Chapitre 5

Recherche de nombres premiers

5.1. Les nombres premiers

Un nombre premier est un nombre qui n'est divisible que par un et par lui-même (par convention, 1 n'est pas un nombre premier). Les 25 nombres premiers inférieurs à 100 sont :

2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89 et 97.

Exercice 5.1

Écrivez un programme qui établit la liste de tous les nombres premiers compris entre 1 et *n*. Utilisez pour cela la définition d'un nombre premier.

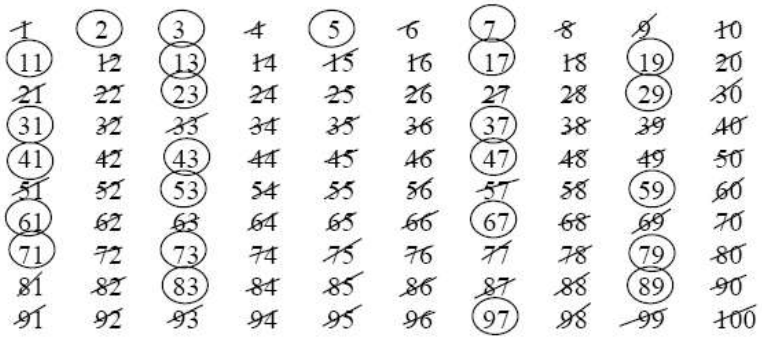
Optimisez votre programme. En effet,
- il est inutile de tester les nombres pairs (à part le 2) et les multiples de 5 ;
- il est inutile de tester les diviseurs pairs ;
- il est inutile de tester les diviseurs jusqu'à *n* (jusqu'où faut-il aller ?).

5.2. Le crible d'Eratosthène

On désigne sous le nom de *crible d'Eratosthène* (env. 276-194 av. J.-C.), une méthode de recherche des nombres premiers plus petits qu'un entier naturel *n* donné.

Algorithme

1. On écrit la liste de tous les nombres jusqu'à *n*.
2. On élimine 1.
3. On entoure 2 et on élimine tous les multiples de 2.
4. Puis on fait de même avec 3.
5. On choisit alors le plus petit nombre non entouré et non éliminé, et on élimine tous ses multiples.
6. On réitère le procédé jusqu'à la partie entière de la racine de *n*. Les nombres non éliminés sont les nombres premiers jusqu'à *n*.



Exercice 5.2

Écrivez un programme qui établit la liste de tous les nombres premiers compris entre 1 et n, en utilisant la méthode du *crible d'Ératosthène* :

1. Créer une liste de n+1 éléments (disons n = 1000), chacun initialisé à la valeur 1.
2. Pour les indices i allant de 2 à $\sqrt{n}$ faire :
3. Si l'élément d'indice i vaut 1, mettre à 0 tous les autres éléments de la liste dont les indices sont des multiples de i.
4. Parcourir toute la liste et écrire les indices des éléments qui sont restés à 1. Ce sont des nombres premiers.

5.3. Le test de primalité de Miller-Rabin

Un algorithme de Monte-Carlo est un algorithme qui utilise une source de hasard, et dont la sortie peut ne pas être la réponse au problème posé, mais c'est un cas très rare.

Le test de Miller-Rabin est un algorithme de Monte-Carlo qui peut décider avec un taux d'erreur très faible si un nombre est premier ou composé. Ce test est en fait un test de non-primalité, c'est-à-dire que s'il répond qu'un nombre n'est pas premier, alors il est sûr que ce nombre ne l'est pas. Il peut aussi répondre qu'un nombre est probablement premier. Dans ce cas, si ce nombre n'est pas premier, la probabilité de ne pas avoir été détecté non premier est infime : on peut imposer par exemple que cette probabilité soit inférieure à 2^{-100}.

L'algorithme peut être écrit de la façon suivante :

```
Données : n : une valeur impaire à tester pour sa primalité ;
          k : un paramètre qui détermine le nombre de fois
              qu'il faut tester la primalité.
Résultat : "composé" si on trouve pour n au moins un Témoin de Miller a,
           autrement il est "très probablement premier"

FONCTION miller_rabin(n,k)
    écrire n comme 2^r × s + 1
    REPETER k fois :
        prendre a aléatoirement dans l'intervalle [1, n-1]
        SI a^s mod n≠1 et a^(s·2^j) mod n≠-1 pour tout j dans [0, r-1] ALORS
            RETOURNER "composé"
    RETOURNER "probablement premier"
```

Plus on teste de valeurs de a (donc plus k est grand), meilleure est la précision du test. **Monier** (1980) et **Rabin** (1980) ont montré qu'un nombre composé passe le test pour au plus 1/4 des valeurs a. Si l'on fait k fois le test, alors on se trompera environ 1 fois sur 4^k.

Exercice 5.3

Programmez en Python le test de Miller-Rabin.

Sources

[1] Wikipédia, « Crible d'Ératosthène », <https://fr.wikipedia.org/wiki/Crible_d'Ératosthène>

[2] Wikipédia, « Test de primalité de Miller-Rabin », <https://fr.wikipedia.org/wiki/Test_de_primalité_de_Miller-Rabin>