```
/*
GTT/Inverses

Find the greatest common divisor g=gcd(a,n) of two integers (a,n).
Find coeffients (x,y) s.th. ax +  ny = g.
If a and n have no common divisor (g=1) then x=inverse(a) (mod n),
and y=inverse(n) (mod a).
Algorithm according to the Extended Euclidean Algorithm,
quoted from Knuth: "The Art of Computer Programming", Vol. 2.
C Program written by R.Grimm at Fri, 10 June 94.
Explanation sent by E-Mail from S.Guergens at Mon, 16 May 94.
Modification of explanation and program by R.G. and D.Pähler Nov 2010.

Behauptung: Fuer a,n gibt es x,y mit ax + ny = ggT(a,n)

Der Algorithmus bestimmt fuer besagte a,n einen Vektor (x,y,g),
so dass ax + ny = g = ggT(a,n). Zur Berechnung dieses Vektors
braucht man noch Hilfsvektoren (v1,v2,v3) und (t1,t2,t3).

1. Schritt: Setze (x,y,g)    := (1,0,a),
                 (v1,v2,v3) := (0,1,n)

2. Schritt: v3 =  0 ? Ja ==> fertig

3. Schritt: Setze h := groesste ganze Zahl <= g/v3
                (t1,t2,t3) := (x,y,g) - (v1,v2,v3)h
                (x,y,g)    := (v1,v2,v3)
                (v1,v2,v3) := (t1,t2,t3)
            Weiter bei Schritt 2.

Ein Beispiel:

a=28, n=5, d.h. zu loesen ist 28x +5y = g = ggT(28,5)

1. Schritt: x = 1,  v1 = 0
            y = 0,  v2 = 1
            g = 28, v3 = 5

2. Schritt: v3 nicht = 0

3. Schritt: h = 5
            t1 = 1,  x = 0, v1 = 1
            t2 = -5, y = 1, v2 = -5
            t3 = 3,  g = 5, v3 = 3

2. Schritt: v3 nicht = 0

3. Schritt: h = 1
            t1 = -1, x = 1,  v1 = =- 1
            t2 = 6,  y = -5, v2 = 6
            t3 = 2,  g = 3,  v3 = 2

2. Schritt: v3 nicht = 0

3. Schritt: h = 1
            t1 = 2,   x = -1, v1 = 2
            t2 = -11, y = 6,  v2 = -11
            t3 = 1,   g = 2,  v3 = 1

2. Schritt: v3 nicht = 0

3. Schritt: h = 2
            t1 = -5, x = 2,   v1 = -5
            t2 = 28, y = -11, v2 = 28
            t3 = 0,  g = 1,   v3 = 0

2. Schritt: v3 = 0 --> fertig
```

```
Nun haben wir 28x + 5y = g = 1 = ggT(28,5), d.h.
              28*2 + 5*-11 = 1

Rechnet man jetzt im Restklassenring Z_28, dann ergibt sich

              28*2 + -11*5 = 1 mod 28, und, da 28*2 mod 28 = 0
                     -11*5 = 1 mod 28

m.a.W., die zu 5 inverse Zahl in Z_28 ist -11, oder, wenn man die
kleinste positive nehmen will, 17.

Genauso ist natuerlich in Z_5 die zu 28 = 3 mod 5 inverse Zahl = 2.

************** The C Source Code Program: ************************
*/

#include <stdio.h>

main() {

long a,n,x,y,g;
long v1,v2,v3;
long h,t1,t2,t3;

printf("Input a and n (a<n): \na=");
scanf ("%ld", &a) ;
printf ("n=");
scanf("%ld",&n) ;
if( a>n ) {
        /* printf(?a>n, therefore the role of a and n are interchanged\n"); */
        h=a; a=n; n=h;
        }
x=1; y=0; g=a;
v1=0; v2=1; v3=n;

while( v3 ) {
        h=g/v3;
        t1=x-v1*h; t2=y-v2*h; t3=g-v3*h;
        x=v1; y=v2; g=v3;
        v1=t1; v2=t2; v3=t3;
        }

/* v3==0 (finish) */
if( x<0 ) {
        x=n+x;
        y=- (x*a-1) /n;
        }

printf("\t gcd(%ld,%ld) = %ld\n", a,n,g);
printf("\t %ld X %ld - %ld X %ld = %ld\n", a,x,n,-y,g);
printf("\t %ld X %ld = %ld (mod %ld)\n", a,x,g,n);
if( g==1 ){
        printf("a=%ld and n=%ld are rel. prime:", a,n);
        printf("\t %ld = inverse(%ld) mod %ld\n\n", x,a,n);
        }
if( g==a ) {
        printf("a=%ld is Null-Divisor of n=%ld:\n", a,n);
        x=n/a; y=1;
        printf("\t %ld X %ld - %ld X %ld = %ld\n", a,x,n,y,g);
        printf("\t %ld X %ld = 0 (mod %ld)\n", a,x,n);
        }

}

/* ggT nach erweitertem Euklidischem Algorithmus,
   Programm von Rüdiger Grimm, 1994 */
```