Les nombres premiers

M.Gouy G.Huvent A.Ladureau

12 mai 2002

1 Etude des nombres premiers

1.1 Comment déterminer si un entier N donné est premier ?

Principe : soit N un entier strictement supérieur à 1, si N est composé alors il existe deux entiers p et q strictement supérieurs à 1 tel que $N = p \times q$. Si p et q sont strictement supérieurs à $\sqrt{N}$ alors on aurait $p \times q > N$. On en déduit donc que tout nombre composé admet au moins un diviseur autre que 1 inférieur ou égal à $\sqrt{N}$. On peut tirer de cette propriété l'algorithme et les programmes suivants :

Algorithmes :

Programme principal
Effacer l'écran
Demander N
Exécuter le programme NPREM1
Si W=0
Alors
afficher N est premier
Sinon
Afficher N est composé, il est divisible par D
Fin du Si

Sous-programme (testant la primalité)
Mettre 2 dans D et O dans W
Tant que $D^2 \leq M$ et $W = 0$
Si D divise N
Mettre 1 dans W
Fin du Si
Augmenter D de 1
Fin du Tant que

Programme principal (NPREM)

TI-83	CASIO	TI-92
`ClrHome`	`ClrText`	`nprem(n)`
`Prompt N`	`''N=''?→N`	`Prgm`
`prgmNPREM1`	`Prog ''NPREM1''`	`ClrIo :nprem1(n)`
`If W=0`	`If W=0`	`If w=0`
`Then`	`Then ''N EST PREMIER''`	`Then`
`Disp ''N PREMIER''`	`Else ''N EST COMPOSE ET DIVISIBLE PAR''`	`Disp ''n PREMIER''`
`Else`	`D-1 ▲`	`Else`
`Disp ''N COMPOSE''`	`IfEnd`	`Disp ''n COMPOSE''`
`Disp ''N DIVISIBLE PAR''`		`Disp ''n DIVISIBLE PAR''`
`Disp D-1`		`Disp d-1`
`End`		`EndIf :EndPrgm`

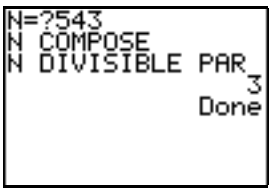
Sous-programme NPREM1 :

TI-83	Casio	TI-92

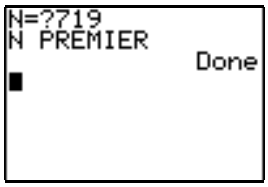
2→D
0→W
While D²≤ N and W=0
If Fpart(N/D)=0
Then
1→W
End
D+1→D
End
	2→D
	0→W
	While D²≤ N and W=0
	If Frac (N/D)=0
	Then 1→W
	IfEnd
	D+1→D
	WhileEnd
		nprem1(n)
		Prgm
		2→ d :0→ w
		While d²≤ n and w=0
		If Fpart(n/d)=0
		Then
		1→w
		EndIf
		d+1→d
		EndWhile
		EndPrgm

Exemples d'écrans :

TI-83



TI-83



Casio

Remarque 1 *On peut améliorer le sous-programme* ***NPREM1*** *en ne considérant que les divisions par* 2 *et les nombres impairs. Pour cela, il suffit de remarquer que si l'on pose initialement* $D = 2$*, on augmente* D *de* 1 *lorsque* $D = 2$ *et de* 2 *lorsque* $D \neq 2$*. Cela peut s'écrire par exemple :*

$$1 + D + (D \neq 2) \to D$$

si la calculatrice travaille avec les booléens, ou quelle que soit la machine

$$2 + D - E\left(\frac{2}{D}\right) \to D$$

puisque $\frac{2}{D}$ *appartient à* $[0, 1[$ *dès que* $D \neq 2$ *et vaut* 1 *lorsque* $D = 2$*.*

Remarque 2 *Ce qui précède revient à tester les divisions par* 2 *et les nombres de la forme* $2k + 1$ $(k > 0)$*, soit un nombre sur deux. On peut faire mieux, en remarquant que, parmi les nombres de la forme* $6k + a$ $(k > 0$ *et* a *entier dans* $[0, 6[\,)$*, seuls les nombres de la forme* $6k + 1$*,* $6k + 5$ *peuvent être premiers. Nous n'aurons alors à tester qu'un nombre sur* 3*. L'algorithme devient alors*

Algorithme : programme NPREM2
Mettre 4 dans E et 0 dans W
Pour D allant de 2 à 3 pas de 1
Si D divise N
Alors mettre 1 dans W
Fin du pour
Mettre 5 dans D
Tant que $D^2 \leq N$ *et* $W = 0$
Si D *divise* N
Mettre 1 *dans* W
Fin du Si
Si $E = 4$
Alors mettre 2 dans E
Sinon mettre 4 *dans* E
Fin du Si
Augmenter D de E
Fin du tant que

TI-83
`0→W :4→E`
`For(D,2,3,1)`
`If Fpart(N/D)=0`
`1→W`
`End`
`5→D`
`While D²≤ N et W=0`
`If Fpart(N/D)=0`
`Then`
`1→W`
`End`
`If E=4`
`Then`
`2→E`
`Else`
`4→E`
`End`
`D+E→D :End`

CASIO
`0→W :4→E`
`For 2→D to 3 Step 1`
`If Frac(N/D)=0`
`1→W`
`Next`
`5→D`
`While D²≤ N et W=0`
`If Frac (N/D)=0`
`Then 1→W`
`IfEnd`
`If E=4`
`Then 2→E`
`Else 4→E`
`IfEnd`
`D+E→D`
`WhileEnd`

Le programme sur 89-92 est laissé aux soins du lecteur.

Tout cela est beau, mais essayez de tester la primalité de 20639383, de 117876683047 ou de leur produit (uniquement sur la 92 vu le nombre de chiffre) !

1.2 Critères utilisant le théorème de Fermat

Prérequis 3 *Soit n un entier naturel*

1. *Si n est premier alors ${}^{\mathbb{Z}}/_{n\mathbb{Z}}$ est un corps*
2. *Si n est premier, alors*
$$x^2 \equiv 1\ [n] \Leftrightarrow \begin{cases} x \equiv 1\ [n] \\ ou\ x \equiv -1\ [n] \end{cases}$$
3. *Si n est premier alors pour tout entier a de $[[1, n-1]]$, $a^{n-1} \equiv 1\ [n]$ (petit théorème de* Fermat*)*

Un entier n impair strictement supérieur à 10 étant donné, comment tester sa primalité à l'aide du théorème de Fermat ? A priori cela n'est pas réalisable car le théorème de Fermat n'admet pas de réciproque (par exemple, $2^{561-1} \equiv 1\ [561]$. On pourra se reporter au document sur Fermat et plus particulièrement à la partie sur les nombres de Carmichael).
Choisissons au hasard un entier a dans $[1, n-1]$

- Si a^{n-1} n'est pas congru à 1 modulo n alors n est composé.
- Si $a^{n-1} \equiv 1[n]$ alors on ne peut rien dire.
  Dans ce cas renouvelons l'opération et tirons un nouvel entier b dans $[1, n-1]$.
  Si b^{n-1} n'est pas congru à 1 modulo n alors n est composé.
  Si $b^{n-1} \equiv 1[n]$ alors on ne peut rien dire.

Si l'on réitère 20 fois cette opération, de deux choses l'une :

- On trouve lors du ième tirage un entier x tel que x^{n-1} n'est pas congru à 1 modulo n et alors on peut conclure que l'entier n est composé.
- Ou alors, pour tous les entiers x choisis, on a $x^{n-1} \equiv 1\ [n]$, on dira alors que n est un nombre "*pseudo-premier*".

Mise en place de l'algorithme et des programmes :

Programme pricipal	Sous-programme calculant le reste de K^P par N
Effacer écran	Tant que exposant P est >0
Demander N	Si P est pair
Initialiser R(=reste) à 1 , I (=compteur) à1	Alors
Tant que R=1 et I $\leq$ 20	Mettre P/2 dans P
Tirer au hasard un nombre K dans [1,N-1]	Mettre le reste de K^2 divisé par N dans K
Mettre N-1 dans l'exposant P	Sinon
Calculer le reste R de K^{N-1}par N	Mettre le reste de K*R divisé par N dans R
Augmenter I de 1	Mettre P-1 dans P
Fin du tant que	Fin du Si
Si R=1	Fin du tant que
Alors	
Afficher N est pseudo premier	
Sinon	
Afficher N est composé	
Fin du Si	

Programme Principal : PREMPRO1

TI-83
```
ClrHome
Prompt N
1→I :1→R
While I≤ 20 and R=1
randInt(1,N-1)→K
N-1→P
prgmPREMR
I+1→I
End
If R=1
Then
Disp ''N est Pseudo-PREMIER''
Else
Disp ''N est COMPOSE''
End
```

Casio
```
ClrText
''N=''?→N
1→I :1→R
While I≤ 20 and R=1
Int((N-1)*rand#)+1→K
N-1→P
Prog ''PREMR''
I+1→I
WhileEnd
If R=1
Then ''N est Pseudo-PREMIER''
Else ''N est COMPOSE''
IfEnd
```

TI-92
```
prempro1(n)
prgm
local i,r
1→i :1→r
While i≤ 20 and r=1
Int((n-1)*rand())+1→k
premr(n,n-1,k)→r
i+1→i
EndWhile
If r=1
Then
Disp ''n est pseudo premier''
Else
Disp ''n est composé''
EndIf :Endprgm
```

Sous-programme :PREMR

TI-83
```
While P>0
If fPart(P/2)=0
Then
P/2→P
K²-N*int(K²/N)→K
Else
K*R-N*int(K*R/N)→R
P-1→ P
End
End
```

Casio
```
While P>0
If Frac(P/2)=0
Then P/2→P
K²-N*int(K²/N)→K
Else K*R-N*int(K*R/N)→R
P-1→P
IfEnd
WhileEnd
```

TI-92
```
premr(n,p,k)
Func
While p>0
If fPart(p/2)=0
Then
p/2→p
k²-n*int(k²/n)→k
Else
k*r-n*int(k*r/n)→r
p-1→p
EndIf
EndWhile
r
EndFunc
```

1.2.1 Analyse statistique du test

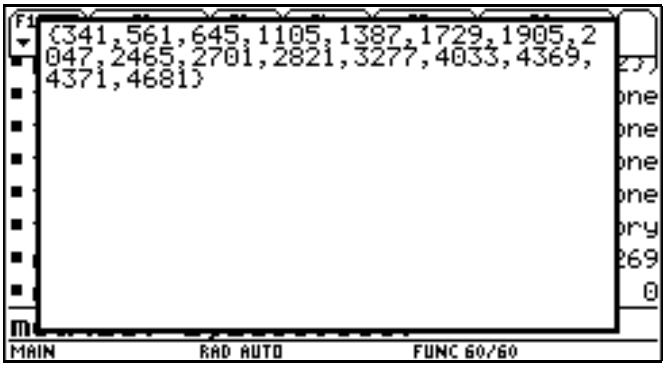
Détermination des entiers a impairs compris entre 3 et m vérifiant $2^{a-1} \equiv 1\ [a]$

Algorithme
Effacer l'écran. Effacer la varaible l1 Mettre liste vide dans l1 Pour i allant de 3 à m avec un pas de deux Si i est composé alors si $2^{i-1} est$congru à 1 modulo i Alors afficher i, stocker i dans l1 Fin du Si Fin du Si Fin du Pour Afficher l1 Fin du programme

TI-92

`prembas2(m)`
`Prgm`
`DelVar l1 :clrIo`
`{}→l1`
`For i,3,m,2`
`If isPrime(i)=false Then`
`If premr(i,i-1,2)=1 Then`
`Disp i :augment(l1,{i})→l1`
`EndIf`
`EndIf`
`EndFor`
`Disp l1`
`EndPrgm`

On obtient ainsi pour $m = 5000$ la liste suivante :



Remarque 4 *Entre 2 et 5000, il y a 2150 entiers impairs composés. Le test $2^{a-1} \equiv 1\ [a]$ va donc détecter dans $\frac{2134}{2150} \times 100\,\%$ (soit plus de 99 %) des cas la primalité de a.*[1]

Plus généralement, on peut déterminer les entiers a impairs compris entre 3 et m vérifiant $b^{a-1} \equiv 1\ [a]$ où b est un entier donné strictement supérieur à 1.
Le programme précédent se modifie facilement en :

Algorithme
Demander m et b Effacer l'écran. Effacer la varaible l1 Mettre liste vide dans l1 Pour i allant de 3 à m avec un pas de deux Si i est composé alors si b^{i-1} est congru à 1 modulo i Alors etc ..

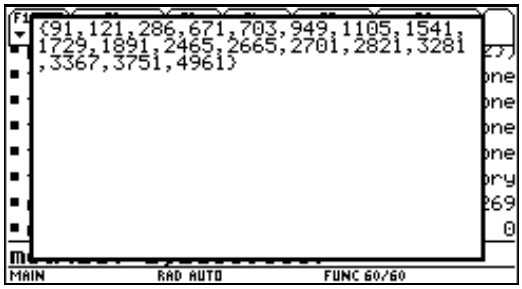
TI-92

`prembasb(m,b)`
`Prgm`
`DelVar l1 :clrIo`
`{}→l1`
`For i,3,m,2`
`If isPrime(i)=false Then`
`If premr(i,i-1,b)=1 Then`
`etc ..`

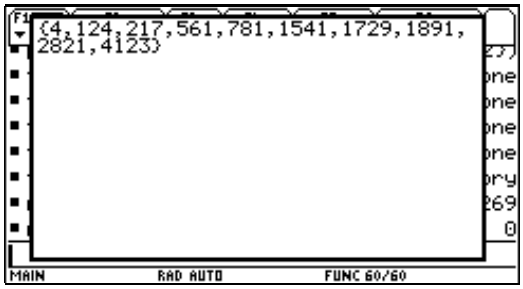
[1] R.Pinch a calculé qu'il existe 28975 entiers a inférieur à 10^{11} tels que $2^{a-1} = 1\ [a]$. Puisqu'il y a 4 118 054 813 nombres premier inférieur à 10^{11}, le test proposée détecte 99.99996978 % des entiers premiers.

Cela donne par exemple :

Pour $b = 3$ et $m = 5000$



Pour $b = 5$ et $m = 5000$



Un rapide coup d'oeil permet de constater qu'entre 3 et 5000, il n'y a que :
- $1105, 1729, 2465, 2701$ et 2821 qui sont communs au critère $b^{a-1} \equiv 1[a]$ avec $b = 2$ et 3.
- 1729 et 2821 qui vérifient $b^{a-1} \equiv 1[a]$ avec $b = 2, 3$ et 5.

Conséquence : On peut en tirer un programme testant la primalité d'un entier inférieur ou égal à 5000.

Algorithme (programme TEST)

```
Effacer l'écran : demander N
Mettre 2 dans D et 1 dans W
Tant que D≤ 5 et W=1
Si D^(N−1) n'est pas congru à 1 modulo N
Mettre 0 dans W ( N est composé)
Fin du si

Passer au candidat D suivant
Fin du tant que
Si W = 1
Alors
Afficher N est premier
Sinon
N est composé
Fin du Si
Fin du programme
```

TI-83

```
ClrHome :Prompt N
2→D :1→W
While D≤5 and W=1
D→K:1→R:N-1→P:prgmPREMR
If R≠1
Then
0→W
End
D+2-int(2/D)→D
End
Disp N
If W=1
Then
Disp "EST PREMIER"
Else
Disp "EST COMPOSE"
End
```

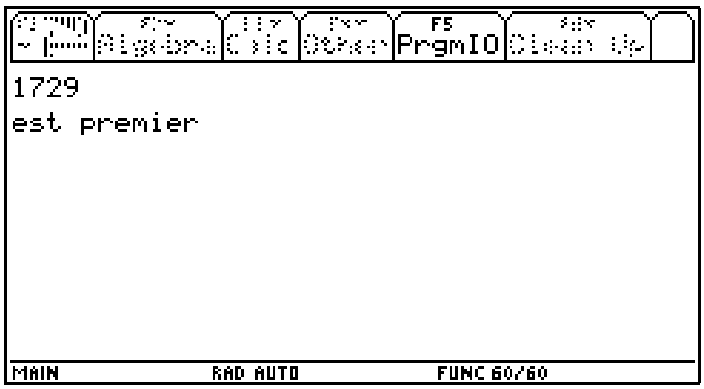
Casio

```
ClrText :"N=" ?→N
2→D :1→W
While D≤5 and W=1
D→K:1→R:N-1→P :Prog "PREMR"
If R≠1
Then 0→W
IfEnd
D+2-int(2/D)→D
WhileEnd
If W=1
Then N :"EST PREMIER" ▲
Else N :"EST COMPOSE" ▲
IfEnd
```

TI-92

```
test(n)
Prgm
2→d :1→w
While d≤5 and w=1
If premr(n,n-1,d)≠1 Then
0→w
EndIf
d+2-int(2/d)→d
EndWhile
ClrIo :Disp n
If w=1 Then
Disp "est premier"
Else
Disp "est compose"
EndIf
EndPrgm
```

on obtient alors (mais est-ce étonnant ?) :



1.2.2 Critère de Miller-Rabin

Introduction : Soit n un entier **premier** impair, $n-1$ est alors un entier pair strictement supérieur à 2. Posons :

$$n-1 = 2^s \times t \text{ où } t \text{ est un nombre impair et } s \text{ un entier au moins égal à 1.}$$

Considérons désormais un entier a compris entre 2 et $n-1$ et posons pour tout entier i de l'intervalle $[0, s-1]$

$$a_i = a^{2^i \times t}$$

On a :

$$\begin{array}{rcl} a_0 & = & a^t, a_s = a^{n-1} \\ \forall i & \in & [0, s-1], (a_i)^2 = a_{i+1} \end{array}$$

Examinons ce que l'on peut dire des a_i, en se rappelant que n est premier.
D'après Fermat :

$$\begin{array}{rcl} a_s & \equiv & 1\,[n] \Longrightarrow (a_{s-1})^2 \equiv 1\,[n] \\ & \Longrightarrow & \left\{ \begin{array}{rl} a_{s-1} \equiv 1[n] & (1) \\ \text{ou } a_{s-1} \equiv -1[n] & (2) \end{array} \right. \end{array}$$

Si la propriété (1) est vraie, on recommence le raisonnement et l'on en déduit que :

$$\left\{ \begin{array}{rl} a_{s-2} \equiv 1[n] & (1.1) \\ \text{ou } a_{s-2} \equiv -1[n] & (2.2) \end{array} \right.$$

Si la propriété (1.1)est vraie, on recommence le raisonnement etc.
On en déduit ainsi qu'au bout de n étapes au maximum, on aboutit à deux possibilités :

1. Pour tout i de $[0, s]$, on a $a_i \equiv 1\,[n]$ et en particulier $a_0 = a^t \equiv 1\,[n]$
2. Il existe i de $[0, s]$ tel que $a_i \equiv -1\,[n]$

D'où les propriétés suivantes :

Proposition 5 *Soit p un entier premier impair. Posons $p-1 = 2^s \times t$ où t est un nombre impair. Pour tout entier a de $[2, p-1]$, on a*

$a^t \equiv 1[p]$
ou il existe un entier i de $[0, s-1]$ tel que $a^{2^i \times t} \equiv -1\,[p]$.

Définition 6 *Un entier a vérifiant la propriété précédente pour p non premier, est appelé un témoin de Miller pour p.*

Conclusion 7 *Soit p un entier impair différent de 1. Posons $p-1=2^s\times t$ où t est un nombre impair. S'il existe un entier a de $[2,p-1]$ tel que a^t n'est pas congru à 1 modulo p* ***et*** *pour tout entier i de $[0,s-1]$, $a^{2^i\times t}$ n'est pas congru à -1 modulo p,alors p est composé.*

Proposition 8 (admise) *Si n est un nombre impair composé, alors au moins trois-quarts des $n-2$ entiers compris entre 2 et $n-1$ sont des témoins de Miller pour n.*

De là, on peut tirer un critère probabiliste pour tester la primalité d'un entier impair. En effet, soit n un entier impair donné. Choisissons au hasard un entier dans $[2,n-1]$. La probabilité de tomber sur un témoin de Miller de n est inférieure où égale à $\frac{1}{4}$. Si l'on renouvelle l'opération 10 fois. On a donc une probabilité maximale de $\frac{1}{4^{10}}$ soit 9.54×10^{-7} de tomber sur exactement 10 témoins de Miller pour p. D'où :

Si lors d'un des tirages, on tombe sur un nombre a non témoin de Miller, on est sûr que n est composé.
Si lors des 10 tirages, on tombe sur exactement 10 témoins de Miller, on peut dire que la probabilité qu'il soit composé est très faible. Dans ce cas, on affichera qu'il est premier.

Cela donne les programmes suivants :

Programme principal : MILLER

Algorithme
Effacer Ecran : Demander N Calculer S et T.Preparer liste 1 de dimension S Mettre 1 dans W et 1 dans J=tour Tant que J$\leq$ 10 et W =1 Tirer un nombre A dans [2,N-1] Calculer R=p1=reste de A^Tdivisé par N Si R $\neq$ 1 Alors mettre R dans l1(1) Pour I allant de 1 à s Mettre le reste de l1(I)^{2}divisé par N dans l1(I+1) Fin du Pour Ajouter 1 à l1 :Calculer les restes de chaque élément par N Si le produit des éléments de l1 $\neq$ 0 ($\Leftrightarrow$ il n'y a pas de -1) Alors mettre 0 dans W (N est donc composé) Fin du Si Fin du Si J augmente de 1 Fin du Tant Que Affichage du résultat

TI-83

ClrHome :Prompt N :ClrList L_1
PrgmDEUEXPO :(N-1)/2^S→T :S→ dim(L_1)
1→J :1→W
While J $\leq$10 and W=1
RandInt(2,N-1)→A
A→K :T→P:1→R :PrgmPREMR
If R$\neq$ 1
Then
R→L_1(1)
For(I,1,S-1,1)
$L_1(I)^2$-N*Int($L_1(I)^2$/N)→L_1(I+1)
End
1+L_1→L_1 :L_1-N*Int(L_1/N)→L_1
If prod(L_1)$\neq$0
Then
0→W
End
End
J+1→J
End
Disp N
If W=1
Then
Disp "EST PREMIER"
Else
Disp "EST COMPOSE"
End

Casio
ClrText :''N=''?→N Prog''DEUEXPO'' :(N-1)/2^S→T :S→dim(List1) 1→W :1→J While J ≤10 and W=1 Int((N-2)*Rand#)+2→A A→K :T→P:1→R:Prog ''PREMR'' If R≠1 Then R→List 1[1] For 1→I To S Step1 List 1[I]^2-N*Int(List1[I]^2/N)→List 1[I+1] Next 1+List 1→List1 :List1-N*Int(List1/N)→List 1 If prod(List1)≠0 Then 0→W IfEnd IfEnd J+1→J WhileEnd If W=1 Then N :''EST PREMIER'' ▲ Else N :''EST COMPOSE'' ▲ IfEnd

TI-92

test2(n)
Prgm
ClrIO :DelVar l1
deuexpo(n)→s :(n-1)/2^s→t
1→w :1→tour
While tour ≤10 and w=1
rand(n-2)+1→a
premr(n,t,a)→p1
If p1≠1 Then
p1→l1[1]
For i,1,s,1
mod(l1[i]^2,n)→l1[1+i]
EndFor
mod(l1+1,n)→l1
If product(l1)≠0 Then
0→w
EndIf
EndIf
tour+1→tour
EndWhile
ClrIo :Disp n
If w=1 Then
Disp ''est premier''
Else
Disp ''est compose''
EndIf
EndPrgm

Programme DEUEXPO

TI-83	Casio	TI-92
N-1→W 0→S While fPart(W/2)=0 S+1→S W/2→W End	N-1→W 0→S While Frac(W/2)=0 S+1→S W/2→W WhileEnd	deuexpo(n) Func n-1→w :0→s While fPart(w/2)=0 s+1→s :w/2→w EndWhile s :endFunc

Exemple d'écran :