Crible d'Eratosthène [th05] - Examen

Karine Zampieri, Stéphane Rivière

Unisciel algoprog Version 22 mai 2018

Table des matières

1 Crible d'Eratosthène / pgeratos **2**
1.1 Algorithme du Crible **(5 points)** 2
1.2 Liste des nombres premiers **(4 points)** 7
1.3 Programme **(1 point)** 8
1.4 Analyse mathématique 9
1.5 Spirale du crible 10

2 Références générales **10**

Python - Crible d'Eratosthène (Solution)

Mots-Clés Théorie des nombres ■
Requis Axiomatique impérative sauf Fichiers ■
Difficulté • • ◦

Objectif
Cet exercice détermine tous les nombres premiers inférieurs à un entier n par la méthode du crible d'Eratosthène.

...(énoncé page suivante)...

1 Crible d'Eratosthène / pgeratos

1.1 Algorithme du Crible (5 points)

Définition

Un **entier** est dit **premier** s'il possède exactement deux diviseurs : 1 et lui-même appelés diviseurs triviaux. Sinon il est dit **composite**.

Propriété

Le **crible d'Eratosthène**[1] permet de connaître en une seule fois un grand nombre d'entiers naturels premiers consécutifs et pas trop grands (par exemple inférieurs à un milliard).

Algorithme du crible

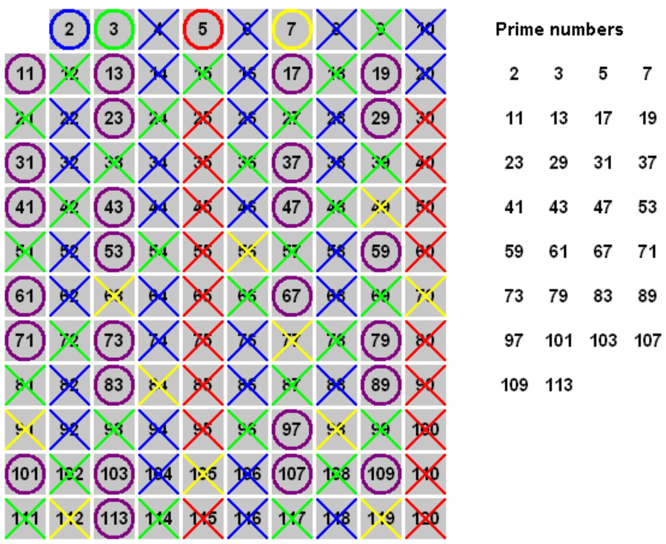
Pour connaître tous les nombres premiers jusqu'à n faire :

- Marquez à `Vrai` tous les entiers de 2 jusqu'à n.
- Marquez à `Faux` tous les multiples de 2 sauf 2.
- A partir de 3, répétez les deux points suivants et arrêtez-vous dès que $n/2$ est atteint (ou mieux : la racine carrée de n) :
  1. Repérez le premier entier *impair* k encore présent.
  2. Marquez à `Faux` tous ses multiples sauf k.

Ce qui reste est la liste des nombres premiers jusqu'à n.

Exemple

Considérons l'applet de WIKIPÉDIA :



Pour obtenir les nombres premiers inférieurs à 120 :

1. Mathématicien et philosophe, connu pour ses travaux en arithmétique et en géométrie, ERATOSTHÈNE vécut au III^e siècle avant J.C. à Alexandrie.

- Barrez tous les multiples de 2 (donc 4, 6, 8, etc., 118, 120)
- Puis tous les multiples de 3 (donc 6, 9, 12, etc., 117, 120)
- Puis tous les multiples de 5 (donc 10, 15, 20, etc., 115, 120)
- Puis tous les multiples de 7 (donc 14, 21, 28, etc., 112, 119)
- Et s'arrêter à 11 car $11^2 = 121 > 120$.

L'entier 84 est barré trois fois puisque 84 a trois facteurs premiers ≤ 11 (à savoir 2, 3 et 7). Ces qui reste (les non barrés) est la liste des entiers premiers inférieurs à 120.

Analyse

On utilise un tableau unidimensionnel de booléens de taille $n+1$, où on fait en sorte que l'élément à l'indice i soit égal à `Vrai` si `i` est premier. Ni l'entier 0, ni l'entier 1 (tous les nombres en sont ses multiples) ne sont premiers. Les calculs seront donc exécutés à partir de 2.

Au départ, tous les éléments du tableau sont initialisés à `Vrai` sauf pour les deux premiers éléments initialisés à `Faux`. Dans le traitement itératif, les nombres non premiers seront supprimés au fur et à mesure. Pour ce faire, pour tout i de 2 jusqu'à $\sqrt{n}$ ($i \leq \sqrt{n} \Leftrightarrow i \times i \leq n$), si l'élément à l'indice i est premier alors on supprime les multiples de i (cette étape nécessite une nouvelle variable correspondant à ces multiples).

Cette suppression commence à partir de i^2 car les multiples de i strictement inférieurs à i^2 ont été supprimés dans les itérations précédentes. Effectivement, tous les multiples de i sont : $2 \times i, 3 \times i, ..., (i-1) \times i, i^2, (i+1) \times i, ...$ (sans dépasser n et sans inclure, au début, le nombre couramment analysé, i, car il est premier). Mais la valeur $2 \times i$ est aussi un multiple de 2, déjà éliminé à l'étape de suppression des multiples de 2, la valeur $3 \times i$ est aussi un multiple de 3 déjà éliminé à l'étape de suppression des multiples de 3, etc. Le plus petit multiple qui n'a pas été supprimé dans les étapes précédentes, selon ce raisonnement, est i^2.

En appliquant ce mécanisme, nous pouvons démontrer que l'opération de suppression peut être faite jusqu'à $\sqrt{n}$, car dans l'ensemble de multiples, $i^2 \leq n$, donc $i \leq \sqrt{n}$.

(1 point) Définissez la constante `CMAX=100000` (nombre maximum de valeurs d'un crible) puis le type `Crible` comme étant un tableau de `CMAX` booléens.

(1 point) Écrivez une procédure `initialiserCrible(c,n)` qui initialise à `Vrai` les éléments d'un `Crible c[..n]`. Par défaut tous les nombres sont premiers. Marquez à `Faux`, les entiers 0 et 1.

Solution Paramètres

Entrants : L'entier `n`

Sortants : Le `Crible c`

(1 point) Écrivez une procédure `eliminerCrible(c,n,k)` qui marque à `Faux` tous les **multiples successifs** de `k` (entier), à savoir $2k, 3k...$, dans un `Crible c[..n]`.

Solution Paramètres

Entrants : Les entiers `n` et `k`

Modifiés : Le `Crible c`

Solution simple

Les multiples de k étant $2k, 3k...$, donc on commence par $k + k$ et on incrémente k de k à chaque tour de boucle.

(1 point) Écrivez une fonction `suivantImpair(c,n,k)` qui recherche et renvoie le suivant **impair** non marqué de `k` (entier impair) dans un `Crible c[..n]`, et qui renvoie -1 s'il n'existe pas (ce qui marquera la fin de la recherche).

Solution simple

Le premier entier impair j qui suit k est $k+2$. Tant qu'il est inférieur ou égal à n et que $c[j]$ est `Faux`, on passe à l'entier impair suivant en incrémentant j de 2. A la sortie de la boucle, si j est plus grand que n, alors c'est fini, sinon c'est j.

Écrivez le code sur cette partie...

Validez vos définitions, procédures et fonction avec la solution.

Solution Python @[pgeratos.py]

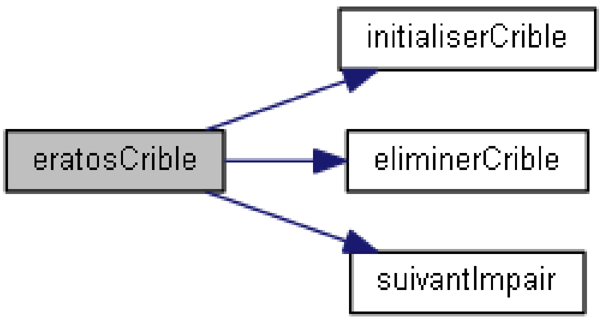
```
def initialiserCrible(n):
   c = []
   for j in range(0,n+1):
      c += [True]
   c[0] = c[1] = False
   return c
```

```
def eliminerCrible(c,n,k):
   for j in range(k+k,n+1,k):
      c[j] = False
```

```
def suivantImpair(c,n,k):
   j = k + 2
   while (j <= n and not c[j]):
      j += 2
   return j if (j <= n) else -1
```

(1 point) Écrivez une procédure `eratosCrible(c,n)` qui calcule les nombres premiers compris entre 1 et `n` dans un `Crible c`. La procédure comporte trois parties :

- L'initialisation par appel de la procédure `initialiserCrible`.
- L'élimination (procédure `eliminerCrible`) de tous les multiples de 2 (2 est le plus petit nombre premier).
- L'élimination de tous les multiples successifs de chaque entier **impair** en commençant par le plus petit 3.



Solution Paramètres

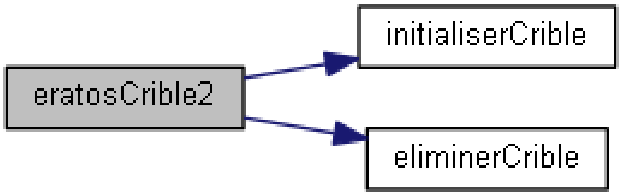
Entrants : L'entier `n`

Sortants : Le `Crible c`

De même, écrivez une procédure `eratosCrible2(c,n)` qui calcule les nombres premiers compris entre 1 et `n` dans un `Crible c` comme suit :

- L'initialisation par appel de la procédure `initialiserCrible`.

- L'élimination (procédure `eliminerCrible`) de tous les multiples de 2 (2 est le plus petit nombre premier).
- L'élimination de tous les multiples successifs des entiers **impairs** `k` non encore traités à partir de 3 tant que la racine carrée de `n` n'est pas atteinte.



Aide simple

Plutôt que d'utiliser $k \leq \sqrt{n}$, écrivez $k * k \leq n$.

Validez vos procédures avec la solution.

Solution Python @[pgeratos.py]

```
def eratosCrible(n):
   c = initialiserCrible(n)
   eliminerCrible(c, n, 2)
   ndiv2 = n // 2
   k = 3
   while (k != -1):
      eliminerCrible(c, n, k)
      k = suivantImpair(c, ndiv2, k)
   return c
```

```
def eratosCrible2(n):
   c = initialiserCrible(n)
   eliminerCrible(c, n, 2)
   k = 3
   while (k*k <= n):
      if (c[k]):
         eliminerCrible(c, n, k)
   return c
```

1.2 Liste des nombres premiers (4 points)

Ce problème détermine la liste des nombres premiers inférieurs à un entier naturel donné. (Pour 100000, il y a 9592 entiers premiers.) Il utilise la procédure `eratosCrible`.

(1 point) Définissez la constante `TMAX=1000` (nombre maximum de valeurs dans une liste), éventuellement le type `ITableau` comme étant un tableau d'entiers d'au plus `TMAX` entiers, puis le type `IListe` comme étant une structure contenant :

- Un `ITableau` contenant les valeurs.
- Un entier `taille` du nombre d'éléments effectifs dans le `ITableau`.

(0.5 point) Écrivez une procédure `initialiserListe(lt)` qui initialise une `IListe lt` à la liste vide (aucun élément, à savoir sa taille est nulle).

(1 point) Écrivez une procédure `ajouterElement(lt,valeur)` qui ajoute une valeur `valeur` (entier) dans une `IListe lt` (à la suite de ceux déjà présents).

Solution simple
On ajoute la valeur dans le tableau des éléments de la `IListe` et on incrémente sa taille.

(1 point) Écrivez une procédure `creerListeNP(n,lt)` qui crée la liste des entiers premiers inférieurs ou égaux à `n` dans une `IListe lt`.

Solution simple
On crée d'abord le crible par appel à la procédure `eratosCrible` : si `c[k]` est resté à `Vrai` alors `k` est premier. D'où, on initialise d'abord `lt` à la liste vide (procédure `initialiserListe`) (`lt` est sortant) puis on traverse le `Crible c` et on ajoute à `lt` tous les éléments `k` qui sont restés à `Vrai` dans `c` (c.-à.-d. si `c[k]` alors `ajouterListe(lt,k)`).

(0.5 point) Écrivez une procédure `afficherListe(lt)` qui affiche les valeurs du tableau d'une `IListe lt`.

1.3 Programme (1 point)

(1 point) Écrivez un script qui saisit un entier (supposé positif et inférieur à `CMAX`) puis calcule et affiche la liste des nombres premiers qui sont inférieurs ou égaux à cet entier.

Testez. Exemple d'exécution :

```
Entier dans [1..99999]? 120
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
   101 103 107 109 113
```

Validez votre script avec la solution.

Écrivez le code sur cette partie...

Validez votre procédure avec la solution.

Solution Python @[pgeratos.py]

```
n = saisieEntier(1,CMAX-1)
cb = eratosCrible(n)
for k in range(0,n+1):
   if (cb[k]):
      print(k," ",sep="",end="")
print()
```

1.4 Analyse mathématique

Prouvez qu'il est inutile d'étudier les entiers j marqués c.-à-d. si j n'est plus premier alors tous ses multiples auront déjà été supprimés.

Solution simple
Si j est marqué alors j n'est pas premier car il peut s'écrire $j = k\,p$ avec $1 < k$ et $p < j$ donc j est multiple de k (également de p). Tous les multiples de j sont aussi multiples de k. Comme k est inférieur à j, tous ses multiples sont déjà marqués.

Prouvez que le premier multiple de j à marquer est j^2 c.-à-d. les multiples $2j, 3j, \ldots$ ont déjà été supprimés.

Solution simple
Les multiples de j s'écrivent $2j$, $3j$, ..., $k\,j$. Or $k\,j$ est un multiple de k et de j donc si $k < j$ alors $k\,j$ est déjà marqué donc j^2 est le premier multiple de j.

Déduisez une version `eratosOptm(c,n)` optimisée.

Solution simple
Les nombres non marqués qui restent sont exactement les nombres premiers au plus égaux à n : en effet, les nombres marqués sont des multiples d'un de leurs prédécesseurs, et ne sont donc pas premiers, et aucun des nombres non marqués ne saurait être non premier, car il serait alors un multiple d'un de ses prédécesseurs, et aurait été marqué. De plus, il est inutile de poursuivre le marquage à partir du premier nombre non marqué $k > \sqrt{n}$: en effet, un premier nombre non marqué n'étant divisible par aucun de ses prédécesseurs est nécessairement premier, ses multiples par un de ses prédécesseurs ont tous été marqués (en tant que multiple d'un diviseur premier d'un prédécesseur), et par conséquent, son premier multiple non marqué est $k^2 > n$: il n'y a donc plus de marquage possible.

1.5 Spirale du crible

Définition

On appelle la n-**spirale des nombres premiers**, dite aussi *Spirale d'*ULAM, la spirale des nombres 1 à n (fig. de gauche) où on affiche un point si le nombre est premier et rien sinon (fig. de droite)

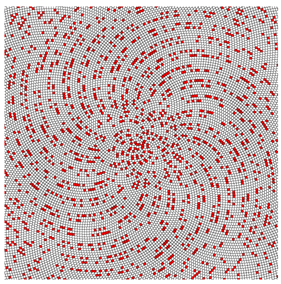
17	16	15	14	13		•				•
18	5	4	3	12			•		•	
19	6	1	2	11	$\longrightarrow$	•		•	•	•
20	7	8	9	10			•			
21	...						...			

Écrivez une fonction `tracerSpirale(c,n)` qui génère la spirale d'ULAM issu d'un `Crible` `c[..n]`.

Aide simple

Vérifiez que les droites $x + y$, $x + y - 1$ et $x - y$ déterminent les quatre régions où dans chacune il convient de faire :

- `++x` quand $(x - y \geq 0 \text{ et } x + y - 1 < 0)$
- `++y` quand $(x + y - 1 \geq 0 \text{ et } x - y > 0)$
- `--x` quand $(x - y \leq 0 \text{ et } x + y > 0)$
- `--y` quand $(x + y \leq 0 \text{ et } -x - y < 0)$



2 Références générales

Comprend [Chappelier-CPP1 :c2 :ex9], [Chaty-PG1 :c7 :ex6], [Engel-AL1 :c2 :xm04], [Felea-PG1 :c3 :ex67], [Maunoury-AL1 :c7 :ex16] ■