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.

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99
100									

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 $\mathrm{L}[\mathrm{k}] = \mathrm{k}$, c'est-à-dire $\mathrm{L} = [0,1,2,3,\ldots,100]$.

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)

2) 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.

3) **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 $\mathrm{L} = [0,0,2,3,0,\ldots,0]$, telle que $\mathrm{L}[\mathrm{k}] = \begin{cases} 0 & si\ \mathrm{k}\ n'est\ pas\ premier \\ \mathrm{k} & si\ \mathrm{k}\ est\ premier \end{cases}$

b) Modifier la fonction **Crible_Eratosthene** pour qu'elle renvoie la liste des nombres premiers $[2,3,5,\ldots,97]$.
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.

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