ARITHMÉTIQUE ET PREMIERS ALGORITHMES

1 Ensembles de nombres

Un *ensemble* d'objets peut être décrit de plusieurs façons. D'abord on peut *énumérer* ses éléments, en les plaçant entre accolades :

$$\mathrm{A} = \{1\,;\,4\,;\,7\,;\,8\,;\,12\,;\,15\,;\,17\,;\,20\,;\,22\}$$
$$\mathrm{B} = \{0\,;\,2\,;\,4\,;\,8\,;\,16\,;\,32\}.$$

Dans un ensemble, l'ordre des éléments *n'a pas d'importance*, et il n'y a *jamais de répétition*. Ainsi on aurait également pu écrire $\mathrm{B} = \{32\,;\,0\,;\,4\,;\,16\,;\,8\,;\,2\}$ ou $\mathrm{B} = \{0\,;\,0\,;\,4\,;\,16\,;\,8\,;\,2\,;\,32\,;\,2\,;\,8\}$: c'est la même chose.

On peut ensuite définir un ensemble à partir d'un autre soit par *transformation*

$$\mathrm{C} = \{x^2 \mid x \in \mathbf{N}\}$$

« l'ensemble des x^2 obtenus *lorsque* x *parcourt* l'ensemble $\mathbf{N}$ »

soit par *sélection*

$$\mathrm{C} = \{y \in \mathbf{N} \mid \text{il existe } x \in \mathbf{N} \text{ tel que } y = x^2\}$$

« l'ensemble des y dans $\mathbf{N}$ *pour lesquels* il existe un x dans $\mathbf{N}$ tel que $y = x^2$ ».

Ici C est l'ensemble des *carrés*, qu'on aurait pu aussi écrire

$$\mathrm{C} = \{0\,;\,1\,;\,4\,;\,9\,;\,16\,;\,25\,;\,36\,;\,49\,;\,64\,;\,81\,;\,100\,;\,121\,;\,\ldots\}$$

mais évidemment l'usage des *pointillés* suppose que la personne qui lit va comprendre la suite logique.

Ici l'ensemble C a été décrit de trois manières différentes. Ce n'est pas toujours possible : le plus souvent, il n'y a qu'une manière qui marche pour décrire (facilement) un ensemble.

Enfin, on peut décrire un ensemble à partir d'autres ensembles en utilisant des *opérations ensemblistes*. Il y en a trois : $\cup$, $\cap$ et $-$.

DÉFINITIONS (OPÉRATIONS ENSEMBLISTES).

i) L'*union* de A et B est l'ensemble des éléments qui sont soit dans A, soit dans B, soit dans les deux à la fois. On la note avec le symbole $\cup$:

$$\mathrm{A} \cup \mathrm{B} = \{0\,;\,1\,;\,2\,;\,4\,;\,7\,;\,8\,;\,12\,;\,15\,;\,16\,;\,17\,;\,20\,;\,22\,;\,32\}.$$

On rappelle qu'il n'y a *pas de répétition* dans un ensemble. Bien sûr $\mathrm{A} \cup \mathrm{B}$ est la même chose que $\mathrm{B} \cup \mathrm{A}$.

ii) L'*intersection* de A et B est l'ensemble des éléments qui sont à la fois dans A et dans B. On la note avec le symbole $\cap$:

$$\mathrm{A} \cap \mathrm{B} = \{4\,;\,8\}.$$

Là encore, $\mathrm{A} \cap \mathrm{B}$ est la même chose que $\mathrm{B} \cap \mathrm{A}$.

iii) La *différence* de A et B est l'ensemble des éléments de A qui ne sont pas dans B. On la note avec le symbole $-$:

$$\mathrm{A} - \mathrm{B} = \{1\,;\,7\,;\,12\,;\,15\,;\,17\,;\,20\,;\,22\} \qquad \mathrm{B} - \mathrm{A} = \{0\,;\,2\,;\,16\,;\,32\}.$$

On aura remarqué que $\mathrm{A} - \mathrm{B}$ n'est pas la même chose que $\mathrm{B} - \mathrm{A}$.

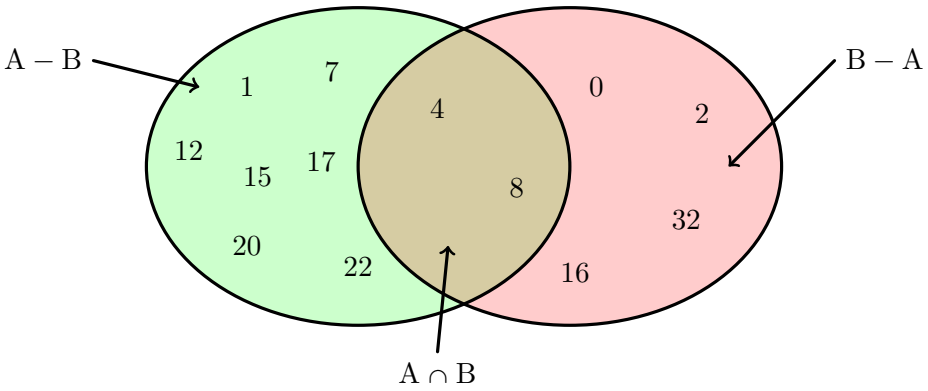
Par exemple on a déjà rencontré $\mathbf{R} - \{0\}$: « tous les nombres sauf zéro ».

Propriété (décomposition d'une réunion).

Par définition, on peut toujours partager l'union $\mathrm{A} \cup \mathrm{B}$ en trois : les éléments qui sont dans A mais pas dans B, les éléments qui sont à la fois dans A et dans B, et les éléments qui sont dans B mais pas dans A. Ce qu'on écrit

$$\mathrm{A} \cup \mathrm{B} = (\mathrm{A} - \mathrm{B}) \cup (\mathrm{A} \cap \mathrm{B}) \cup (\mathrm{B} - \mathrm{A}).$$

On peut représenter cela schématiquement par un *diagramme de Venn* (les ensembles sont représentés par des *patatoïdes* qui se chevauchent) :



Les trois morceaux sont deux à deux *disjoints* (ils n'ont pas d'éléments en commun).

Terminons ce paragraphe avec des notions elles-aussi déjà rencontrées.

Définitions.

i) L'*ensemble vide* est celui qui ne contient aucun élément : on le note $\varnothing$.

ii) On écrit $x \in \mathrm{A}$ pour dire que l'objet x *appartient à* (ou *est un élément de*) l'ensemble A.

iii) On dit que deux ensembles A et B sont *disjoints* lorsqu'ils n'ont pas d'élément en commun, c'est-à-dire lorsque $\mathrm{A} \cap \mathrm{B} = \varnothing$.

2 Quelques programmes en Python

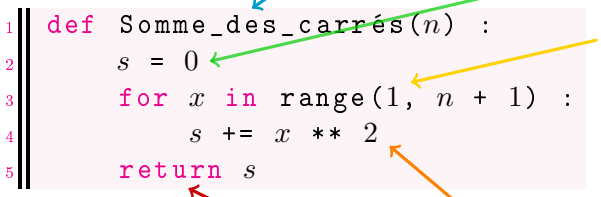
Dans un *environnement* de programmation en Python, il y a toujours *deux* zones : celle pour *écrire* les programmes (que nous représenterons sur fond *rose*) et celle pour les *utiliser* (on l'appelle la *console*, et nous la représenterons sur fond *mauve*). La console peut également être utilisée pour faire n'importe quel calcul :

```
>>> 1 + 2 + 3 + 4
10
```

Voici notre premier programme. Il calcule la somme des n premiers carrés (sans compter $0^2 = 0$), c'est-à-dire

$$s = 1^2 + 2^2 + 3^2 + 4^2 + 5^2 + \ldots + n^2.$$

Décortiquons-le.



```
def Somme_des_carrés(n) :
    s = 0
    for x in range(1, n + 1) :
        s += x ** 2
    return s
```

Un programme Python commence toujours ainsi : le mot-clé `def`, suivi du nom du programme, suivi de parenthèses, et enfin les deux points. Entre les parenthèses on place les *entrées* du programme : ici n, car le résultat du programme *dépend* de n.

On *initialise* une *variable* s à la valeur 0.

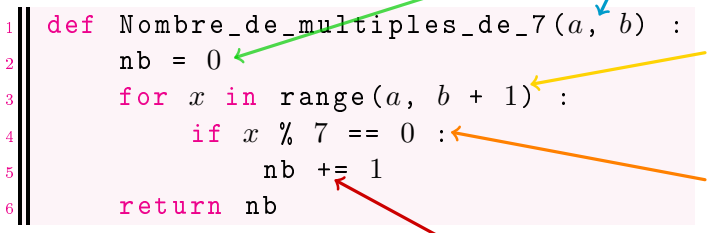
Pour *répéter* une action, on utilise une *boucle* : elle commence avec le mot-clé `for`, suivi d'un nom de variable, ici x (« le compteur »), suivi du mot-clé `in`, suivi de l'ensemble des valeurs que prendra x, et enfin des deux-points. Ici l'ensemble est `range(1, n + 1)` qui signifie « les nombres de 1 (inclus) à $n+1$ (exclu) ». Autrement dit : x vaut successivement 1, 2, 3, etc., n.

On augmente la variable s de la valeur x^2.

Le résultat du programme est indiqué par le mot-clé `return`.

Pour calculer avec les *entiers*, on dispose en Python de six opérations, chacune proposée en deux versions : `+` et `+=`, `-` et `-=`, `*` et `*=`, `//` et `//=` (quotient dans la division euclidienne), `%` et `%=` (reste dans la division euclidienne), enfin `*` et `**=` (pour les puissances). La première version sert aux calculs, la deuxième version sert à *modifier* une variable. Ainsi « augmenter la variable v de la valeur 3 » s'écrira `v += 3`.

Deuxième programme : on souhaite compter le nombre de multiples de 7 entre a et b (inclus).



```
def Nombre_de_multiples_de_7(a, b) :
    nb = 0
    for x in range(a, b + 1) :
        if x % 7 == 0 :
            nb += 1
    return nb
```

Le résultat du programme dépend de a et de b : ce sont les deux *entrées* qu'on place entre les parenthèses.

La variable `nb` représente le nombre de multiples de 7 qu'on a trouvés. Au début, on n'en a pas encore trouvé donc on l'initialise à la valeur 0.

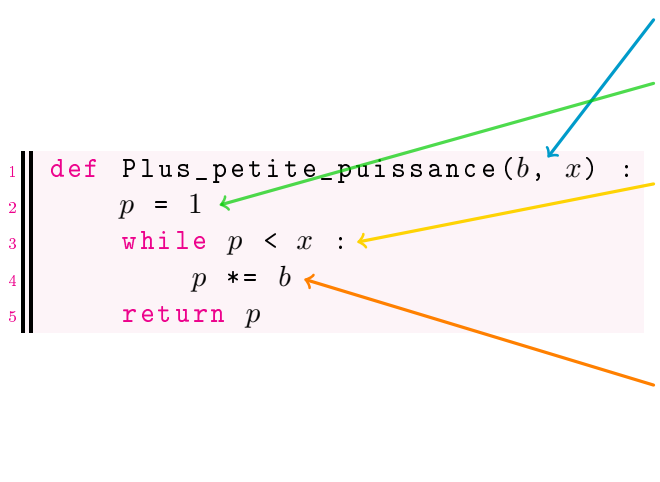
On essaie tous les x de a (inclus) jusqu'à $b+1$ (exclu), c'est-à-dire que x vaut successivement a, $a+1$, $a+2$, etc., jusqu'à b.

Pour savoir si x est divisible par 7, on regarde si le reste dans la division euclidienne est égal à zéro. Le mot-clé `if` indique que les actions suivantes ne sont exécutées que *si le test à réussi*. Le *test*, c'est la condition écrite juste après `if`.

À chaque multiple de 7 qu'on trouve, on augmente de 1 la variable `nb`.

Attention à la différence entre `=` qui sert à définir une variable, et `==` qui sert à tester si deux choses sont égales. Attention aussi aux alinéas : ils sont obligatoirement formés de *quatre* espaces, et indiquent les *blocs* du programme. Dans cet exemple, les lignes 2 à 6 ont un alinéa car elles font *toutes* partie du programme `Nombre_de_multiples_de_7`. Les lignes 4 et 5 ont un deuxième alinéa, car elles font partie de la boucle `for` (ce sont donc les lignes qui sont répétées). Enfin, la ligne 5 possède un troisième alinéa car elle fait partie de la structure conditionnelle (c'est la ligne qui est exécutée seulement si le test réussi).

Troisième programme : on cherche la plus petite puissance de b supérieure ou égale à un x donné.



Pour *utiliser* un programme, on va dans la console. On écrit le nom du programme, suivi de parenthèses, et des valeurs qu'on souhaite donner aux entrées. Par exemple, pour connaître la plus petite puissance de 2 qui dépasse 2000, on écrit ceci.

```
>>> Plus_petite_puissance(2, 2000)
2048
```

3 Multiples et réduction au même dénominateur

Définition (multiple).
On dit que x est un *multiple* de d (ou qu'il est *divisible* par d) lorsqu'il se trouve dans la table de multiplication par d. Autrement dit, lorsqu'on peut obtenir x en multipliant d par un *entier*. On appelle $\mathscr{M}(d)$ l'ensemble des multiples de d.

On peut considérer *tous* les multiples

$$\{\dots ; -12 ; -9 ; -6 ; -3 ; 0 ; 3 ; 6 ; 9 ; 12 ; 15 ; \dots\}$$

ou seulement (c'est ce que nous ferons ici) les multiples *positifs*

$$\mathscr{M}(3) = \{0 ; 3 ; 6 ; 9 ; 12 ; \dots\} = \{3 \times k \mid k \in \mathbf{N}\}.$$

Propriétés.
i) L'ensemble des multiples de d est toujours *infini*, sauf pour $d = 0$: $\mathscr{M}(0) = \{0\}$.
ii) On passe d'un multiple de d au suivant en ajoutant d.

$$0 \xrightarrow{+3} 3 \xrightarrow{+3} 6 \xrightarrow{+3} 9 \xrightarrow{+3} 12 \xrightarrow{+3} 15 \xrightarrow{+3} 18 \xrightarrow{+3} 21 \xrightarrow{+3} 24 \xrightarrow{+3} \dots$$

On a en fait déjà vu cette propriété avec les tables de multiplication (voir la leçon **2**).

Définition (plus petit commun multiple).
Soient $a, b \in \mathbf{N} - \{0\}$. Leur PPCM est le plus petit nombre *strictement positif* qui est à la fois un multiple de a et un multiple de b. Autrement dit

$$\mathrm{PPCM}(a ; b) = \min\left((\mathscr{M}(a) \cap \mathscr{M}(b)) - \{0\}\right).$$

Lorsque $a = 0$ ou $b = 0$, on *convient* que $\mathrm{PPCM}(a ; b) = 0$. C'est donc le *seul* cas où le PPCM est nul.

Le PPCM est le *meilleur* dénominateur commun qu'on puisse trouver à deux fractions. Lorsque a et b sont petits, on le trouve en explorant leurs tables de multiplication. Dans le cas général, on utilise l'algorithme d'Euclide, qu'on verra un peu plus loin dans cette leçon.

à connaître

Exemple 1 : calculer $\frac{1}{18}+\frac{5}{42}$.

On écrit « en parallèle » les tables de 18 et de 42, et on s'arrête dès qu'on a trouvé un multiple en commun (c'est donc forcément le plus petit, puisqu'on les cherche *dans l'ordre*).

$$\begin{array}{l} 18\times 1 = 18 \\ 18\times 2 = 36 \\ 18\times 3 = 54 \\ 18\times 4 = 72 \\ 18\times 5 = 90 \\ 18\times 6 = 108 \\ \boxed{18\times 7 = 126} \end{array} \qquad \begin{array}{l} 42\times 1 = 42 \\ 42\times 2 = 84 \\ \boxed{42\times 3 = 126} \end{array}$$

Ainsi on peut prendre 126 comme dénominateur commun et on trouve

$$\frac{1\times 7}{18\times 7}+\frac{5\times 3}{42\times 3}=\frac{7}{126}+\frac{15}{126}=\frac{22}{126}.$$

Le résultat se simplifie, on verra cela dans les paragraphes suivants.

à connaître

Exemple 2 : calculer $\frac{1}{4}+\frac{1}{5}-\frac{1}{6}$.

La même méthode marche pour trois nombres (ou plus) : on écrit les multiples de 4, de 5 et de 6 dans l'ordre.

(de 4 en 4)	4	8	12	16	20	24	28	32	36	40	44	48	52	56	$\underset{4\times 15}{60}$
(de 5 en 5)	5	10	15	20	25	30	35	40	45	50	55	$\underset{5\times 12}{60}$			
(de 6 en 6)	6	12	18	24	30	36	42	48	54	$\underset{6\times 10}{60}$					

Ainsi on peut prendre 60 comme dénominateur commun et on trouve

$$\frac{1\times 15}{4\times 15}+\frac{1\times 12}{5\times 12}-\frac{1\times 10}{6\times 10}=\frac{15}{60}+\frac{12}{60}-\frac{10}{60}=\frac{17}{60}.$$

Écrivons le programme qui calcule le PPCM de deux entiers. Je l'appelle « naïf » parce que nous verrons un algorithme bien plus efficace dans deux paragraphes.

```
1  def PPCM_naïf(a, b) :
2      m_a = a ; m_b = b
3      while m_a != m_b :
4          if m_a < m_b :
5              m_a += a
6          else :
7              m_b += b
8      return m_a
```

La variable m_a représente les multiples de a et la variable m_b représente les multiples de b.

On continue *tant qu*'on n'a pas trouvé deux multiples égaux (ce sera donc le multiple « commun »).

On parcourt les multiples de a et de b « en même temps », c'est-à-dire qu'à chaque étape on avance dans la table de multiplication de a ou de b, selon où on a le « plus petit score ».

Le mot-clé `else` indique ce que l'on doit faire lorsque le test (après le `if`) a échoué. Ainsi à chaque étape, *soit* on augmente m_a, *soit* on augmente m_b (mais jamais les deux).

```
>>> PPCM_naïf(15, 25)
75
```

4 Critères de divisibilité

Propriétés (critères de divisibilité par 2, par 5 et par 10).
Ce sont les critères basés sur le *dernier* chiffre.

i) Un nombre est divisible par 2 si et seulement si son dernier chiffre est 0, 2, 4, 6, ou 8.

ii) Un nombre est divisible par 5 si et seulement si son dernier chiffre est 0 ou 5.

iii) Un nombre est divisible par 10 si et seulement si son dernier chiffre est 0.

Preuve. C'est la même démonstration pour les trois cas. On prend un nombre et on sépare son « dernier chiffre »

$$17\,928 = 1\,792 \times 10 + 8.$$

Puisque 10 est divisible par 2, on peut ré-écrire

$$17\,928 = 1\,792 \times 10 + 8 = (1\,792 \times 5) \times 2 + 8$$

et on voit donc que 17 928 est divisible par 2 puisqu'on peut aussi factoriser 2 dans son dernier chiffre 8. En revanche le même calcul donne

$$17\,928 = 1\,792 \times 10 + 8 = (1\,792 \times 2) \times 5 + 8$$

ce qui montre que 17 928 n'est pas divisible par 5 puisqu'on ne peut pas factoriser 5 dans 8. **C.Q.F.D.**

Propriétés (critères de divisibilité par les puissances de 2, de 5 et de 10).
Ce sont des extensions des critères précédents.

i) Un nombre est divisible par $4 = 2^2$ si et seulement si ses *deux* derniers chiffres forment un multiple de 4. Il est divisible par $8 = 2^3$ si et seulement si ses *trois* derniers chiffres forment un multiple de 8.

ii) Un nombre est divisible par $25 = 5^2$ si et seulement si ses *deux* derniers chiffres forment un multiple de 25, c'est-à-dire sont 00 ou 25 ou 50 ou 75. Il est divisible par $125 = 5^3$ si et seulement si ses *trois* derniers chiffres forment un multiple de 125, c'est-à-dire sont 000 ou 125 ou 250 ou 375 ou 500 ou 625 ou 750 ou 875.

iii) Un nombre est divisible par $100 = 10^2$ si et seulement si ses *deux* derniers chiffres forment un multiple de 100, c'est-à-dire s'il se termine par 00. Il est divisible par $1\,000 = 10^3$ si et seulement si ses *trois* derniers chiffres forment un multiple de 1 000, c'est-à-dire sont 000.

Remarque : on comparera attentivement le critère de divisibilité par 4 et les petites fractions en *quarts*, ainsi que le critère de divisibilité par 8 et les petites fractions en *huitièmes*, dans la leçon **13**.

Propriétés (critères de divisibilité par 3, par 9 et par 11).
Ce sont les critères basés sur la *somme* des chiffres.

i) Un nombre est divisible par 3 si et seulement si la somme de ses chiffres est un multiple de 3.

ii) Un nombre est divisible par 9 si et seulement si la somme de ses chiffres est un multiple de 9.

iii) Un nombre est divisible par 11 si et seulement si la somme de ses chiffres de rangs pairs et la somme de ses chiffres de rangs impairs diffèrent d'un multiple de 11.

Preuve de la règle ii). Prenons un nombre à quatre chiffres, qu'on écrit $abcd$. Ce nombre est donc égal à

$$1000 \times a + 100 \times b + 10 \times c + d.$$

On va essayer de mettre 9 en facteur dans cette écriture :

$$1000 \times a + 100 \times b + 10 \times c + d = 999a + a + 99b + b + 9c + c + d = (999a + 99b + 9c) + (a + b + c + d)$$

$$= 9 \times (111a + 11b + c) + (a + b + c + d).$$

La question est donc de savoir si 9 se met aussi en facteur dans $a + b + c + d$ (c'est-à-dire la somme des chiffres) : si c'est le cas, le nombre est divisible par 9, sinon il ne l'est pas. **C.Q.F.D.**

Propriétés (critères de divisibilité par 6 et par 12).

Ce sont les critères qui *se scindent en deux*.

i) Un nombre est divisible par 6 si et seulement si il est divisible à la fois par 2 et par 3.

ii) Un nombre est divisible par 12 si et seulement si il est divisible à la fois par 3 et par 4.

iii) Plus généralement, lorsque d_1 et d_2 sont *premiers entre eux* (voir le paragraphe suivant) : un nombre est divisible par $d_1 \times d_2$ si et seulement si il est divisible à la fois par d_1 et par d_2.

5 Diviseurs et simplification des fractions

Définition (diviseur).

On dit que d est un *diviseur* de x lorsque la division euclidienne de x par d « tombe juste », c'est-à-dire donne un reste nul :

$$x = d \times q + \underbrace{r}_{=0}.$$

Ainsi « d est un diviseur de x » est la même chose que « x est un multiple de d » ou « x est divisible par d ». On note $\mathscr{D}(d)$ l'ensemble des diviseurs de d.

Comme pour les multiples, on peut considérer *tous* les diviseurs

$$\{-12\,;\,-6\,;\,-4\,;\,-3\,;\,-2\,;\,-1\,;\,1\,;\,2\,;\,3\,;\,4\,;\,6\,;12\}$$

ou seulement (c'est ce que nous ferons) les diviseurs *positifs*

$$\mathscr{D}(12) = \{1\,;\,2\,;\,3\,;\,4\,;\,6\,;\,12\}.$$

L'ensemble des diviseurs de x est toujours un ensemble *fini*, sauf lorsque $x = 0$ (par convention *tous* les nombres sont des diviseurs de zéro).

Exemple : trouver tous les diviseurs de 100.

On trouve les diviseurs *par deux* : si d est un diviseur de x, alors x/d est aussi un diviseur de x. On essaie *tous* les nombres, dans l'ordre, à partir de 1 :

à connaître

$$\begin{aligned} 100 = \quad & 1 \times 100 \\ & 2 \times 50 \\ & 3 \text{ ne marche pas} \\ & 4 \times 25 \\ & 5 \times 20 \\ & 6,\ 7,\ 8,\ 9 \text{ ne marchent pas} \\ & 10 \times 10 \end{aligned}$$

et on s'arrête parce « les nombres de droite ont rejoint les nombres de gauche ». On trouve

$$\mathscr{D}(100) = \{1\,;\,2\,;\,4\,;\,5\,;\,10\,;\,20\,;\,25\,;\,50\,;\,100\}.$$

Définition (plus grand commun diviseur).

i) Soient $a, b \in \mathbf{N} - \{0\}$. Le PGCD de a et b est le plus grand nombre positif qui est à la fois un diviseur de a et un diviseur de b, c'est-à-dire

$$\mathrm{PGCD}(a\,;\,b) = \max\left(\mathscr{D}(a) \cap \mathscr{D}(b)\right).$$

C'est le plus grand nombre par lequel on peut *simplifier* la fraction a/b.

ii) On dit que a et b sont *premiers entre eux* lorsque leur PGCD est égal à 1. Dans ce cas, on dit que la fraction a/b est *irréductible* (on ne peut pas la simplifier).

Comme *tous* les nombres sont des diviseurs de zéro, on convient que $\mathrm{PGCD}(a\,;\,0) = a$ et $\mathrm{PGCD}(0\,;\,b) = b$. Plus subtil, on convient que $\mathrm{PGCD}(0\,;\,0) = 0$.

à connaître

Exemple 1 : justifier que la fraction 55/18 *est irréductible.*

On a $55 = 1 \times 55 = 5 \times 11$ (et rien d'autre) donc les diviseurs de 55 sont 1, 5, 11 et 55. De même on a $18 = 1 \times 18 = 2 \times 9 = 3 \times 6$ (et rien d'autre) donc les diviseurs de 18 sont 1, 2, 3, 6, 9 et 18. Le seul diviseur commun à 55 et 18 est donc 1, et la fraction 55/18 est irréductible.

à connaître

Exemple 2 : soit n un entier strictement positif. Prouver que la fraction $\dfrac{n}{n+1}$ *est irréductible.*

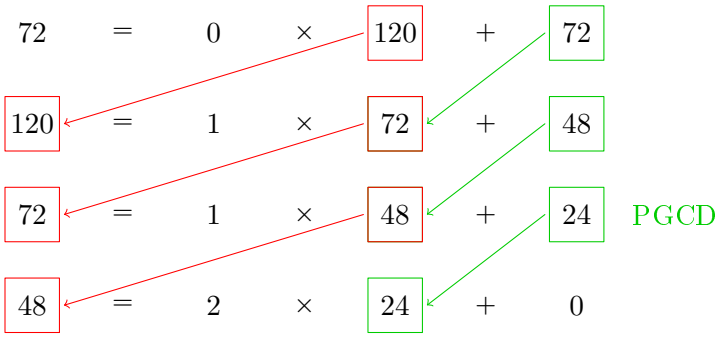
Soit d un diviseur commun à n et $n+1$. Quand on additionne (ou soustrait) deux multiples de d, on obtient encore un multiple de d : si $n = k \times d$ et $n + 1 = k' \times d$, alors

$$(n+1) - n = k' \times d - k \times d = (k' - k) \times d.$$

Donc $(n+1) - n = n + 1 - n = 1$ est lui aussi divisible par d. Mais le seul diviseur de 1 est 1, donc la seule possibilité est $d = 1$. Ainsi $\mathrm{PGCD}(n\,;\,n+1) = 1$ et la fraction est irréductible.

6 Algorithme d'Euclide

Pour trouver le PGCD de deux nombres, on effectue des divisions euclidiennes successives (à chaque étape, l'ancien diviseur devient le dividende et l'ancien reste devient le diviseur) jusqu'à obtenir un reste égal à zéro. Le PGCD est alors le reste de l'étape précédente. Prenons $a = 72$ et $b = 120$. On a



Remarque : lorsque a est plus petit que b, la première étape de l'algorithme d'Euclide ne fait qu'*échanger* le dividende et le diviseur.

Traduisons cela en Python. On ajoute une étape (avec des valeurs absolues) pour éliminer les signes éventuels de a et de b : cela ne change pas le PGCD.

```
def PGCD(a, b) :
    a = abs(a) ; b = abs(b)
    while b > 0 :
        (a, b) = (b, a % b)
    return a
```

L'avantage de l'algorithme d'Euclide est qu'il peut s'appliquer à des nombres très grands (jusqu'à plusieurs millions de chiffres).

```
>>> PGCD(9840913355, 654087980820)
485
```

Revenons maintenant au PPCM. Il est lié au PGCD par la propriété suivante.

Propriété (relation entre le PGCD et le PPCM).
Soient a et b deux nombres entiers.

i) Lorsque a et b sont positifs, on a

$$\mathrm{PGCD}(a\,;b)\times\mathrm{PPCM}(a\,;b)=a\times b.$$

Lorsque a et b sont de signes quelconques, la même relation est vraie, mais il faut ajouter des valeurs absolues pour retirer les signes.

ii) En particulier lorsque si a et b sont différents de zéro, on peut déduire le PPCM du PGCD en divisant :

$$\mathrm{PPCM}(a\,;b)=\frac{a\times b}{\mathrm{PGCD}(a\,;b)}$$

avec la même remarque sur les signes.

Et comme le PPCM est nul lorsqu'un des deux nombres est nul, on obtient l'algorithme suivant.

```
def PPCM(a, b) :
    if a == 0 or b == 0 :
        return 0
    else :
        return abs(a * b) // PGCD(a, b)
```

```
>>> PPCM(9840913355, 654087980820)
13271800300611385260
```

Rappelons que le PGCD permet de rendre les fractions irréductibles. Écrivons par exemple le programme qui simplifie (au maximum) la fraction a/b.

```
def Simplifier(a, b) :
    d = PGCD(a, b)
    return (a // d, b // d)
```

```
>>> Simplifier(1001, 455)
(11, 5)
```

et donc $\dfrac{1001}{455}=\dfrac{11}{5}$ et cette dernière fraction est irréductible.

7 Nombres premiers

Définitions (nombres premiers, nombres composés).

i) On dit qu'un entier $x \in \mathbf{N} - \{0\,;\,1\}$ est *premier* lorsque ses seuls diviseurs sont 1 et lui-même ($\mathscr{D}(x) = \{1\,;\,x\}$), c'est-à-dire lorsque la seule manière de le *factoriser* est d'écrire $1 \times x$ ou $x \times 1$.

ii) On dit qu'un entier $x \in \mathbf{N} - \{0\,;\,1\}$ est *composé* lorsqu'il admet au moins un diviseur « non trivial » (c'est-à-dire autre que 1 et lui-même), et qu'il y a donc une factorisation $x = a \times b$ avec a et b différents de 1 et x.

Attention , 0 et 1 sont à part : *ils ne sont ni composés ni premiers*. C'est dû à leurs propriétés spéciales : 0 est divisible par tout le monde et ne divise personne (à part lui-même), et 1 n'est divisible par personne (à part lui-même) et divise tout le monde (sans jamais produire de résultat « à virgule »).

Propriété (recherche des diviseurs).

Si un nombre x est *composé*, alors il admet au moins un diviseur entre 2 et $\sqrt{x}$.

Preuve. On a déjà vu l'argument : lorsqu'on cherche les diviseurs de x, on essaie *toutes* les factorisations à partir de $1 \times x$, puis $2 \times \ldots$, puis $3 \times \ldots$, etc., et on s'arrête lorsque le premier opérande est plus grand que le deuxième. Imaginons qu'on a une factorisation $x = a \times b$ (donc avec $a \leqslant b$ sinon on s'est déjà arrêté) et que a est strictement plus grand que $\sqrt{x}$. Ça n'est pas possible, puisqu'on aurait alors

$$x = a \times b \overset{\text{car } b \geqslant a}{\geqslant} a \times a \overset{\text{car } a > \sqrt{x}}{>} \sqrt{x} \times \sqrt{x} = x,$$

et que x ne peut pas être strictement plus grand que lui-même. Donc a est, au maximum, égal à $\sqrt{x}$. **C.Q.F.D.**

Par exemple, pour tester si 1 000 003 est premier, il faut essayer de le diviser par tous les nombres entre 2 et $\sqrt{1\,000\,003} \simeq 1\,000$. Évidemment on ne fait pas ça à la main, on utilise un programme pour le faire automatiquement.

```
from numpy import sqrt
def EstPremier(x) :
    d = 2 ; Racine_de_x = sqrt(x)
    while d <= Racine_de_x :
        if x % d == 0 :
            return False
        d += 1
    return True
```

```
>>> EstPremier(1000001)
False
>>> EstPremier(1000003)
True
```

On verra dans la leçon **32** comment établir la liste de tous les nombres premiers (en tous cas ceux inférieurs à 1 000 000 000, parce que sinon il y en a une infinité).