DEVOIR SURVEILLÉ N°1

Samedi 13 octobre — 2 heures

On a le droit d'utiliser, dans une question, les programmes des questions antérieures, *même si* l'on n'a pas réussi à les écrire. Les trois exercices sont indépendants.

Et pour ceux qui lisent les préambules, il y a un piège dans l'avant-dernière question.

Exercice I — Écriture scientifique des réels

Dans cet exercice on pourra utiliser (ou pas) les fonctions habituelles sur les flottants.

```
from numpy import inf, exp, log as ln, floor, ceil, sqrt
```

Tout réel x différent de zéro peut s'écrire

$$x = s \times m \times 10^e$$

où $s = \pm 1$ (le *signe*), $m \in [1\,;10[$ (la *mantisse*) et $e \in \mathbf{Z}$ (l'*exposant*) : c'est *l'écriture scientifique.*

QUESTION 1.1 — *Écrire le programme* `Signe(x)` *qui étant donné le réel non nul x calcule et renvoie son signe s.*

QUESTION 1.2 — *Écrire le programme* `MantisseEtExposant(x)` *qui étant donné un réel $x > 0$ calcule et renvoie le couple* `(m, e)`*. Insistons : m est lui-même un réel, et e est un entier.*

La partie entière $\lfloor m \rfloor$ donne le premier chiffre de la mantisse. Notons m_1, m_2, m_3, … ceux qui apparaissent après la virgule, dans cet ordre. Si x est décimal, ils sont tous nuls à partir d'un certain rang.

QUESTION 1.3 — *Écrire le programme* `Chiffre(m, k)` *qui étant donné un entier $k \geqslant 1$ calcule et renvoie l'entier m_k (qui est donc entre 0 et 9). On suppose que k n'est pas très grand et qu'on peut négliger tous les problèmes d'arrondi.*

Exercice II — Mille et mille un, par exemple

On dit qu'un entier naturel x est *sans facteur carré* s'il n'existe aucun $k \geqslant 2$ tel que k^2 divise x.

QUESTION 2.1 — *Écrire un programme* `EstSansFacteurCarré(x)` *qui teste si x est sans facteur carré, et renvoie* `True` *ou* `False` *en conséquence.*

QUESTION 2.2

a) *Écrire un programme* `PlusPetitFacteur(x)` *qui étant donné $x \geqslant 2$ renvoie le plus petit nombre $p \geqslant 2$ qui divise x.*

b) *Justifier que ce nombre p est un nombre premier. En particulier, si $p = x$, alors x est premier.*

c) *En déduire un programme* `NombreFacteurs(x)` *qui étant donné $x \geqslant 1$ calcule et renvoie le nombre r dans la factorisation*

$$x = p_1 p_2 \dots p_r$$

de x en produit de nombres premiers (distincts ou pas).

On note $\mathscr{E}$ l'ensemble des entiers naturels qui sont produit d'exactement trois nombres premiers distincts. Le plus petit élément de $\mathscr{E}$ est $2 \times 3 \times 5 = 30$, et son plus petit élément impair est $3 \times 5 \times 7 = 105$.

QUESTION 2.3 — *Écrire un programme* `EstDansE(x)` *qui étant donné un entier naturel x détermine s'il est dans $\mathscr{E}$ et renvoie* `True` *ou* `False` *pour le dire.*

QUESTION 2.4 — *Écrire finalement un programme* `Prochain(n_min)` *qui détermine le plus petit entier $n \geqslant n_{min}$ tel que $n^3 + 1 \in \mathscr{E}$.*

```
>>> Prochain(0)
9

>>> Prochain(9 + 1)
10

>>> Prochain(10 + 1)
12
```

```
>>> Prochain(12 + 1)
13

>>> Prochain(13 + 1)
21

>>> Prochain(21 + 1)
25
```

```
>>> Prochain(21 + 1)
30

>>> Prochain(30 + 1)
34

>>> Prochain(34 + 1)
36
```

Exercice III — Triangles emboîtés

On considère trois nombres complexes a_0, b_0 et c_0, et on définit les suites $(a_n)_{n\geqslant 0}$, $(b_n)_{n\geqslant 0}$ et $(c_n)_{n\geqslant 0}$ par les relations de récurrence

$$a_{n+1} = \frac{a_n + b_n}{2}, \quad b_{n+1} = \frac{b_n + c_n}{2} \quad \text{et} \quad c_{n+1} = \frac{c_n + a_n}{2}.$$

QUESTION 3.1 — *Écrire un programme* `Triangle(a0, b0, c0, n)` *qui calcule et renvoie le triplet (a_n, b_n, c_n).*

Rappelons que Python calcule avec les nombres complexes aussi bien qu'avec les nombres réels. Le nombre i tel que $i^2 = -1$ est noté `1j` ; si `z` est un nombre complexe on écrit `z.real` et `z.imag` pour avoir sa partie réelle et sa partie imaginaire, `abs(z)` pour avoir son module. Enfin, avec

```
from numpy import exp, pi as π
```

on peut calculer des exponentielles complexes.

```
>>> exp(1j * π)
(-1 + 1.2246467991473532e-16j)
```

QUESTION 3.2 — *On donne le programme ci-dessous. Expliquer son intérêt lorsqu'on travaille avec des flottants.*

```
def EstNul(x) :
    return abs(x) < 1e-12
```

Une droite de $\mathbf{C}$ peut être entièrement décrite par deux nombres complexes m (un point de référence) et $n \neq 0$ (un vecteur directeur) : la droite est alors

$$\mathscr{D}(m, n) = \{z = m + \lambda n \mid \lambda \in \mathbf{R}\}.$$

On voit alors que $z \in \mathscr{D}(m, n) \Leftrightarrow \mathrm{im}((z-m)/n) = 0$.

QUESTION 3.3

a) *On suppose que* n_1 *et* n_2 *sont deux nombres complexes non nuls, et que les vecteurs d'affixes* n_1 *et* n_2 *ne sont pas colinéaires. Justifier que* $\mathrm{im}(n_1/n_2) \neq 0$.

b) *Toujours sous les mêmes hypothèses, et en ajoutant deux complexes quelconques* m_1 *et* m_2*, démontrer que l'unique intersection des droites* $\mathscr{D}(m_1, n_1)$ *et* $\mathscr{D}(m_2, n_2)$ *a pour affixe*

$$z = m_2 + \lambda n_2 \quad \textit{avec} \quad \lambda = -\frac{\mathrm{im}\left(\dfrac{m_1 - m_2}{n_2}\right)}{\mathrm{im}\left(\dfrac{n_1}{n_2}\right)}$$

et écrire le programme `Intersection(m1, n1, m2, n2)` *qui renvoie ce nombre* z.

QUESTION 3.4 — *Soient* a *et* b *deux nombres complexes distincts. On définit*

$$m = \frac{a+b}{2} \quad \textit{et} \quad n = e^{i\pi/2} \times \frac{b-a}{|b-a|}.$$

Que dire de la droite $\mathscr{D}(m, n)$ *(on justifiera la réponse) ? Écrire le programme* `M_d__tr__e(a, b)` *qui renvoie ce couple* (m, n).

QUESTION 3.5 — *On suppose que les affixes* a*,* b *et* c *correspondent à trois points distincts et non alignés. Écrire le programme* `CentreDuCercleCirconscrit(a, b, c)` *qui calcule et renvoie ce qu'on imagine.*

QUESTION 3.6

a) *Écrire le programme* `PetiteCouverture(a, b, c)` *qui étant données les trois affixes des sommets d'un triangle non aplati calcule et renvoie le couple* (z_Ω, R) *correspondant au plus petit disque (l'affixe de son centre, et son rayon) couvrant totalement ce triangle.*

b) *On suppose que* a_0*,* b_0 *et* c_0 *sont les affixes des sommets d'un triangle non aplati. Écrire le programme* `Éloignement(a0, b0, c0, n)` *qui calcule le rayon* R *du plus petit disque couvrant le triangle dont les sommets ont pour affixes* a_n*,* b_n *et* c_n.

```
>>> Éloignement(1, 1j, -1j, 0)
1.0

>>> Éloignement(1, 1j, -1j, 1)
0.5

>>> Éloignement(1, 1j, -1j, 10)
0.0009765625
```

FIN