Un programme pour Bézout

1. La théorie.

a) L'algorithme d'Euclide.

On considère $a, b \in \mathbf{N}$ avec $b \neq 0$. On pose $a = r_0$, $b = r_1$. On effectue la division euclidienne de a par b : $a = bq+r$ avec $0 \leq r < b$. On pose $q = q_1$, $r = r_2$. On a donc $r_0 = r_1q_1 + r_2$ avec $0 \leq r_2 < r_1$ et $d = pgcd\,(a, b) = pgcd\,(r_0, r_1) = pgcd\,(r_1, r_2)$.
On construit ainsi par récurrence des entiers $r_0, r_1, \cdots, r_k, r_{k+1}$ et $q_1, \cdots, q_k$ avec $r_{k-1} = q_kr_k + r_{k+1}$, $0 \leq r_{k+1} < r_k$ et $d = pgcd\,(r_k, r_{k+1})$. Si on a $r_{k+1} = 0$ on a $d = r_k$ et on s'arrête, sinon on continue l'algorithme en divisant r_k par r_{k+1}.
Comme on a $0 \leq r_{k+1} < r_k < \cdots < r_1$ on voit que l'on obtient nécessairement un reste nul au bout d'au plus r_1 opérations. Si on désigne par r_n le dernier reste non nul on a donc $r_{n-1} = q_nr_n$ d'où, $d = pgcd\,(r_{n-1}, r_n) = r_n$.

b) Le théorème de Bézout.

Pour montrer Bézout, on utilise l'algorithme d'Euclide. On va montrer, par récurrence sur k que, pour tout k avec $0 \leq k \leq n$, il existe des entiers $u_k, v_k \in \mathbf{Z}$ vérifiant $r_k = u_ka + v_kb$.
L'assertion est vraie pour $k = 0$ puisqu'on a $r_0 = a = 1 \times a + 0 \times b$ et pour $k = 1$ puisqu'on a $r_1 = b = 0 \times a + 1 \times b$. Supposons l'assertion prouvée pour tout entier $\leq k$, avec k fixé vérifiant $1 \leq k < n$, et montrons la pour $k + 1$. On a $r_{k+1} = r_{k-1} - q_kr_k = u_{k-1}a + v_{k-1}b - q_k(u_ka + v_kb)$ d'où la relation cherchée en posant $u_{k+1} = u_{k-1} - q_ku_k$ et $v_{k+1} = v_{k-1} - q_kv_k$.
Si on applique l'assertion au cas $k = n$, comme on a $r_n = d = pgcd\,(a, b)$, on obtient bien la relation de Bézout cherchée.

2. Le programme.

a) Le programme sur TI-92.

```
bezout()
Prgm
Local a,b,u,v,x,y,c,d,q,r
Prompt a,b
1 → u   0 → v   0 → x   1 → y
While b > 0
reste (a,b) → r   (a-r)/b → q
u → c   v → d   x → u   y → v
c-q*x → x   d-q*y → y   b → a   r → b
EndWhile
Disp "pgcd", a   Disp "u" , u   Disp "v", v
EndPrgm
```

b) Discussion et exemple.

Le programme suit exactement l'algorithme théorique. On part de a, b donnés (c'est le sens du `Prompt a,b`) disons 35 et 95 (je fais exprès de prendre $a < b$).
Première difficulté de programmation, comme on a une récurrence double sur u_k, v_k il faut deux variables pour rendre compte de chacune des deux suites u_k, v_k. Moralement, u, v sont u_{k-1} et v_{k-1}, tandis que x, y sont u_k, v_k. Ainsi, pour $k = 1$ on initialise $u = u_0 = 1$, $v = v_0 = 0$ et $x = u_1 = 0$, $y = v_1 = 1$.
Ensuite on fait une boucle `While` tant que $b > 0$. Moralement, a, b sont r_{k-1} et r_k, de sorte que, vu les définitions, r est r_{k+1} et q est q_k. À l'étape suivante, on remplace a par b et b par r ce qui revient à faire avancer k d'un pas.
Il s'agit de faire avancer aussi k pour u_k et v_k, par les deux formules : $u_{k+1} = u_{k-1} - q_k u_k$ et $v_{k+1} = v_{k-1} - q_k v_k$. C'est là qu'arrive la seconde difficulté de programmation. Ce qu'on veut c'est remplacer u_{k-1} par u_k donc u par x et u_k par $u_{k-1} - q_k u_k$, donc x par $u - qx$. Mais on ne peut pas dire d'emblée : $x \to u$ et $u - qx \to x$ car en faisant cela, le u change à la première manœuvre et la deuxième est donc erronée. On ne peut pas non plus faire $u - qx \to x$ et $x \to u$ car ici c'est le x qui change. C'est pourquoi on a besoin de deux variables c, d supplémentaires qui permettent de garder u, v au frais pendant le calcul. Elles ne servent que temporairement et n'ont pas d'autre importance.

Étape 0.

$a = 35$, $b = 95$, $u = 1, v = 0$; $x = 0, y = 1$.

Étape 1.

On divise a par b : $r = 35$, $q = 0$. On stocke u, v en c, d, puis on change : u devient x, etc. i.e. $u = 0, v = 1$; $x = 1 - 0 * 0 = 1, y = 0 - 0 * 1 = 0$, et on remplace a, b par b, r : $a = 95, b = 35$.
Commentaire : on voit que cette opération remet les données dans le bon sens : $a > b$.

Étape 2.

On divise a par b : $r = 25, q = 2$. On obtient $u = 1, v = 0$ et $x = 0 - 2 * 1 = -2, y = 1 - 2 * 0 = 1$ et on remplace a, b par b, r : $a = 35, b = 25$. On constate qu'on a bien les relations :
$a = u \times 35 + v \times 95$, i.e. $35 = 1 \times 35 + 0 \times 95$ et $b = x \times 35 + y \times 95$, i.e. $25 = -2 \times 35 + 1 \times 95$.
On peut noter ce qui se serait passé sans les variables auxiliaires c, d. On fait $x \to u$ donc u devient 1 et $u - qx \to x$ ce qui donne $x = 1 - 2 \times 1 = -1$ et ça ne marche plus.

Étape 3.

On obtient $r = 10, q = 1$; $u = -2, v = 1$; $x = 3, y = -1$; $a = 25, b = 10$.

Étape 4.

On obtient $r = 5, q = 1$; $u = 3, v = -1$; $x = -8, y = 3$; $a = 10, b = 5$.
La seconde relation de Bézout est $r = x \times 35 + y \times 95$, soit $5 = -8 \times 35 + 3 \times 95$.

Étape 5.

On obtient $r = 0, q = 2$; $u = -8, v = 3$; x, y peu importe, $a = 5, b = 0$. Comme on a $b = 0$ la boucle `While` s'arrête et on fait afficher le pgcd a et les coefficients de Bézout u, v. La relation est celle écrite ci-dessus.