Algorithmes de calcul approché de $\sqrt{3}$

1. **Algorithme de balayage**
   L'algorithme peut se décrire ainsi :

Algorithme 0.1: Calcul de $\sqrt{3}$ par balayage()

variables entières p
variables réelles r
$r \leftarrow 1$
pour $p \leftarrow 1$ **jusqu'à** 10
faire
tant que $r^2 < 3$
$\{r \leftarrow r + 10^{-p}$
$r \leftarrow r - 10^{-p}$
Afficher (r)

La traduction en Python peut ressembler à ceci :

```
r=1
for p in range(10):
    while(r*r<3):
        r=r+10**(-p)
    r=r-10**(-p)
    print(r)
```

Tester ce programme, et vérifier que les valeurs approchées de $\sqrt{3}$ données sont toutes par défaut (des troncatures).

Pour améliorer son comportement, on peut utiliser les nombres décimaux de Python : `getcontext.prec()` désigne le nombre de chiffres voulus. On peut le fixer à 101 pour avoir 100 décimales de $\sqrt{3}$ avec l'algorithme précédent :

```
from decimal import *
getcontext().prec=101
r=Decimal(1)
for p in range(101):
    while(r*r<Decimal(3)):
        r=r+Decimal(10)**(-p)
    r=r-Decimal(10)**(-p)

print(r)
```

2. **Algorithme de Heron d'Alexandrie**

(a) Simplifier l'expression $\dfrac{3}{\sqrt{3}}$ en utilisant le fait que $3 = \sqrt{3} \times \sqrt{3}$:

$$\frac{3}{\sqrt{3}} = \text{.....}$$

(b) On admet que le tableau de variations de $x \mapsto \dfrac{3}{x}$ est le suivant sur $[1; +\infty[$:

x	1	$+\infty$
$\dfrac{3}{x}$	$\searrow$	

Alors si x est une valeur approchée de $\sqrt{3}$ par défaut (soit $x \leqslant \sqrt{3}$), que peut-on dire de $\dfrac{3}{x}$? .....

(c) Réciproquement, $x = 2$ est une valeur approchée par excès de $\sqrt{3}$ (parce que $2^2 = 4 > 3$) ; que peut-on alors dire de $\dfrac{3}{x} = \dfrac{3}{2}$? (on comparera son carré $\left(\dfrac{3}{2}\right)^2 = \dfrac{\ldots.}{\ldots.}$ avec 3) : .....

(d) Calculer la moyenne de 2 et $\dfrac{3}{2}$; on admettra que c'est une meilleure valeur approchée de $\sqrt{3}$ que 2 et que $\dfrac{3}{2}$:.....

L'algorithme de Heron d'Alexandrie consiste à partir d'une valeur approchée x de $\sqrt{3}$, puis à la remplacer répétitivement par sa moyenne avec $\dfrac{3}{x}$:

```
x=1
for n in range(10):
    x=(x+3/x)/2
    print(x)
```

Au bout de combien d'itérations a-t-on 10 décimales correctes ?.....

Toutes les approximations de $\sqrt{3}$ ainsi obtenues sont des fractions :

```
from fractions import *
x=Fraction(1)
for n in range(8):
    x=(x+3/x)/2
    print(x)
```

3. **Questions non notées**

L'algorithme de Heron est suffisamment rapide pour calculer presque instantanément 100 décimales de $\sqrt{3}$:

```
from decimal import *
x=Decimal(1)
for n in range(10):
    x=(x+Decimal(3)/x)/2

print(x)
```

Vérifier que l'algorithme de Heron peut calculer 1000 décimales de $\sqrt{3}$ en moins d'une seconde, ce qui n'est pas le cas avec l'algorithme par balayage.