Test de primalité

Le problème

Il est important en arithmétique de savoir si un nombre entier est premier ou s'il ne l'est pas. Pour ce faire, on utilise ce que l'on appelle un « test de primalité ».

On convient que 1 n'est pas premier. Un entier supérieur ou égal à 2 est premier si et seulement s'il n'admet aucun diviseur autre que 1 et lui-même. On peut montrer qu'il sera premier si et seulement s'il n'admet aucun diviseur premier inférieur ou égal à sa racine carrée. Pour savoir si un nombre est premier ou pas, il suffit donc de tester sa divisibilité par chacun des entiers premiers inférieurs ou égaux à sa racine carrée.

Complément culturel : le crible d'Ératosthène

On attribue au mathématicien grec Ératosthène (vers – 276 ; vers – 194) l'idée de réaliser un *crible* pour identifier les nombres premiers inférieurs à un entier E. Pour cela, il a écrit tous les entiers de 2 à E. Le plus petit est 2. Il l'a entouré (il est premier) et a rayé ses multiples. Le plus petit nombre non rayé est 3. Il l'a entouré (il est premier) et a rayé ses multiples. Le plus petit nombre non rayé est alors 5. Il l'a entouré (il est premier) et a rayé ses multiples. Et ainsi de suite…

Les programmes

En testant la divisibilité de N par chaque entier 2, 3, 5, 7, **9**, 11, 13, **15**, 17, 19, **21**, 23, **25**, **27**… sans dépasser la racine carrée de N, il est vrai que nous faisons faire à l'ordinateur un peu trop de travail, car parmi ces entiers beaucoup ne sont pas des nombres premiers (c'est le cas de ceux que nous avons fait figurer en caractères gras dans la liste ci-dessus). Mais cela évite un programme *récursif*, un peu difficile à ce stade.

le programme

```
>>> from math import sqrt
>>> def primalité(N):
        d=2
        r=N%d
        if r==0:
                return (N,'non premier')
        d=3
        while d<=sqrt(N):
                r=N%d
                if r==0:
                        return(N,'non premier')
                d=d+2
        return (N,'premier')
```

le résultat

```
>>> primalité(2011)
(2011, 'premier')
```

Eh oui, 2 011 est un nombre premier !

Scratch

Le crible d'Ératosthène nous permet de comprendre que 2 est le seul entier à la fois premier et pair. À partir de 3, les entiers premiers sont tous impairs. Dans les programmes, on est parti d'un entier $N > 2$, on a testé la division de l'entier N par 2, puis par tous les entiers impairs d inférieurs ou égaux à $\sqrt{N}$. À la fin du programme, on aura donc bien testé la divisibilité de N par tous les entiers premiers inférieurs ou égaux à $\sqrt{N}$.

avec AlgoBox

La fonction floor() est la fonction *partie entière*. Le test floor(N/d)==N/d permet de savoir si N est divisible (ou non) par d. Si le nombre N testé n'est pas premier, le programme renverra la phrase « N non premier », puis « Programme terminé. Arrêter le programme ». Il suffit dans ce cas de cliquer sur le bouton rouge d'arrêt du programme. Attention à bien tester **uniquement des entiers supérieurs ou égaux à 3**.

```
VARIABLES
    N EST_DU_TYPE NOMBRE
    d EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
    LIRE N
    SI (floor(N/2)==N/2) ALORS
        DEBUT_SI
        AFFICHER "N non premier"
        AFFICHER "Programme terminé.  Arrêter
        PAUSE
        FIN_SI
    d PREND_LA_VALEUR 3
    TANT_QUE (d*d<=N) FAIRE
        DEBUT_TANT_QUE
        SI (floor(N/d)==N/d) ALORS
            DEBUT_SI
            AFFICHER "N non premier"
            AFFICHER "Programme terminé. Arr
            FIN_SI
        d PREND_LA_VALEUR d+2
        FIN_TANT_QUE
    AFFICHER "N est premier"
FIN_ALGORITHME
```

Prolongement*

Au début du XVII^{e} siècle, Pierre de Fermat énonce son « petit théorème » : *« Si un entier N est premier, alors, pour tout entier a non multiple de N, le reste de la division euclidienne de a^{N-1} par N vaut 1. »* La réciproque de ce théorème est fausse : si l'on prend l'entier 1 729, le reste de la division de $2^{1\,728}$ par 1 729 vaut 1, et pourtant 1 729 n'est pas premier ($1\,729 = 7 \times 13 \times 19$). On peut donc construire, à partir de cette propriété, un test qui élimine certains nombres non premiers mais qui ne garantit pas à 100 % qu'un nombre est premier.

Nous vous laissons programmer ce test qui pourrait afficher, par exemple :

le résultat

```
>>> primfermat(2011)
'il est possible que N soit premier'
>>> primfermat(1623)
(1623, 'non premier')
```