

Programmation avec Maple (2)

Table des matières

| | | |
|-----|---|---|
| 1 | Quelques rappels sur les nombres premiers. | 1 |
| 2 | Feuilles de calcul | 1 |
| 2.1 | Compter les nombres premiers compris entre 1 et 100 | 1 |
| 2.2 | Énumérer les nombres premiers compris entre 1 et 100 | 2 |
| 2.3 | Compter les couples de nombres premiers jumeaux compris entre 1 et 100 | 2 |
| 2.4 | Énumérer les couples de nombres premiers jumeaux compris entre 1 et 100 | 2 |
| 3 | Écriture de fonctions | 3 |
| 3.1 | Introduction | 3 |
| 3.2 | Compter les nombres premiers compris dans un intervalle $[[a, b]]$ | 3 |
| 3.3 | Compter les couples de nombres premiers jumeaux compris dans un intervalle $[[a, b]]$ | 3 |
| 3.4 | Recherche de triplets de nombres premiers (p, q, r) vérifiant $p + r = 2q$ | 4 |
| 4 | Compléments culturels | 4 |

► Nous présentons dans une première partie des exemples de programmes rédigés sous la forme d'une feuille de calcul. Dans une deuxième partie, nous apprenons à rédiger des fonctions, qui offrent des fonctionnalités analogues aux programmes précédents, mais dotées de plus de souplesse.

1 Quelques rappels sur les nombres premiers.

Définition : un nombre premier est un naturel qui possède exactement deux facteurs : 1 et lui-même ; par exemple, les nombres premiers inférieurs à 10 sont 2, 3, 5 et 7. L'ensemble des nombres premiers est infini.

Définition : deux nombres premiers sont dits *jumeaux* si leur différence est égale à 2. Exemple : 17 et 19 sont deux nombres premiers jumeaux. On ne sait pas s'il existe une infinité de couples de nombres premiers jumeaux.

Définition : *incrémenter* un compteur c consiste à remplacer c par $c + 1$.

2 Feuilles de calcul

Nous utiliserons plusieurs fonctions prédéfinies dans Maple, et qui portent sur les nombres premiers :

- `isprime(p)` indique si le nombre p est premier ;
- `nextprime(p)` fournit le plus petit nombre premier strictement supérieur à p .

2.1 Compter les nombres premiers compris entre 1 et 100

Ce premier exercice est très facile ; nous utiliserons une boucle `for .. do .. od`, la fonction `isprime` et un compteur. L'algorithme utilisé est :

1. nous initialisons le compteur avec la valeur 0 ;
2. nous balayons l'intervalle discret $[[1, 100]]$: à chaque entier premier rencontré, nous incrémentons le compteur ;
3. une fois la boucle terminée, nous affichons le compteur.

```
compteur := 0;
for k from 1 to 100
do
  if isprime(k) then compteur := compteur + 1 fi
od ;
compteur ;
```

Enregistrez ce programme sur votre ordinateur, puis exécutez-le: vous constaterez qu'il existe 25 nombres premiers dans l'intervalle discret $\llbracket 1, 100 \rrbracket$.

2.2 Énumérer les nombres premiers compris entre 1 et 100

Ce deuxième exercice est lui aussi très facile; en plus des outils utilisés dans le précédent exercice, nous ferons appel à la structure de liste. En Maple, une liste a l'allure suivante: $[x_1, x_2, \dots, x_n]$. La liste vide s'écrit $[\]$; pour ajouter un objet k à la fin d'une liste, nous disposons de la syntaxe suivante:

```
ma_liste := [op(ma_liste),k] ;
```

Voici un programme qui va énumérer les nombres premiers compris entre 1 et 100:

```
liste_premiers := [] ;
for k from 1 to 100
do
  if isprime(k) then liste_premiers := [op(liste_premiers),k] fi
od ;
liste_premiers ;
```

Enregistrez ce programme sur votre ordinateur, puis exécutez-le; vous obtiendrez la liste suivante, qui compte exactement 25 éléments:

```
[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]
```

2.3 Compter les couples de nombres premiers jumeaux compris entre 1 et 100

L'algorithme proposé est très simple:

- nous mettons à zéro un compteur;
- nous balayons la liste des entiers de 1 à 100, en examinant chaque nombre premier p : si $p+2$ est premier, tout en étant inférieur à 100, alors p et $p+2$ sont jumeaux: dans ce cas, nous incrémentons le compteur.

Voici un programme qui va compter les couples de nombres premiers jumeaux compris entre 1 et 100:

```
compteur := 0 ;
for k from 1 to 100
do
  if isprime(k) and isprime(k+2) and k+2 <= 100 then compteur := compteur + 1 fi
od;
compteur ;
```

Enregistrez ce programme sur votre ordinateur, puis exécutez-le: il existe huit couples de nombres premiers jumeaux dans l'intervalle discret $\llbracket 1, 100 \rrbracket$.

2.4 Énumérer les couples de nombres premiers jumeaux compris entre 1 et 100

Cette fois, nous voulons construire la liste des couples $(p, p+2)$ de nombres premiers jumeaux, compris entre 1 et 100. Il suffit de s'inspirer du programme proposé à la section 2.2; chaque couple de premiers jumeaux sera présenté comme une liste de deux éléments. Le résultat affiché sera donc une liste de listes.

```
liste_jumeaux := [] ;
for k from 1 to 100
do
  if isprime(k) and isprime(k+2) and k+2 <= 100
  then liste_jumeaux := [op(liste_jumeaux),[k,k+2]] fi
od;
liste_jumeaux ;
```

Voici la liste résultant de ce calcul; vous constaterez qu'elle comporte bien huit couples.

```
[[3,5], [5,7], [11,13], [17,19], [29,31], [41,43], [59,61], [71,73]]
```

3 Écriture de fonctions

3.1 Introduction

Nous allons reprendre chacun des exemples précédents, mais en rédigeant à chaque fois une *fonction* : ceci nous donnera une grande souplesse d'utilisation.

Commençons par rappeler l'allure générale de la définition d'une fonction dans le langage Maple :

```
ma_fonction := proc(u,v) local x, y, resultat ;
  .. une instruction ..
  .. une autre instruction ..
  .. d'autres instructions ..
  RETURN(resultat)
end ;
```

Le nom de la fonction est `ma_fonction` ; `u` et `v` sont les *arguments* de la fonction ; enfin, `x`, `y` et `resultat` sont des *variables locales* : elles sont créées lors de l'appel de la fonction, et leur vie se terminera lorsque la fonction rendra son résultat.

3.2 Compter les nombres premiers compris dans un intervalle $\llbracket a, b \rrbracket$

Nous rédigeons une fonction `compte_preliers` ; cette fonction attend deux arguments `a` et `b`, qui doivent être des naturels et vérifier $a \leq b$. Le résultat rendu par la fonction sera le nombre de nombre premiers compris entre `a` et `b` inclus.

```
compte_preliers := proc(a,b) local compteur, k ;
  compteur := 0 ;
  for k from a to b
  do
    if isprime(k) then compteur := compteur + 1 fi
  od ;
  RETURN(compteur)
end ;
```

Il vous suffit maintenant de passer la commande `compte_preliers(1000,10000)` ; pour apprendre qu'il y a exactement 1061 nombre premiers compris entre 1000 et 10000 inclus !

Remarque : par mesure de sécurité, nous devrions vérifier que les valeurs `a` et `b` transmises à la fonction sont des entiers vérifiant (à tout le moins) $2 \leq a < b$.

3.3 Compter les couples de nombres premiers jumeaux compris dans un intervalle $\llbracket a, b \rrbracket$

Replaçons le schéma utilisé à la section 2.3 dans une fonction :

```
compte_jumeaux := proc(a,b) local compteur, k ;
  compteur := 0 ;
  for k from a to b
  do
    if isprime(k) and isprime(k+2) and k+2 <= b then compteur := compteur + 1 fi
  od ;
  RETURN(compteur)
end ;
```

L'exécution de ce programme vous permettra d'apprendre qu'il y a cinq couples de nombres premiers jumeaux dans l'intervalle $\llbracket 1000, 1100 \rrbracket$, quatre dans l'intervalle $\llbracket 10000, 10100 \rrbracket$ et aucun dans l'intervalle $\llbracket 100000, 100100 \rrbracket$.

3.4 Recherche de triplets de nombres premiers (p, q, r) vérifiant $p + r = 2q$

Nous rédigeons une fonction qui, dans un intervalle $\llbracket a, b \rrbracket$ fixé, va chercher des triplets de nombres premiers en progression arithmétique. Voici deux exemples de tels triplets: $(3, 5, 7)$ et $(17, 23, 29)$.

Décrivons l'algorithme utilisé :

1. déterminons le plus petit nombre premier p dans l'intervalle $\llbracket a, b \rrbracket$; si nous n'en trouvons aucun, c'est fini ;
2. déterminons le plus petit nombre premier q appartenant à $\llbracket p+1, b \rrbracket$; si nous n'en trouvons aucun, c'est fini ;
3. sinon, pour chaque nombre premier q appartenant à $\llbracket p+1, b \rrbracket$, notons $r = 2q - p$; si r est premier et nous avons trouvé un triplet répondant à la question ;
4. une fois que l'on a visité tous les nombres premiers de l'intervalle $\llbracket p+1, b \rrbracket$, on remplace p par `nextprime(p)` et q par `nextprime(nextprime(p))` ; si $q < b$, on reprend à l'étape 3 ; sinon, c'est fini.

Voici une proposition pour le programme correspondant :

```
recherche := proc(a,b) local p,q,r,liste;
  if type(n,integer) and type(b,integer) and a>=0 and b>a
    then RETURN() fi;
  liste := [] ;
  if isprime(a) then p := a else p := nextprime(a) fi;
  if p>b then RETURN() fi;
  q := nextprime(p);
  while q<b do
    r := 2*q-p;
    if r<= b and isprime(r)
      then liste := [op(liste),[p,q,r]] fi;
    q := nextprime(q);
    if q>=b then p := nextprime(p); q := nextprime(p) fi;
    if p>b then RETURN(liste) fi;
    if q>b then RETURN(liste) fi;
  od;
  RETURN(liste)
end ;
```

Il est facile de généraliser, pour obtenir des progressions arithmétiques ne contenant que des nombres premiers, et dont la longueur est imposée ; par exemple, la séquence 5, 11, 17, 23, 29.

4 Compléments culturels

Le *théorème des nombres premiers* s'énonce ainsi : si nous notons $\pi(x)$ le nombre de nombres premiers inférieurs à x , alors $\pi(x) \underset{x \rightarrow \infty}{\sim} \frac{x}{\ln(x)}$. Notons $p(n)$ le n -ième nombre premier ; on peut déduire du théorème précédent le résultat suivant : $p(n)$ est équivalent à $n \ln(n)$ lorsque n tend vers l'infini.

FIN