**Crible d’Eratosthène**

Méthode de détermination des premiers nombres premiers

**I] Description de la méthode et mise en œuvre**

*On dispose ci-contre d’une grille donnant les 101 premiers nombres entiers.
Le but est de barrer tous les nombres de la grille qui ne sont pas premiers.*

On considère l’algorithme ci-dessous :

****
► On dispose de la liste des nombres entiers de 0 à 100.

► Barrer 0 et 1.

► Parcourir dans l’ordre tous les entiers k de 2 à 100 :

 Si le nombre k n’est pas barré :

 Entourer k

 Barrer tous les multiples stricts de k dans la liste

► Renvoyer la liste des nombres qui ont été entourés.

1. Appliquer cet algorithme à la grille ci-contre.
2. Justifier brièvement pourquoi, après exécution de l’algorithme :
* les nombres barrés ne sont pas premiers ;
* les nombres entourés sont premiers.

**Cette méthode de détermination des premiers nombres premiers est appelée Crible d’Eratosthène.**

**II] Implémentation en langage Python**

1. Ecrire une instruction Python qui génère une liste Python $L$ de longueur 101 telle que $L\left[k\right]=k$, c'est-à-dire $L=[0,1,2,3,…,100]$.

Le but est de remplacer tous les nombres non premiers d’une telle liste par des 0 à l’aide du crible d’Eratosthene.

(0 correspondra ainsi à un nombre barré de la grille vue dans la partie I)

1. Tester le script Python ci-dessous pour différentes valeurs entières non nulles de k.

**for** j **in range**(2\*k,101,k):

 **print**(j)

Dans chaque cas, que représentent les valeurs affichées ? Justifier.

1. **a)** A l’aide des questions précédentes, écrire une fonction Python **Crible\_Eratosthene** qui applique l’algorithme de la partie I :
La fonction renverra la liste $L=[0,0,2,3,0,…,0]$, telle que $L\left[k\right]=\left\{ \begin{matrix}0\\k\end{matrix}\right.\begin{matrix} si k n^{'}est pas premier\\si k est premier\end{matrix}$

**b)** Modifier la fonction **Crible\_Eratosthene** pour qu’elle renvoie la liste des nombres premiers $[2,3,5,…,97]$.

**c)** Modifier la fonction **Crible\_Eratosthene** pour qu’elle reçoive en argument un entier $N$ et renvoie la liste de tous les nombres premiers inférieurs à $N$.

1. Calculer la fréquence des nombres premiers parmi les nombres inférieurs à $N$ pour $N=100$, $N=10000$ puis $N=1000000$.
Dans chaque cas, comparer cette fréquence avec $\frac{1}{ln⁡(N)}$, où ln est la fonction logarithme népérien.

Remarque : Il a été prouvé que, pour de grandes valeurs de *n*, la fréquence d’apparitions des nombres premiers entre 1 et *n*, est proche de $\frac{1}{ln⁡(n)}$ où ln est la fonction logarithme népérien (Théorème des nombres premiers). Cette fréquence étant décroissante, on dit qu’il y a raréfaction des nombres premiers.

Pour aller plus loin : <http://python.jpvweb.com/python/mesrecettespython/doku.php?id=liste_des_nombres_premiers>

© 2019/2020 – Franck CHEVRIER **** [www.python-lycee.com](http://www.python-lycee.com)