



Algorithmes de recherche du PGCD et des coefficients de la relation de BEZOUT.

NIVEAU : Terminale S , spécialité mathématiques.

EXERCICE :

Soient a et b deux entiers non nuls et $d = \text{PGCD}(a ; b)$

La relation de BEZOUT est : il existe deux entiers u et v tels que $au + bv = d$

Voici un rappel de l'algorithme d'Euclide permettant de trouver le PGCD de deux nombres et de la démarche associée pour trouver les valeurs des coefficients de BEZOUT u et v correspondants.

Divisions euclidiennes successives

Etape 0 : Division euclidienne de a par b : $a = b q_0 + r_0$

Etape 1 : Division euclidienne de b par r_0 : $b = r_0 q_1 + r_1$

Etape 2 : Division euclidienne de r_0 par r_1 : $r_0 = r_1 q_2 + r_2$

Etape k : Division euclidienne de r_{k-2} par r_{k-1} : $r_{k-2} = r_{k-1} q_k + r_k$

Etape n : Division euclidienne de r_{n-2} par r_{n-1} : $r_{n-2} = r_{n-1} q_n + d$

Ecriture des restes

En fonction de a et b

$$r_0 = a - b q_0 = a u_0 + b v_0$$

$$r_1 = b - r_0 q_1 = a u_1 + b v_1$$

$$r_2 = r_0 - r_1 q_2 = a u_2 + b v_2$$

$$r_k = r_{k-2} - r_{k-1} q_k = a u_k + b v_k$$

$$d = r_{n-2} - r_{n-1} q_n = a u_n + b v_n$$

Cet algorithme s'arrête lorsque le reste de la division euclidienne suivante est nul, ici lorsque le reste de la division euclidienne de r_{n-1} par d est 0 .

- 1) Etablir une relation de récurrence qui permet de déterminer u_k et v_k à l'étape k en fonction des deux étapes précédentes (k-1) et (k-2) .
- 2) En déduire un algorithme permettant de déterminer d, u et v pour deux entiers donnés.
- 3) Ecrire le programme correspondant sur la calculatrice.

1) Dans l'étape k nous allons exprimer r_k : $r_k = r_{k-2} - r_{k-1} q_k$

On sait que $r_{k-2} = a u_{k-2} + b v_{k-2}$ et que $r_{k-1} = a u_{k-1} + b v_{k-1}$

D'où $r_k = (a u_{k-2} + b v_{k-2}) - (a u_{k-1} + b v_{k-1}) q_k$

D'où $r_k = a (u_{k-2} - u_{k-1} q_k) + b (v_{k-2} - v_{k-1} q_k)$

On en déduit les relations de récurrence suivantes :

$$u_k = u_{k-2} - u_{k-1} q_k \quad \text{et} \quad v_k = v_{k-2} - v_{k-1} q_k$$

où q_k est le quotient de r_{k-2} par r_{k-1} .

On a donc pour le triplet (u,v,r) la même relation de récurrence (u,v,r) au rang k = (u,v,r) au rang k-2 - (u,v,r) au rang k-1 x q_k



initialisations :

au premier passage de l'algorithme on a $u_0 = 1$ et $v_0 = -q$ et $r_0 = a - bq$ donc

on a $(u,v,r) = (1,0,a) - q(0,1,b)$ il faut initialiser le rang $k-2$ avec $(1,0,a)$ et le rang $k-1$ avec $(0,1,b)$

2) Nous allons utiliser cette relation pour écrire le programme.

A chaque étape du programme nous allons avoir besoin de sauvegarder les triplets (u,v,r) des deux étapes précédentes.

Variables :

a, b : les deux entiers

List1 : liste de 3 valeurs u,v et r au rang $k-2$

List2 : Liste des 3 valeurs u,v,r au rang $k-1$

List3 : liste des 3 valeurs u,v,r au rang k

Entrées :

Entrer les deux entiers concernés a et b .

Initialisations :

List1 = { 1,0,a }

List2 = { 0,1,b }

Traitement :

tant que le reste de la division euclidienne de r_{k-2} par r_{k-1} est non nul

List3 = List 1 - (quotient de r_{k-2} par r_{k-1})x List2

On avance d'une étape : List 1 = List2 et List2 = List3

Fin tant que.

Affichage du résultat :

le dernier triplet (u,v,r) dans la list 1 correspond aux coefficients de la relation de Bezout et au PGCD d . on a $au+bv=d$

Fin.



Initialisation de list1 et list2

SHIFT X 1 , 0 , ALPHA X,θ,T SHIFT ÷ →
 SHIFT 1 1 EXE
 SHIFT X 0 , 1 , ALPHA log SHIFT ÷ →
 SHIFT 1 2 EXE

boucle tant que le reste est non nul

SHIFT VARS F1 F6 F6 F1 (While)

SHIFT 1 2 SHIFT + 3 SHIFT -
 SHIFT VARS F6 F3 F2 0 EXE (List2[3] ≠ 0)

SHIFT 1 1 -
 SHIFT 1 2 X OPTN F6 F4 F2
 (SHIFT 1 1 SHIFT + 3 SHIFT -
 ÷ SHIFT 1 2 SHIFT + 3 SHIFT -
) → SHIFT 1 3 EXE

(List1-List2×ent(list1[3] /list2[3]))

SHIFT 1 2 → SHIFT 1 1 EXE (List2 →list1)

SHIFT 1 3 → SHIFT 1 2 EXE (List3→list2)

SHIFT VARS F1 F6 F6 F2 (Whileend)

EXE

```

=====BEZOUT =====
"      A      ET      B"↵
"A ="?→A↵
"B ="?→B↵
<1,0,A>→List 1↵
<0,1,B>→List 2↵
|
|TOP|ETM|SRC|MENU|A↵B|CHAR|
    
```

```

=====BEZOUT =====
While List 2[3]≠0↵
List 1-List 2×Int (Li
st 1[3]÷List 2[3])→Li
st 3↵
List 2+List 1↵
List 3→List 2↵
|TOP|ETM|SRC|MENU|A↵B|CHAR|
    
```


1^{re}/Terminale

Graph 35+ USB



SHIFT VARS F6 F4 F1 1 , 5 , SHIFT ALPHA
x10⁻³ 1 SHIFT . ALPHA x10³ EXE
F1 3 , 5 , SHIFT 1 1 SHIFT + 1
SHIFT - (Affichage de U=valeur de U)

EXE F1 1 , 6 , SHIFT ALPHA x10³ 2 SHIFT
ALPHA x10³ EXE
F1 3 , 6 , SHIFT 1 1 SHIFT + 2
SHIFT - EXE (Affichage de V=valeur de V)

sortie du programme

EXIT EXIT EXIT

```
=====BEZOUT =====  
Locate 1,3,"PGCD="␣  
Locate 6,3,List 1[3]␣  
Locate 1,4,"BEZOUT:AU  
+BU=PGCD"␣  
Locate 1,5,"U="␣  
Locate GthY Send Recv
```

```
=====BEZOUT =====  
+BU=PGCD"␣  
Locate 1,5,"U="␣  
Locate 3,5,List 1[1]␣  
Locate 1,6,"U="␣  
Locate 3,6,List 1[2]␣  
Locate GthY Send Recv
```

```
Liste programmes  
BEZOUT : 332  
SOM2DES : 200  
EXE EDIT NEW DEL DELA
```

1^{re}/Terminale

Graph 35+ USB



Exécution de Bezout :

F1 **1** **5** **3** **EXE** **9** **3** **EXE**

On obtient bien les résultats voulus :

Essai en inversant A et B :

EXIT **EXE** **9** **3** **EXE** **1** **5** **3** **EXE**

On remarque qu'on a bien $-23 \times 93 + 14 \times 153 = 3$

ENTREZ LES ENTIERS

A ET B

A = ?

153

B = ?

93

A=153

B=93

PGCD=3

BEZOUT: AU+BV=PGCD

U=14

V=-23

A=93

B=153

PGCD=3

BEZOUT: AU+BV=PGCD

U=-23

V=14