Maths Expertes
Terminale

PGCD, Bézout & Gauss

CORRIGÉ DE L'EXERCICE

Bézout

01

Correction

Cet exercice a pour objectif de rechercher, sur quelques exemples, une combinaison entière $au + bv$ de deux entiers relatifs *a* et *b* égale à leur PGCD.

Cette recherche peut être assurée par « **l'algorithme d'Euclide étendu** ». Il s'agit de l'algorithme d'Euclide usuel, mais en outre, à chaque étape, l'algorithme « étendu » calcule une combinaison entière des deux nombres initiaux *a* et *b* égale au reste courant. En dernier lieu, l'algorithme donne une combinaison entière égale à leur PGCD (en particulier égale à 1 si et seulement si *a* et *b* sont premiers entre eux).

Considérons l'algorithme d'Euclide usuel appliqué à deux entiers $a > b > 0$:

- Division numéro 1 : $a = q_1 b + r_1$ donc $r_1 = a - q_1 b$
- Division numéro 2 : $b = q_2 r_1 + r_2$ donc $r_2 = b - q_2 r_1$
- ….
- Division numéro *k* : $r_{k-2} = q_k r_{k-1} + r_k$ donc $r_k = r_{k-2} - q_k r_{k-1}$
- ….

Si l'on pose $r_j = u_j a + v_j b$ pour chaque indice *j*, nous avons lors de la division numéro *k* :

$r_k = r_{k-2} - q_k r_{k-1} = (u_{k-2} a + v_{k-2} b) - q_k (u_{k-1} a + v_{k-1} b)$ d'où les relations de récurrence :

$$\begin{cases} u_k = u_{k-2} - q_k u_{k-1} \\ v_k = v_{k-2} - q_k v_{k-1} \end{cases}$$

Ce sont ces relations de récurrence qu'il s'agit de mettre en œuvre.

Nous nous proposons ici d'utiliser une mouture Python de cet algorithme, nommée « **euclidetendu** » admettant deux arguments *a* et *b* entiers tels que $a > b > 0$, qui nous servira à traiter l'exercice.

Dans chaque question, nous exécuterons cet algorithme.

Il nous faudra adapter notre réponse aux circonstances de chaque équation, selon les signes des coefficients et les valeurs relatives de leurs valeurs absolues.

```
>>> def euclidetendu(a,b):
        x=a
        y=b
        s=1
        t=0
        u=0
        v=1
        n=1
        while y>0:
                r=x%y
                q=int((x-r)/y)
                if n>1:
                        print("Combinaison de ",a,"et",b,"égale au reste :")
                print(y,"=(",u,")x",a,"+(",v,")x",b)
                print("Division numéro",n,":")
                print(x,"=",q,"x",y,"+",r)
                n=n+1
                c=u
                d=v
                u=s-q*u
                v=t-q*v
                s=c
                t=d
                x=y
                y=r
```

1. Montrons que 26 et 23 sont premiers entre eux et trouvons deux entiers adéquats :

23 est un nombre premier, il est premier avec tout nombre qu'il ne divise pas, ce qui est le cas de 26.

26 et 23 sont premiers entre eux.

Mettons en œuvre l'algorithme « **euclidetendu** » avec 26 et 23.

Nous obtenons en fin de compte la combinaison entière :

$$\mathbf{8 \times 26 - 9 \times 23 = 1}$$

Un couple d'entiers relatifs vérifiant $23x - 26y = 1$ est le couple $(x = -9,\ y = 8)$.

```
>>> euclidetendu(26,23)
23 =( 0 )x 26 +( 1 )x 23
Division numéro 1 :
26 = 1 x 23 + 3
Combinaison de  26 et 23 égale au reste :
3 =( 1 )x 26 +( -1 )x 23
Division numéro 2 :
23 = 7 x 3 + 2
Combinaison de  26 et 23 égale au reste :
2 =( -7 )x 26 +( 8 )x 23
Division numéro 3 :
3 = 1 x 2 + 1
Combinaison de  26 et 23 égale au reste :
1 =( 8 )x 26 +( -9 )x 23
Division numéro 4 :
2 = 2 x 1 + 0
```

2. Montrons que 221 et 331 sont premiers entre eux et trouvons deux entiers adéquats :

Nous observons que la décomposition en produit de facteurs premiers de 221 est $221 = 13 \times 17$.

Or, 331 n'est divisible ni par le facteur premier 13 ni par le facteur premier 17.

331est premier avec 221.

Mettons en œuvre l'algorithme avec 331 et 221.
Nous obtenons en fin de compte la combinaison entière :
$\mathbf{3 \times 221 - 2 \times 331 = 1}$
Un couple d'entiers relatifs vérifiant $221x - 331y = 1$ est le couple : $(x = 3,\ y = 2)$.
	`>>> euclidetendu(331,221)`
	`221 =( 0 )x 331 +( 1 )x 221`
	`Division numéro 1 :`
	`331 = 1 x 221 + 110`
	`Combinaison de 331 et 221 égale au reste :`
	`110 =( 1 )x 331 +( -1 )x 221`
	`Division numéro 2 :`
	`221 = 2 x 110 + 1`
	`Combinaison de 331 et 221 égale au reste :`
	`1 =( -2 )x 331 +( 3 )x 221`
	`Division numéro 3 :`
	`110 = 110 x 1 + 0`

3.a et b : Déterminons le PGCD de 58 et de 24 et simultanément construisons une combinaison entière de ces nombres égale au PGCD :

Mettons en œuvre « **euclidetendu** » avec 58 et 24. Cet algorithme incorpore l'algorithme d'Euclide usuel. Le PGCD est le dernier reste non nul, nous lisons sur l'affichage que ce PGCD est égal à 2.
Nous obtenons la combinaison entière égale au PGCD suivante :
$\mathbf{5 \times 58 - 12 \times 24 = 2}$
Un couple d'entiers relatifs vérifiant $58x - 24y = 2$ est le couple : $(x = 5,\ y = 12)$.
	`>>> euclidetendu(58,24)`
	`24 =( 0 )x 58 +( 1 )x 24`
	`Division numéro 1 :`
	`58 = 2 x 24 + 10`
	`Combinaison de 58 et 24 égale au reste :`
	`10 =( 1 )x 58 +( -2 )x 24`
	`Division numéro 2 :`
	`24 = 2 x 10 + 4`
	`Combinaison de 58 et 24 égale au reste :`
	`4 =( -2 )x 58 +( 5 )x 24`
	`Division numéro 3 :`
	`10 = 2 x 4 + 2`
	`Combinaison de 58 et 24 égale au reste :`
	`2 =( 5 )x 58 +( -12 )x 24`
	`Division numéro 4 :`
	`4 = 2 x 2 + 0`

NB. Dans les exemples 1 et 2, nous aurions pu nous dispenser de vérifier au préalable que les coefficients étaient premiers entre eux. Il suffisait d'exécuter l'algorithme.
Le seul fait que l'algorithme affiche en fin de compte une combinaison entière des nombres initiaux égale à 1 démontre sans aucune ambigüité, grâce au théorème de Bézout, que les deux entiers initiaux en jeu sont premiers entre eux.