Un corrigé du devoir maison n°3

On appelle *suite de Syracuse* une suite (u_n) d'entiers naturels définie de la manière suivante :

- le premier terme u_0 est un entier naturel non nul que l'on pourra choisir;
- pour tout entier $n \geq 0$:

$$\begin{cases} u_{n+1} = \frac{u_n}{2} \text{ si } u_n \text{ est pair} \\ u_{n+1} = 3u_n + 1 \text{ si } u_n \text{ est impair} \end{cases}$$

1. (a) Que remarque-t-on lorsqu'on calcule les premiers termes de la suite de Syracuse en prenant $u_0 = 1$?

- $u_0 = 1$;
- $u_1 = 3 \times u_0 + 1 = 3 + 1 = 4$ car u_0 est impair;
- $u_2 = \frac{u_1}{2} = 2$ car u_1 est pair;
- $u_3 = \frac{u_2}{2} = 1$ car u_2 est pair;
- $u_4 = 3 \times u_3 + 1 = 4$;
- $u_5 = \frac{u_4}{2} = 2$;
- $u_6 = \frac{u_5}{2} = 1$;
- Etc.

On remarque que les valeurs de la suite se répètent, bouclent, etc. En mathématiques on a inventé un terme : la suite est *périodique*, de période 3 (car 3 termes se répètent) et ce sera le cas pour toutes les suites de Syracuse si jamais l'un des termes vaut 1 : à partir de ce terme, elle sera périodique.

- (b) Le nombre 1 figure-t-il dans la suite de Syracuse lorsque $u_0 = 10$?

$u_0 = 10, u_1 = 5, u_2 = 16, u_3 = 8, u_4 = 4, u_5 = 2, u_6 = 1$ donc le nombre 1 figure bien dans la suite de Syracuse de premier terme $u_0 = 10$.

La conjecture de Syracuse, non démontrée à ce jour, s'énonce ainsi : « Quel que soit l'entier naturel choisi pour u_0 , le nombre 1 est atteint par un terme de la suite »

Par ailleurs, l'observation de la suite montre que la suite peut s'élever assez haut avant de retomber. Les graphiques font penser à la chute chaotique d'un grêlon ou bien à la trajectoire d'une feuille emportée par le vent. De cette observation est né tout un vocabulaire imagé : on parle du *vol* de la suite et on définit alors :

le temps de vol : c'est le plus petit indice de n tel que $u_n = 1$;

le temps de vol en altitude : c'est le plus petit indice n tel que $u_{n+1} < u_0$;

l'altitude maximale : c'est la valeur maximale de la suite.

2. L'objectif est de créer un algorithme permettant de déterminer le temps de vol de la suite.

- (a) Compléter le programme en Python ci-dessous.

```
def Syracuse(u) :  
    if u%2 == 0 :  
        u = u//2  
    else :  
        u = 3 * u + 1  
    return u
```

Remarque. Comme la division par 2 ne se fait que lorsque u est pair, le reste de la division euclidienne est nul, aussi j'ai choisi le signe de double division qui donne le quotient *entier* de la division (euclidienne); cela permet d'éviter de transformer les termes de la suite en des flottants qui sont manipulés avec moins de précision par Python. Les résultats obtenus seront de la forme « 2 » plutôt que de la forme « 2.0 ».

```
def Liste_Syracuse(u):
```

```
    L=[u]
```

```
    while u != 1:
```

```
        L.append(Syracuse(u))
```

```
        u = Syracuse(u)
```

```
    return L
```

(b) Quel nombre est renvoyé par $Syracuse(6)$? $Syracuse(7)$?

Je n'attendais pas de justification mais j'en donne dans le corrigé.

$Syracuse(6)$ renvoie 3 (comme $u = 6$ est pair, le reste de la division de 6 par 2 est nul donc u prend la valeur $6//2 = 3$).

$Syracuse(7)$ renvoie 22 (comme $u = 7$ est impair, le reste de la division de 7 par 2 n'est pas nul donc u prend la valeur $3 * 7 + 1 = 22$).

(c) À quel test correspond $u\%2 == 0$?

$u\%2$ renvoie le reste de la division de u par 2, le test $u\%2 == 0$ est donc vrai si u est pair et faux si u est impair. Cela revient à tester si u est pair.

(d) Quel est le rôle de la fonction $Syracuse$?

Beaucoup de réponses de votre part ont négligé de ce ramener au contexte de l'exercice et donc à la suite de Syracuse. C'était pourtant ce qui était attendu.

Lorsque $u = u_n$, $Syracuse(u)$ renvoie u_{n+1} .

3. (a) Que retourne l'instruction :

$Liste_Syracuse(14)$?

$Liste_Syracuse(14)$ renvoie $[14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1]$.

(b) Que fait l'instruction :

$L.append(Syracuse(u))$?

Là encore beaucoup de réponses n'étaient pas contextualisées. C'était pourtant ce qui était attendu.

Cette instruction ajoute u_{n+1} à la liste L constituée des termes de la suite $[u_0, u_1, \dots, u_n]$.

(c) Est-on sûr que pour tout entier naturel u non nul, $Liste_Syracuse(u)$ renvoie une liste?

La boucle « while » est ce qu'on appelle une boucle non bornée : elle ne se termine que si le test $u \neq 1$ devient faux, c'est-à-dire que si $u = 1$. Cette fonction ne renvoie une liste que si 1 apparaît dans les termes de la suite. Or, comme l'indique l'énoncé, à ce jour on n'a pas trouvé de terme u_0 tel que 1 n'apparaît pas; mais cela n'a pas été démontré et cela reste une conjecture. Il est possible qu'un jour un mathématicien trouve un u_0 tel que 1 n'apparaisse pas. La réponse est donc qu'on n'est pas sûr qu'un 1 apparaisse, et donc on n'est pas sûr que la boucle s'arrête, et donc on n'est pas sûr que l'instruction renvoie une liste.

4. On rappelle que $len(L)$ renvoie la longueur de la liste L et $max(L)$ renvoie la plus grande valeur de la liste L .

(a) Si $u_0 = 7$, quelle est le temps de vol de la suite et quelle est son altitude maximale?

On attendait ici une justification pour chacune des réponses.

- $len(Liste_Syracuse(u))$ renvoie 17 donc la liste est de taille 17 mais comme la suite commence à u_0 et qu'elle est de taille 17, alors $L = [u_0, u_1, \dots, u_{16}]$. C'est donc pour $n = 16$ que u_n vaut 1 pour la première fois (avant que la période 1,4,2 ne commence) et donc son temps de vol est 16.
- $max(Liste_Syracuse(u))$ renvoie 52 donc son altitude maximale est 52.

- (b) Écrire un programme qui détermine la plus petite valeur u_0 qui donne un temps de vol supérieur à 100.

On a vu dans la question précédente que $len(Liste_Syracuse(u))$ renvoie le temps de vol +1 ce qui est équivalent à dire que le temps de vol est $len(Liste_Syracuse(u)) - 1$.

L'idée de la fonction ci-dessous est donc de tester avec $u_0 = 1$ puis $u_0 = 2$, etc. si le temps de vol de la suite débutant à u_0 est supérieur à 100.

Cela donne, par exemple :

```
def SuperieurACent():  
    u=1  
    while len(Liste_Syracuse(u)) - 1 < 100 :  
        u = u + 1  
    return u
```

Remarque. Cette fonction se lance sans paramètre. Certain-e-s ont choisi de mettre le seuil de 100 en paramètre, la fonction devant alors :

```
def SuperieurACent(s):  
    u=1  
    while len(Liste_Syracuse(u)) - 1 < s :  
        u = u + 1  
    return u
```

Mais iels n'ont pas du tout expliqué le rôle de ce s . Si je peux deviner qu'en la lançant avec $s = 100$, la fonction reverra le bon résultat, je suis obligé de sanctionner la réponse comme incomplète : ce n'est pas au correcteur de deviner comment fonctionne la fonction que vous proposez.

- (c) Quelle est alors l'altitude de ce vol?

Le programme précédent renvoie 27, donc la plus petite valeur de u_0 telle que le temps de vol est supérieur à 100 est $u_0 = 27$.

$max(Liste_Syracuse(27))$ renvoie 9 232 donc son altitude maximale est 9 232.

Remarques.

- On attendait des justifications.
- On peut vérifier que son temps de vol est supérieur à 100 avec $len(Liste_Syracuse(u)) - 1$ qui vaut 111 mais cela n'était pas nécessaire car cela ne prouve pas que pour les valeurs de u_0 inférieures à 27 le temps de vol est bien inférieur à 100.
- Dans le cas où vous avez mis un paramètre s , il faut ici expliquer que vous lancez la fonction `SuperieurACent(100)` sinon vous êtes aussi sanctionné-e

5. Écrire un algorithme permettant d'obtenir le temps de vol en altitude pour une valeur donnée de u_0 .

La fonction ci-dessous est un exemple de fonction donnant le temps de vol en altitude.

```
def TempsDeVolEnAltitude(u):  
    valeur_initiale = u  
    n = 0  
    u = Liste_Syracuse(u)  
    while u >= valeur_initiale:  
        u = Liste_Syracuse(u)  
        n = n + 1  
    return n
```

Remarque. Soit T le temps de vol en altitude. Beaucoup de fonctions proposées en réponse à cette question donnaient $T + 2$. Les raisons ayant mené à cette erreur sont variées mais la principale est la suivante : certain-e-s ont oublié que la numérotation de la suite commençait à 0 et

que si, par exemple, c'est au 4^e terme que la suite repasse sous sa valeur initiale, alors le temps de vol en altitude est 2, puisque le dernier terme au-dessus de la valeur initiale est le 3^e et donc c'est u_2 .

Dans tous les cas, il vous faut tester vos fonctions avec des essais pour voir si votre fonction donne le bon résultat.

Par exemple $Liste_Syracuse(5)$ renvoie $[5, 16, 8, 4, 2, 1]$. Le vol reste en altitude pour $u_1 = 16$, $u_2 = 8$ et passe en dessous de $u_0 = 5$ pour $u_3 = 4$. Le temps de vol en altitude est le dernier indice n tel que $u_n > 5$, c'est donc, ici, 2. Si votre fonction ne renvoie pas 2 pour le temps de vol en altitude dans le cas où $u_0 = 5$, c'est qu'elle n'est pas correcte.