```
/*
    Crittografia polinomiale-grafica-matriciale (PGM):
    Autore: Andrea Olivari
*/

Chiave (privata):
 - una coppia (stringa, polinomio) = (S,p(x)) = (A,p(x)), o equivalentemente (matrice, polinomio)

    Caratteristiche:
        - La matrice A, associata alla stringa S, deve essere NON ortogonale e l'inversa del prodotto con se stessa non
        deve contenere valori -0.0005 < v < 0.0005.
        Il polinomio p(x) ha lunghezza e grado arbitrari (sono particolarmente consigliati polinomi non tendenti a ±
        infinito)
        - La matrice A può contenere un numero indefinito elementi ripetuti

Descrizione dell'algoritmo mediante un esempio:

Chiave:

    p(x) = sinx^2-tanx/3

    A:  // Utilizziamo la tabella ASCII per ottenere i rispettivi valori per ogni simbolo della stringa chiave.
        // equivalente a: "4/4 AEOGC IMFNHQBDPL" // la stringa comincia sempre con il numero di righe/colonne, seguito
        da uno spazio.

    {65,69,79,71},
    {67,32,73,77},
    {70,78,72,81},
    {66,68,80,76}

Messaggio da spedire:

    "ciao a"

Procedimento:

    AAt:

    {20268,17797,21371,20698},
    {17797,16771,18679,18290},
    {21371,18679,22729,21840},
    {20698,18290,21840,21156}

    /* Eliminiamo gli eventuali duplicati nella matrice, incrementando, di una unità, i numeri coinvolti tante volte
    quante necessario (ovvero fin quando non otteniamo un numero non presente nella matrice) */
```

```
    AAt (senza duplicati):

    {20268,17798,21372,20699},
    {17797,16771,18680,18291},
    {21371,18679,22729,21841},
    {20698,18290,21840,21156}

    det(AAt) = 133614
    norma infinito di A = 84620

    Inv(AAt): // i numeri ottenuti andranno a ridursi a, minimo, tre cifre decimali (si smette di inserire le cifre
    decimali solo quando queste sono ALMENO 3, di cui ALMENO una significativa)

    {0.199,0.025,0.024,-0.241},
    {0.026,0.005,0.005,-0.034},
    {0.026,0.005,0.01,-0.04},
    {-0.243,-0.034,-0.038,0.306}

    Inv(AAt) (senza duplicati):

    {0.199,0.025,0.024,-0.241},
    {1.026,1.005,2.005,0.966},
    {0.026,0.005,0.01,-0.04},
    {-0.243,-0.034,-0.038,0.306}

E' NECESSARIO per il corretto funzionamento dell'algoritmo che il grafico inviato contenga ALMENO n+3 punti scrivibili,
dove n indica la lunghezza del messaggio che si intende inviare (in questo caso 6).
Nel caso il numero di indici tra l'indice di partenza del messaggio e quello finale non fosse sufficiente a coprire
l'intero messaggio, si ripartirà ciclicamente, ovvero, una volta terminate le ascisse, si continueranno a inserire i
valori partendo dalle prime di esse (trascurando ovviamente quelle già impegnate, come, ad esempio, quella che esprime
la lunghezza del messaggio).
Scegliamo di utilizzare 55 punti (ovviamente solo 9 saranno significativi, gli altri serviranno solo per incrementare
il dubbio di un eventuale intercettatore malintenzionato).

    idet = det(AAt) in 55 = 133614 in 55 = 19
    istart = norm(AAt) in 55 = 84620 in 55 = 30 // Nel caso sfortunato in cui idet e istart risultassero uguali (i),
    manterremmo nella posizione trovata 'i' idet e modificheremmo istart in 'i+1'
    ijump = norm(AAt)*det(AAt) in 55 = 11306416680 in 55 = 20 // La stessa identica cosa vale per ijump, il quale andrà
    però a confrontarsi sia con idet che con istart, incrementandosi ogni volta di uno, se necessario.

Dunque abbiamo che in posizione x=19 (idet) inseriamo la lunghezza del messaggio, mentre in posizione x=30 (istart)
inseriamo l'indice di partenza del messaggio e in posizione x=20 (ijump) inseriamo il valore 'jump' che esprime quanti
salti effettuare tra un valore significativo e il suo successivo.
(Scelgo come indice di partenza x=31 e chiaramente quando scegliamo l'indice di partenza dobbiamo ricordare che NON
POSSIAMO indicare gli indici x=19, x=30 e x=20).
```

Scelgo come parametro 'jump' il valore 5, e questo significa che solo le x multiple di 5 potranno contenere valori significativi.
L'ordine di elaborazione è: idet, istart, ijump. (se non si desidera saltare alcun indice incontrato, è sufficiente dichiarare come parametro 'jump' il valore 1).
PRIMA DI SCEGLIERE I PARAMETRI 'indice di partenza' e 'jump' è necessario assicurarsi che con le opzioni scelte sia possibile rappresentare tutti i valori significativi. In questo caso abbiamo 55 posizioni totali e dobbiamo inserire 9 valori, escludendo (PER I VALORI DEL MESSAGGIO VERO E PROPRIO) tutti i punti tali che x non è multiplo di jump=5.
Dunque, nel caso peggiore (ovvero quello in cui det, norm e jump in mod 55 si trovino su una x multipla di 5), avremo a disposizione 55/5-3 = 11-3 = 8 punti per i 6 valori del messaggio "ciao a"; questo significa che siamo perfettamente in grado di inviare il messaggio desiderato con le attuali impostazioni.
E' opportuno notare che il parametro jump avrà la sua utilità solo per parte della fase di crypting, infatti in seguito i valori verranno rimescolati in ordine sparso (secondo la logica spiegata più avanti) e disposti su varie x tra 1 e 55, indipendentemente che quest'ultime siano multiple o no del valore jump=5.
Partiamo dall'indice 31 a inserire il messaggio e ogni volta che incontriamo una posizione non scrivibile avanziamo a quella successiva (può anche capitare che sia l'indice di partenza stesso un indice da saltare).
Procediamo come segue (indicando con vi l'i-esimo valore da inserire, con ival i valori contenuti nella matrice inversa, con tval i valori contenuti nella matrice AAt e con aval i valori della matrice A):

yi = vi*ival((i*tval(i mod k)) mod k)*tval(i mod k)/aval((aval(i mod j)*i) mod j) // dove k è uguale al numero di elementi della matrice AAt e j il numero di elementi della matrice A (in questo specifico caso k=j=16)

---

y(idet) = y19 = 6*-0.241*21372/aval(13) = 6*-0.241*21372/66 = -468.241
y(istart) = y30 = 31*-0.04*18290/aval(8) = 31*-0.04*18290/77 = -294.540
y(ijump) = y20 = 5*-0.04*20699/aval(12) = 5*-0.04*20699/81 = -51.109

---

y35 = 99*-0.241*21372/aval(13) = 99*-0.241*21372/66 = -7725.978
y40 = 105*0.966*18291/aval(8) = 105*0.966*18291/77 = 24094.235
y45 = 97*0.025*20698/aval(10) = 97*0.025*20698/78 = 643.496
y50 = 111*-0.04*17798/aval(10) = 111*-0.04*17798/78 = -1013.117
y55 = 32*0.966*18680/aval(15) = 32*0.966*18680/80 = 7217.952
y5 = 97*0.026*17797/aval(15) = 97*0.026*17797/80 = 561.050

---

x	5	19	20	30	35	40	45	50	55
y	561.050	-468.241	-51.109	-294.540	-7725.978	24094.235	643.496	-1013.117	7217.952

Eseguo i vari spostamenti nel seguente modo (indicando np=55 e con "y(i) --> h" il fatto che il valore all'i-esima posizione verrà spostato in h-esima posizione). Se la posizione risultante non è più disponibile, si avanza di un posto fin quando non se ne trova uno libero (in questo caso NON importa più se la posizione trovata è multipla di jump, vanno bene tutte a parte quelle già occupate da valori significativi).

```
    yi --> (i*tval(i mod k)) mod np

    y(idet)   = -468.241  --> 3
    y(istart) = -294.54   --> 20
    y(ijump)  = -51.109   --> 50
    y(35)     = -7725.978 --> 21
    y(40)     = 24094.235 --> 30
    y(45)     = 643.496   --> 40
    y(50)     = -1013.117 --> 55
    y(55)     = 7217.952  --> 1
    y(5)      = 561.05    --> 51

Schema aggiornato (mostrando solo le x significative):

    ---------------------------------------------------------------------------------------------
    x |    1    |    3    |    20   |    21    |    30    |   40   |   50   |   51   |    55    |
    ---------------------------------------------------------------------------------------------
    y | 7217.952 | -468.241 | -294.54 | -7725.978 | 24094.235 | 643.496 | -51.109 | 561.05 | -1013.117 |
    ---------------------------------------------------------------------------------------------

Adesso questi 9 valori del grafico andremo a rappresentarli come:

    yi = l(i)/p(i) // dove p(i) indica il risultato del polinomio in funzione di i, ridotto al minimo numero di cifre
    decimali (ovvero con la stessa logica di prima: minimo tre cifre decimali, di cui almeno una significativa)

Il numero di cifre decimali da utilizzare nei valori risultanti non è definito a priori, ma DEVE essere tale da
permettere il ritrovamento del numeratore originale, mediante la moltiplicazione tra esso e il denominatore originale.
Il calcolo, eseguito con un calcolatore, si spinge inizialmente fino a 100 cifre decimali (qui di seguito sostituerò le
cifre con dei puntini), che poi verranno ridotte alle minime necessarie mediante l'algoritmo mostrato poco più avanti.


y[1]  = 7217.952/0.012   = 601496                    -> 601496
y[3]  = -468.241/0.139   = -3368.64028776978417...   -> -3368.641
y[20] = -294.540/0.521   = -565.33589251439539...    -> -565.336
y[21] = -7725.978/0.860  = -8983.695348837209...     -> -8983.695
y[30] = 24094.235/-0.088 = -273798.125               -> -273798.13
y[40] = 643.496/0.062    = 10378.96774193548387...   -> 10378.96
y[50] = -51.109/-0.739   = 69.1596752368064952...    -> 69.159
y[51] = 561.050/0.576    = 974.0451388888888888...   -> 974.046
y[55] = -1013.117/0.098  = -10337.92857142857...     -> -10337.928
```

Determino il massimo e il minimo valore significativo:

    min = -273798.12 e max = 601496

Modifichiamo i due valori sottraendo e sommando, rispettivamente, il valore di norm, ottenendo:

    min = -358418.12 e max = 686116

Infine aggiungiamo 55-9 = 46 numeri randomici compresi tra min e max negli indici non utilizzati.

Mediante un semplice codice Java ho generato i 46 numeri randomici, e il vettore finale da trasmettere è:

[601496.0, 678970.343, -3368.64, 555642.373, 645469.464, 455219.799, -232347.797, -167980.952, 12570.797, 162153.469, 619691.55, 500089.875, -349641.234, 520705.019, 581677.457, 487027.382, -48096.711, -243615.481, 90258.14, -565.335, -8983.695, 156634.846, -73487.498, 559146.204, 88180.874, -268880.674, -22948.585, -180940.468, 476475.303, -273798.12, 465448.591, -278242.994, -74531.63, 217394.18, 289548.244, 386855.376, 50530.518, 168889.214, 78812.839, 10378.96, -287011.879, -129593.876, -215135.339, -219648.068, 605839.897, -229244.612, 237581.092, 100501.395, -181748.011, 69.159, 974.045, -201710.717, -75216.326, 603502.959, -10337.928]

Possiamo decidere di generare il relativo grafico (.fig) usando Matlab ed inviarlo al destinatario, oppure possiamo semplicemente trasmettere il vettore contenente i 55 valori.