

Thème : arithmétique

Cet énoncé est tiré de l'exercice-jury proposé aux candidat(e)s le 27 Juin 2005, lors de la deuxième épreuve orale (épreuve sur dossier) du Capes Externe de mathématiques.

Pour obtenir l'énoncé exact (ainsi que la production demandée au candidat), on se reportera au site officiel du jury, à l'adresse <http://capes-math.org/>

L'exercice proposé au candidat

Pour tout entier n non nul, on considère les nombres :

$$a_n = 4 \times 10^n - 1, \quad b_n = 2 \times 10^n - 1, \quad c_n = 2 \times 10^n + 1$$

1. Calculer $a_1, b_1, c_1, a_2, b_2, c_2, a_3, b_3, c_3$.
2. Combien les écritures décimales des nombres a_n et c_n ont-elles de chiffres ?
Montrer que a_n et c_n sont divisibles par 3.
3. Montrer, en utilisant la liste des nombres premiers inférieurs à 100 donnée ci-dessous, que b_3 est premier.
4. Montrer que, pour tout entier naturel non nul n , $b_n \times c_n = a_{2n}$.
En déduire la décomposition en produit de facteurs premiers de a_6 .
5. Montrer que $\text{PGCD}(b_n, c_n) = \text{PGCD}(c_n, 2)$.
En déduire que b_n et c_n sont premiers entre eux.

Liste des nombres premiers inférieurs ou égaux à 100.

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

Proposition de corrigé avec le Classpad 300

Corrigé de l'exercice

On définit les fonctions $n \mapsto a(n)$, $n \mapsto b(n)$ et $n \mapsto c(n)$.

On a immédiatement
$$\begin{cases} a(1) = 39 \\ b(1) = 19 \\ c(1) = 21 \end{cases} \begin{cases} a(2) = 399 \\ b(2) = 199 \\ c(2) = 201 \end{cases} \begin{cases} a(3) = 3999 \\ b(3) = 1999 \\ c(3) = 2001 \end{cases}$$

Bien sûr, on n'a pas vraiment besoin de la calculatrice pour ça, mais c'est l'occasion (fig1) de montrer comment appliquer une fonction à toute une liste de valeurs et obtenir la liste des résultats.

Les entiers ayant k chiffres sont les entiers de $[10^{k-1}, 10^k - 1]$.

Pour tout n ,
$$\begin{cases} 10^n \leq a_n < 10^{n+1} \\ 10^n \leq b_n < 10^{n+1} \\ 10^n \leq c_n < 10^{n+1} \end{cases} \text{ donc } a_n, b_n, c_n \text{ ont } n+1 \text{ chiffres.}$$

On sait que $10 \equiv 1 \pmod 3$ donc $10^n \equiv 1 \pmod 3$.

Ainsi
$$\begin{cases} a_n \equiv 4 - 1 \equiv 0 \pmod 3 \\ c_n \equiv 2 + 1 \equiv 0 \pmod 3 \end{cases} \text{ donc } a_n, c_n \text{ sont divisibles par } 3.$$

Plus précisément, avec $\begin{cases} 10^n = 100 \dots 0 & \text{où } 0 \text{ apparaît } n \text{ fois} \\ 10^n - 1 = 99 \dots 9 & \text{où } 9 \text{ apparaît } n \text{ fois} \end{cases}$, on a (voir fig1 avec $n = 20$) :

D'une part :
$$\begin{aligned} a_n &= 4 \times 10^n - 1 = 3 \times 10^n + (10^n - 1) = 399 \dots 9 && (\text{où } 9 \text{ apparaît } n \text{ fois}) \\ &= 3 \times 133 \dots 3 && (\text{où } 3 \text{ apparaît } n \text{ fois}). \end{aligned}$$

D'autre part :
$$\begin{aligned} c_n &= 2 \times 10^n + 1 = 200 \dots 01 && (\text{où } 0 \text{ apparaît } n - 1 \text{ fois}) \\ &= 3 \times 10^n - (10^n - 1) \\ &= 3(100 \dots 0 - 3 \dots 3) && (\text{où } 0 \text{ et } 3 \text{ apparaissent } n - 1 \text{ fois}) \\ &= 3 \times 66 \dots 67 && (\text{où } 6 \text{ apparaît } n - 1 \text{ fois}). \end{aligned}$$

Montrons que $b_3 = 1999$ est premier (en n'utilisant que la liste de l'énoncé). Supposons par l'absurde que ce ne soit pas le cas. Alors b_3 a un plus petit diviseur $p \geq 2$ (nécessairement premier).

En notant $b_3 = qp$, on a $q \geq p$ (minimalité de p) donc $p \leq \sqrt{b_3}$.

Or $\sqrt{b_3} = \sqrt{1999} \approx 44,71$ donc $p \leq 45$.

L'entier premier p est donc nécessairement (voir liste de l'énoncé) dans la liste $L = \{2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43\}$.

Il suffit de vérifier que b_3 n'est divisible par aucun des entiers de L pour aboutir à une contradiction (des critères classiques de divisibilité permettraient d'éliminer 2, 3, 5 et même 11).

On voit (fig2) comment calculer le reste dans la division de b_3 par chacun des entiers de L : aucun de ces restes n'est nul.

Conclusion : l'entier $b_3 = 1999$ est premier.

Cela est évidemment confirmé par l'instruction **factor**.

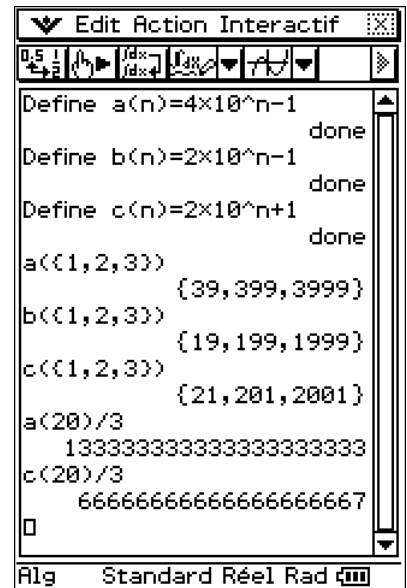


fig1 : les fonctions a, b, c

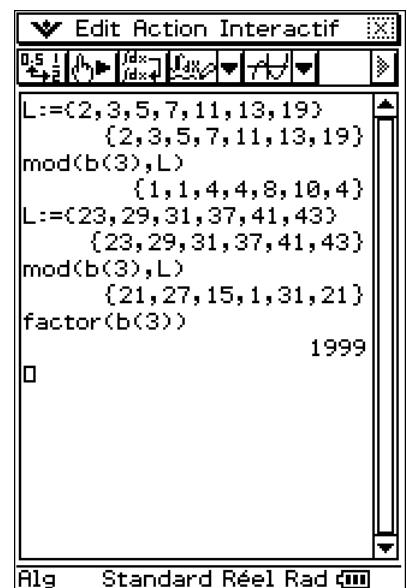


fig2 : l'entier b_3 est premier

Pour tout entier naturel non nul n , il est clair que :

$$b_n \times c_n = (2 \times 10^n - 1)(2 \times 10^n + 1) = 4 \times 10^{2n} - 1 = a_{2n}$$

En particulier $a_6 = 3999999$ s'écrit $a_6 = b_3 c_3 = 1999 \times 2001$.

On sait que $b_3 = 1999$ est premier, et $c_3 = 2001 = 3 \times 667$.

Or $667 = 23 \times 29$ (et 23, 29 sont premiers).

On en déduit $a_6 = 3 \times 23 \times 29 \times 1999$, décomposition en produit de facteurs premiers de a_6 .

On voit (fig3) comment l'instruction `factor` permet de factoriser (par exemple) les entiers a_n de $n = 6$ à $n = 13$.

Si factoriser a_6 était possible "à la main", il n'en va pas de même pour a_{13} par exemple, dont la factorisation fait apparaître un entier premier de 13 chiffres. Quand n est pair ($n = 2m$), c'est toujours un peu plus simple grâce à l'égalité $a_{2m} = b_m c_m$.

```

Edit Action Interactive
factor(a(6))
3*23*29*1999
factor(a(7))
3*13*1025641
factor(a(8))
3*7*59*113*2857
factor(a(9))
3*157*8492569
factor(a(10))
3*163*409*199999
factor(a(11))
3*107*1009*1234991
factor(a(12))
3*17*71*1657*666667
factor(a(13))
3*13*1025641025641

```

fig3 : factorisation des a_n

Nous allons maintenant montrer que b_n et c_n sont premiers entre eux.

Rappelons que, pour tous entiers m, n, q , on a l'égalité $\text{pgcd}(m, n) = \text{pgcd}(n, n - mq)$. Cela résulte du fait que les deux entiers m, n d'une part, et les deux entiers $n, n - mq$ d'autre part, ont exactement les mêmes diviseurs.

En particulier, on toujours l'égalité $\text{pgcd}(m, n) = \text{pgcd}(n, n - m)$.

On trouve donc $\text{pgcd}(b_n, c_n) = \text{pgcd}(c_n, c_n - b_n) = \text{pgcd}(c_n, 2) = 1$ (car c_n est impair).

Cela montre que les entiers b_n et c_n sont toujours premiers entre eux.

Un peu de programmation avec le Classpad

Dans le « travail demandé au candidat » le 27 juin 2005, il était demandé de « présenter un algorithme permettant d'obtenir le PGCD de deux entiers naturels non nuls ».

On peut toujours écrire au tableau un algorithme dans un « pseudo-langage », mais c'est préférable de le porter sur une calculatrice et de faire la démonstration qu'il fonctionne!

Nous allons donc implémenter l'algorithme d'Euclide (évidemment) sur le Classpad 300.

Commençons par en rappeler le principe : on veut calculer $\text{pgcd}(a, b)$, avec a, b dans \mathbb{Z} .

On suppose $b \neq 0$ (sinon $\text{pgcd}(a, b) = |a|$) et même $b > 0$ (quitte à remplacer b par $-b$).

Soit $a = bq_1 + r_1$ la division euclidienne de a par b : on sait que $0 \leq r_1 < b$.

On a $\text{pgcd}(a, b) = \text{pgcd}(b, a - bq_1) = \text{pgcd}(b, r_1)$. Si $r_1 = 0$ alors $\text{pgcd}(a, b) = b$.

Sinon, soit $b = r_1 q_2 + r_2$ la division euclidienne de b par r_1 . On a $0 \leq r_2 < r_1$.

Si $r_2 = 0$, alors $\text{pgcd}(a, b) = \text{pgcd}(b, r_1) = r_1$.

Sinon on divise r_1 par r_2 , et le procédé se poursuit.

On forme ainsi une suite $b > r_1 > r_2 > r_3 > \dots \geq 0$ strictement décroissante d'entiers.

On peut passer de r_k à r_{k+1} tant que $r_k \neq 0$.

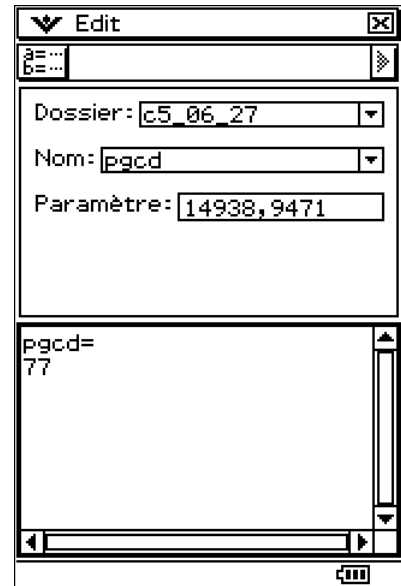
Cette suite de premier terme b est nécessairement finie.

Il existe donc un entier naturel n tel que $r_n > 0$ et $r_{n+1} = 0$. On a alors $\text{pgcd}(a, b) = r_n$.

Ainsi $\text{pgcd}(a, b)$ est le *dernier reste non nul* dans cette succession de divisions.

On voit ci-dessous la succession des divisions qui donnent $\text{pgcd}(14938, 9471) = 77$.

$$\begin{aligned}
 14938 &= 1 \cdot 9471 + 5467 \\
 9471 &= 1 \cdot 5467 + 4004 \\
 5467 &= 1 \cdot 4004 + 1463 \\
 4004 &= 2 \cdot 1463 + 1078 \\
 1463 &= 1 \cdot 1078 + 385 \\
 1078 &= 2 \cdot 385 + 308 \\
 385 &= 1 \cdot 308 + 77 \\
 308 &= 4 \cdot 77
 \end{aligned}$$



$\text{pgcd}(14938, 9471) = 77$

fig4 : le programme pgcd

fig5 : exemple d'utilisation

On voit (fig4) un programme très simple, appelé `pgcd`, et qui calcule le pgcd de deux entiers a et b donnés en arguments. Sur l'exemple (fig5), on retrouve $\text{pgcd}(14938, 9471) = 77$.

On peut améliorer le programme `pgcd` en lui faisant afficher les divisions successives.

La nouvelle version s'appelle `pgcd2` (voir fig6), et fait appel à un sous-programme d'affichage appelé `subpgcd` (fig7). On voit (fig8) un exemple d'utilisation du programme `pgcd2`, où il s'agit encore de calculer $\text{pgcd}(14938, 9471)$.

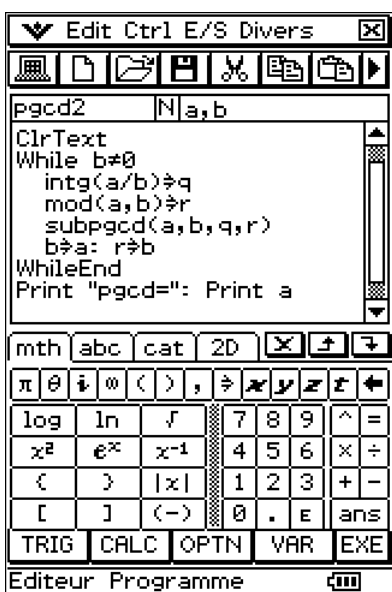


fig6 : programme pgcd2

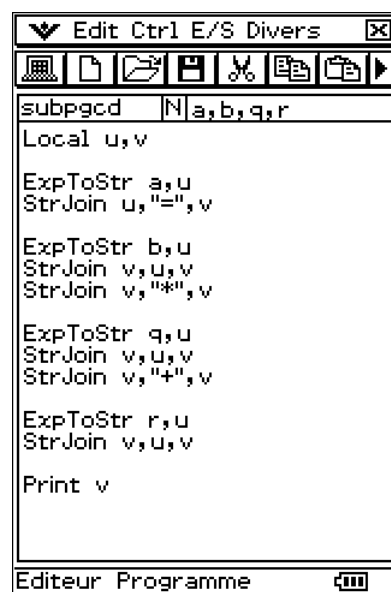


fig7 : sous-programme subpgcd

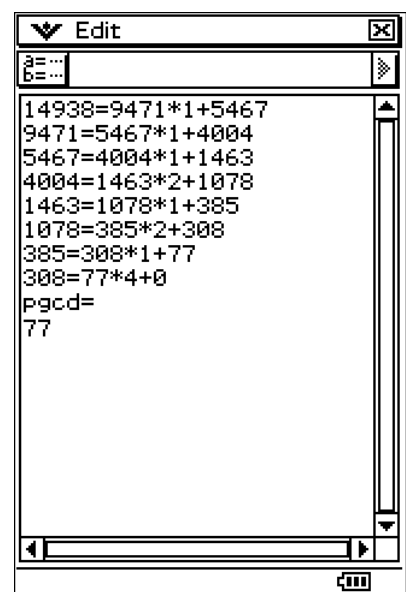


fig8 : exemple d'utilisation

Complément : l'algorithme du pgcd étendu

Soient a, b deux entiers relatifs non nuls. Alors il existe une infinité de couples (x, y) de \mathbb{Z}^2 tels que $ax + by = \text{pgcd}(a, b)$. Chacun d'eux est appelé un *couple de coefficients de Bezout* de (a, b) .

L'*algorithme du pgcd étendu* consiste à fournir, à partir du couple (a, b) , non seulement leur pgcd δ , mais en outre un couple (x, y) de \mathbb{Z}^2 tel que $ax + by = \delta$.

Notons E_d l'équation $ax + by = d$, où d est donné dans \mathbb{Z} , et où (x, y) est l'inconnue dans \mathbb{Z}^2 .

On doit donc trouver la valeur de $\delta = \text{pgcd}(a, b)$ et une solution (x, y) de E_δ .

On remarque que $(1, 0)$ est une solution de E_a et que $(0, 1)$ est une solution de E_b .

Soient α et β deux éléments de \mathbb{Z} , avec $\beta \neq 0$.

Soient (x_1, y_1) une solution de E_α et (x_2, y_2) une solution de E_β .

Soit $\alpha = q\beta + r$ la division euclidienne de α par β .

Les égalités $\begin{cases} ax_1 + by_1 = \alpha \\ ax_2 + by_2 = \beta \end{cases}$ impliquent $a(x_1 - qx_2) + b(y_1 - qy_2) = \alpha - q\beta = r$.

Autrement dit le couple $(x_3 = x_1 - qx_2, y_3 = y_1 - qy_2)$ est solution de E_r .

Si on applique l'idée précédente et l'algorithme d'Euclide au couple (a, b) , on va former, pour chacun des restes successifs r_k de cette méthode, une solution (x_k, y_k) de l'équation E_{r_k} .

Si r_n est le dernier reste non nul (donc $r_n = \delta = \text{pgcd}(a, b)$) alors on obtient une solution (x_n, y_n) de l'équation E_{r_n} , c'est-à-dire de l'équation $ax + by = \delta$.

Voici un programme baptisé `extpgcd` qui utilise cette méthode (fig9).

On a donné un exemple d'utilisation, avec $a = 14938$ et $b = 9471$.

On trouve $\text{pgcd}(a, b) = 77$, et aussi l'égalité $26a - 41b = 77$ (fig10).

```

extpgcd  N|a,b
Local r,s1,s2
ClrText
Print "{a,b}="
Print {a,b}
{1,0}↔s1: {0,1}↔s2
While b≠0
  intg(a/b)↔q
  mod(a,b)↔r
  b↔a: r↔b
  s1↔t: s2↔s1
  t-q*s2↔s2
WhileEnd
Print "pgcd(a,b)=d, d="
Print a
Print "ax+by=d, {x,y}="
Print s1
  
```

fig9 : programme extpgcd

```

Dossier: c5_06_27
Nom: extpgcd
Paramètre: 14938,9471

{a,b}={14938,9471}
pgcd(a,b)=d, d=77
ax+by=d, {x,y}={26,-41}
  
```

fig10 : exemple d'utilisation

Remarque : on rappelle l'existence, sur le Classpad 300, des instructions `gcd`, `lcm` pour calculer respectivement le pgcd et ppcm de deux entiers, ainsi que l'instruction `factor` pour factoriser un entier en produit de facteurs premiers.