la division des nombres ploutons
comme si vous y étiez

Cet avant-dernier article sur la division euclidienne vous permettra de faire connaissance avec les nombres amis et les nombres parfaits. Comment en arriver là ? Grâce à un petit détour par les ploutons. Rassurez-vous ce ne sont que de biens inoffensifs nombres entiers même si leur nom peut vous évoquer quelque étrange voracité animale.

L'exercice N° 24 demandait d'écrire un programme qui décompose un entier en produit de facteurs premiers. Nous vous présentons deux versions de ce programme. La première est celle qui vient immédiatement à l'esprit :

```
01 LIRE N
02 DIV ← 1
03 x
04 DIV ← DIV + 1
05 x
06 Q ← N/DIV
07 SI (Q ≠ Int (Q)) ALORS ALLER
   A 04
08 ECRIRE DIV
09 N ← Q
10 x
11 SI (N ≠ 1) ALORS ALLER A 06
12 x
13 ECRIRE 1
14 FIN
```

On fait afficher 1 lorsque la décomposition est terminée.

Ce programme a le mérite d'être simple. Il n'est pas besoin d'une étude attentive pour constater qu'il « fait des tas de choses inutiles ».

En effet, on divise le dernier quotient obtenu par tous les entiers qui lui sont inférieurs. Ainsi, 19 par exemple, est divisé successivement par 3, 4, 5, ..., 17.

Mode d'emploi			
N°	Instructions ou données	Touches	Affichage
01	Passer en mode programme	LRN	
02	Introduire le programme	.....	.....
03	Passer en mode « calcul »	LRN	
04	Initialiser le pointeur	RST	0.
05	Introduire le nombre à décomposer		N
	Afficher sa décomposition	R/S	DIV. et 1.
06	Pour une autre décomposition, aller en 05		

Affichage		Touches
Pas	Codes	
00	32 5	STO 5
01	01	1
02	32 1	STO 1
03	86 1	Lbl 1
04	01	1
05	34 1	SUM 1
06	86 2	Lbl 2
07	33 5	RCL 5
08	45	÷
09	33 1	RCL 1
10	85	=
11	32 7	STO 7
12	49	Int
13	−66	INV x=t
14	51 1	GTO 1
15	33 1	RCL 1
16	36	Pause
17	33 7	RCL 7
18	32 5	STO 5
19	01	1
20	−66	INV x=t
21	51 2	GTO 2
22	81	R/S
23	71	RST

Registres	
R1	DIV
R5	N
R7	Tests

18 puis 19 alors qu'il suffit de s'arrêter à 4 puisque $\sqrt{19} \simeq 4.36$ et Int ($\sqrt{19}$) = 4. Cette constatation conduit à un deuxième programme, bien plus rapide.

```
01 LIRE N
02 x
03 DIV ← 1
04 DIV ← DIV + 1
05 SI (DIV² > N) ALORS ALLER A 14
06 x
07 Q ← N/DIV
08 SI (Q ≠ Int (Q)) ALORS ALLER A 04
09 x
```

Mode d'emploi			
N°	Instructions ou données	Touches	Affichage
01	Passer en mode « programme »	LRN	
02	Introduire le programme	.....	.....
03	Passer en mode « calcul »	LRN	
04	Initialiser le pointeur	RST	0.
05	« nettoyer » l'affichage	CLR	0
06	Introduire le nombre N à décomposer Afficher sa décomposition	 R/S	N DIV et 1.
07	Pour une autre décomposition, aller en 05		

Affichage		
Pas	Codes	Touches
00	32 5	STO 5
01	01	1
02	32 1	STO 1
03	86 1	Lbl 1
04	01	1
05	34 1	SUM 1
06	33 1	RCL 1
07	23	x^2
08	32 7	STO 7
09	33 5	RCL 5
10	−76	INV x≥t
11	51 3	GTO 3
12	86 2	Lbl 2
13	33 5	RCL 5
14	45	÷
15	33 1	RCL 1
16	85	=
17	32 7	STO 7
18	49	int
19	−66	INV x=t
20	51 1	GTO 1
21	32 5	STO 5
22	33 1	RCL 1
23	36	Pause
24	51 2	GTO 2
25	86 3	Lbl 3
26	36	Pause
27	01	1
28	81	R/S
29	71	RST

```
10 ECRIRE DIV
11 N ← Q
12 ALLER A 07
13 x
14 ECRIRE N, 1
15 FIN
```

Les registres utilisés sont les mêmes que dans le programme précédent.

L'exercice n° 25 proposait un programme qui décompose un entier en produit de facteurs primaires. Il se différencie donc du précédent par un compteur, désigné par CPTR, qui enregistre le nombre d'occurences d'un même facteur premier. Afin de ne pas confondre à l'affichage un facteur premier et son exposant, ce dernier est précédé du signe "-". Ainsi, 2^3 sera affiché 2 puis -3.

```
01 LIRE N
02 DIV ← 1
03 x
04 DIV ← DIV + 1
05 CPTR ← 0
06 SI (DIV² > N) ALORS ALLER A 18 ; x FINI
07 x
08 Q ← N/DIV
09 SI (Q ≠ Int (Q)) ALORS ALLER A 14
10 CPTR ← CPTR + 1
11 N ← Q
12 ALLER A 08
13 x
14 SI (CPTR = 0) ALORS ALLER A 04
15 ECRIRE DIV, -CPTR
16 ALLER A 04
17 x
18 ECRIRE N, 1
19 FIN
```

Le mode d'emploi de ce programme est le même que celui du programme précédent, à ceci près que l'affichage donne à présent, un facteur premier, puis l'opposé de son exposant. La décomposition est terminée quand un 1 est affiché. Les registres utilisés sont

Affichage		
Pas	Codes	Touches
00	32 5	STO 5
01	01	1
02	32 1	STO 1
03	86 1	Lbl 1
04	01	1
05	34 1	SUM 1
06	00	0
07	32 2	STO 2
08	33 1	RCL 1
09	23	x^2
10	32 7	STO 7
11	33 5	RCL 5
12	−76	INV x≥t
13	51 4	GTO 4
14	86 2	lbl 2
15	33 5	RCL 5
16	45	÷
17	33 1	RCL 1
18	85	=
19	32 7	STO 7
20	49	Int
21	−66	INV x=t
22	51 3	GTO 3
23	32 5	STO 5
24	01	1
25	−34 2	INV SUM 2
26	51 2	GTO 2
27	86 3	Lbl 3
28	33 2	RCL 2
29	32 7	STO 7
30	00	0
31	66	x=t
32	51 1	GTO 1
33	33 1	RCL 1
34	36	Pause
35	33 2	RCL 2
36	36	Pause
37	51 1	GTO 1
38	86 4	Lbl 4
39	36	Pause
40	01	1
41	81	R/S
42	71	RST

les mêmes, avec en plus, R2 qui contient ici le compteur CPTR.

Voici quelques-unes des décompositions demandées :

216	$2^3.3^3$
991	991
3500	$2^2.5^3.7$
3850	$2.5^2.7.11$
2450	$2.5^2.7^2$
1080	$2^3.3^3.5$
18900	$2^2.3^3.5^2.7$
30024	$2^3.3^3.139$

Méfiez-vous des nombres amis

Passons maintenant aux ploutons, aux nombres amis et aux nombres parfaits

On appelle plouton un entier qui a plus de diviseurs que les nombres qui le précèdent.

Si a est un entier naturel, notons Da l'ensemble des diviseurs de a, et /Da/ le nombre d'éléments de cet ensemble. Ainsi, par exemple, D12 = { 1, 2, 3, 4, 6, 12 } et /D12/ = 6 puisque D12 a 6 éléments.

12 est un plouton, car il a 6 diviseurs, donc davantage que les entiers 1, 2, 3, ..., 11, placés avant lui dans la liste des entiers.

De même, 24 est un plouton.

EX. N° 26 : Ecrivez un programme qui affiche la liste des ploutons.

Utilisez ce programme pour établir la liste des ploutons inférieurs à 1000.

On peut démontrer que si $p_1, p_2, ..., p_r$ sont les facteurs premiers de la décomposition de a, avec les exposants $k_1, k_2, ..., k_r$, on a $/Da/ = (k_1 + 1) \times (k_2 + 1) \times ... \times (k_r + 1)$.
Ainsi,
$12 = 2^2.3^1$
et donc
$p_1 = 2, k_1 = 2$
$p_2 = 3, k_2 = 1$
par conséquent,
$/D12/ = (2 + 1) \times (1 + 1) = 6$

EX. N° 27 : Ecrivez un programme qui utilise cette propriété pour calculer et afficher la liste des ploutons.

Quel est le programme le plus rapide ?

Les nombres amis répondent eux à la définition suivante : soient a et b deux entiers, et posons da = Da - { a }. Par exemple, d12 = { 1, 2, 3, 4, 6 } alors que D12 = { 1, 2, 3, 4, 6, 12 } Nous dirons que a et b sont amis si, et seulement si la somme des éléments de da est égale à b et la somme des éléments de db est égale à a, ce que nous écrirons :

Sda = b et Sdb = a.

Ainsi, 220 et 284 sont amis. En effet :
D220 = { 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110, 220 }
d220 = { 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110 }
Sd220 = 1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284.

Vérifiez que Sd284 = 220.

EX. N° 28 : Ecrivez un programme qui calcule et affiche les nombres amis.

Les nombres amis sont rares et les programmes évidents qui permettent de les calculer sont très lents. Rechercher les nombres amis en imbriquant 2 boucles de calculs demande 4 000 000 d'opérations si on se limite aux nombres inférieurs à 2 000. On peut limiter le nombre de calculs nécessaires en remarquant que a et b sont amis si, et seulement si :

Sda = b et Sdb = a

Mais alors Sd(Sda) = Sdb = a. D'où l'algorithme :

. on décrit les entiers naturels à partir de 2, et pour chaque nombre, on calcule Sda ;

. si Sda < a, on passe à l'entier suivant, car alors Sda est un nombre qui a déjà été étudié précédemment ;

. sinon, on calcule Sd(Sda) et on le compare à a ;

S'ils sont égaux, a et b sont amis.

Même ainsi, le programme est très lent et il faudra beaucoup de patience pour avoir quelques résultats. Quels sont les deux premiers couples affichés ? Pourquoi ?

Le programme précédent vous a permis de constater que 6 et 28 sont amis d'eux-mêmes. On a en effet, Sd6 = 6 et Sd28 = 28.

6 et 28 sont appelés des nombres parfaits. Plus précisément, on dit qu'un nombre entier naturel est parfait lorsqu'il est ami de lui-même. Ou encore : un entier a est parfait lorsqu'il est égal à Sda.

EX. N° 29 : Ecrivez un programme qui calcule et affiche les nombres parfaits.

Ce programme, s'il calcule Sda pour les entiers successifs est également très lent. La recherche des entiers parfaits parmi les seuls entiers pairs vous permettra de gagner beaucoup de temps, mais une bonne dose de patience vous sera tout de même nécessaire.

Euler, une fois de plus, vient à notre secours pour nous éviter une crise de nerfs.

Il a démontré, en effet, qu'un entier naturel de la forme $2^{p-1}.(2^p-1)$ est parfait lorsque p et $2^p - 1$ sont premiers. Ainsi, pour p = 2, on a : $2^{2-1}.(2^2 - 1) = 2.(4 - 1) = 6$ qui est parfait, et on a bien p = 2 et $2^p - 1 = 3$ premiers.

Les nombres de la forme $2^p - 1$ avec p premier sont appelés les nombres de Mersenne.

EX. N° 30: Ecrivez un programme qui affiche les nombres parfaits en les calculant par les nombres de Mersenne.

Ce programme devra donc engendrer les entiers premiers, calculer $2^p - 1$ et tester si ce nombre est premier. Si c'est le cas, il affiche $2^{p-1} \times (2^p - 1)$.

Comme vous pourrez le constater, on obtient les nombres parfaits incomparablement plus vite que par le programme précédent, mais la capacité du calculateur est vite dépassée, les nombres parfaits étant encore moins nombreux que les nombres amis.

Cet exercice termine la série que nous vous présentons depuis quelques mois sur la division euclidienne et ses prolongements.

Le mois prochain, le dernier article présentera des solutions à ces cinq exercices.

Christophe Haro