers termes $h_0 = 0$ et $h_1 = h_2 = \cdots = h_{m-1} = 1$, et vérifiant la relation de récurrence : $h_{n+m} = h_n + h_{n+m-1}$ pour tout $n \in \mathbb{N}$.

Question 3. Rédiger en Python une fonction itérative `h1(m, n)` prenant en arguments les entiers m et n et retournant la valeur de h_n. On utilisera un tableau de $n+1$ cases pour stocker les valeurs successives de la suite h.

Rappel sur les tableaux. Si `n` est une variable de type *int*, l'instruction `t = [None] * n` crée un tableau de n cases indexées entre 0 et $n-1$. Si $k \in [\![0, n-1]\!]$, l'instruction « `t[k] = a` » permet d'affecter à la $(k+1)^{\text{e}}$ case de ce tableau la valeur a et l'instruction « `t[k]` » d'utiliser cette valeur dans un calcul.

Question 4. Écrire enfin une fonction `h2(m, n)` retournant toujours la valeur de h_n mais utilisant cette fois un tableau de m cases.

Exercice 2 Nombres heureux

Un entier naturel est dit *heureux* lorsqu'en faisant la somme des carrés de ses chiffres en base 10 puis en réitérant ce procédé on finit par aboutir à 1. Dans le cas contraire il est dit *malheureux*.
Par exemple, 7 est un nombre heureux puisque la suite qui lui est associée est :

$$t_0 = 7, \quad t_1 = 7^2 = 49, \quad t_2 = 4^2 + 9^2 = 97, \quad t_3 = 9^2 + 7^2 = 130, \quad t_4 = 1^2 + 3^2 + 0^2 = 10, \quad t_5 = 1^2 + 0^2 = 1.$$

En revanche, 8 est un nombre malheureux puisque la suite qui lui est associée est :

$$\begin{array}{lllll} t_0 = 8, & t_1 = 8^2 = 64, & t_2 = 6^2 + 4^2 = 52, & t_3 = 5^2 + 2^2 = 29, & t_4 = 2^2 + 9^2 = 85, \\ t_5 = 8^2 + 5^2 = 89, & t_6 = 8^2 + 9^2 = 145, & t_7 = 1^2 + 4^2 + 5^2 = 42, & t_8 = 4^2 + 2^2 = 20, & t_9 = 2^2 + 0^2 = 4, \\ t_{10} = 4^2 = 16, & t_{11} = 1^2 + 6^2 = 37, & t_{12} = 3^2 + 7^2 = 58, & t_{13} = 5^2 + 8^2 = 89, & t_{14} = \cdots \end{array}$$

et il n'est pas nécessaire de poursuivre le calcul puisque le nombre 89 a déjà été obtenu ; la séquence 89, 145, 42, 20, 4, 16, 37, 58 va se répéter indéfiniment.
En fait, il est possible de prouver (*et nous admettrons ce résultat*) qu'un nombre est malheureux si et seulement s'il atteint l'un des nombres de la séquence exposée ci-dessus.

Question 5. Rédiger en Python une fonction `somme_carre(n)` qui prend en argument un entier naturel n et qui retourne la somme des carrés de ses chiffres en base 10.

Question 6. Rédiger alors une fonction `heureux(n)` qui prend en argument un entier naturel n et qui retourne la valeur booléenne `True` lorsque n est un nombre heureux, et `False` sinon.

Exercice 3 Recherche de facteurs carrés dans une chaîne de caractères

On dit qu'une chaîne de caractères c est un carré s'il existe une chaîne de caractères u telle que $c = uu$. Par exemple, `'papa'` est un carré mais pas `'maman'`.

Question 7. Rédiger une fonction `est_un_carre(c)` qui prend en argument une chaîne de caractères c et qui retourne le booléen `True` si c est un carré et `False` sinon.
Lorsque c est une chaîne de caractères de longueur n, combien de comparaisons entre caractères individuels sont nécessaires dans le pire des cas pour déterminer si c est un carré ?

On dit qu'une chaîne de caractères s contient un facteur carré s'il existe un carré *non vide* c et deux chaînes u et v (éventuellement vides) telle que $s = ucv$. Par exemple, `'maman'` contient un facteur carré puisque $u =$ `''`, $c =$ `'mama'` et $v =$ `'n'` conviennent.

Question 8. Rédiger une fonction `contient_un_carre(s)` qui prend en argument une chaîne de caractères s et qui retourne le booléen `True` si s contient un carré, et `False` sinon.
Lorsque s est une chaîne de caractères de longueur n, majorer en fonction de n le nombre de comparaisons entre caractères individuels réalisé par votre algorithme pour déterminer si s contient un carré.

♦ ♦
♦