Pràctiques Integrades
1er de Matemàtiques

Pràctica 18

curs 2002–03

18 Codificació de missatges amb RSA

El primer que s'ha de fer per a poder treballar amb l'algoritme d'encriptaciò RSA és obtenir un parell de nombres primers prou grans i que es puguin mantenir en secret. Utilitzant les comandes `rand` i `nextprime` podem generar de forma aleatòria dos primers amb un nombre de xifres considerable. La comanda `randomize()` del principi fa que el generador de nombres aleatoris utilitzi com a llavor un nombre que depèn del rellotge del sistema i, d'aquesta forma, els nombres que surten canvien en cada sessió de Maple que arranquem (si no es fa així, la seqüència de nombres aleatoris sempre serà la mateixa).

```
> randomize();
> nombre1:= rand(10^80)();
> nombre2:= rand(10^80)();
> Primer1:= nextprime(nombre1);
> Primer2:= nextprime(nombre2);
```

(els nombres de 80 xifres caben prou bé en una sola línia i ja són prou grans com per fer força llargs els càlculs de descomposició del seu producte).

Seguidament s'han d'obtenir els dos nombres màgics que fan funcionar el sistema. El producte dels dos primers i el producte de cada un dels primers menys una unitat.

```
> n:= Primer1*Primer2;
> phin:= (Primer1-1)*(Primer2-1);
```

Podeu observar que, tot i que és fàcil determinar que `n` i `phin` no són primers, no és tan fàcil obtenir la seva descomposició.

```
> isprime(n);
> isprime(phin);
> ifactor(n);
> ifactor(phin);
```

Finalment s'ha de decidir quin serà el nombre `e` que s'ha de fer públic (juntament amb `n`) perquè ens puguin enviar missatges encriptats. Aquest nombre ha de ser coprimer amb `phin` i com que s'ha de fer públic es pot generar de qualsevol forma simple. Una opció consisteix en triar un nombre primer (així hi ha més opcions per a que no hi hagi factors comuns amb `phin`) d'algun tipus especial fàcil de generar. Per exemple podem fer

```
> e:= 2^16+1;
> isprime(e);
> igcd(e,phin);
```

Però també es pot deixar l'el·lecció a l'atzar encara que no sigui necessari en absolut.

Exercici 18.1

Trieu un nombre primer de fins a 6 xifres que sigui coprimer amb el nombre `phin` que teniu en aquests moments.

La clau privada per a desencriptar els missatges encriptats utilitzant la clau que acabeu de generar és el nombre `d` tal que `e*d` és 1 mòdul `phin`. Recordeu que la comanda `mod` pot fer la feina.

Observeu que si es coneix `phin` la clau `d` es pot obtenir fàcilment. Recordeu que, encara que conegueu `n`, el valor de `phin` no és immediat de calcular. En el paquet **numtheory** hi ha la comanda `phi` que pot determinar `phin` en funció de `n` i, per tant, la clau per a desencriptar es pot calcular a partir de les dades que són públiques (o no?).

Exemple 18.1

```
> with(numtheory);
> nombre:= 17*13;
> phi(nombre);
> phi(n);
```

Exercici 18.2

Determineu la clau `d` per a desencriptar que correspon a la clau pública `(e,n)` que acabeu de generar.

Exercici 18.3

Fabriqueu una funció `encript` que encripti un nombre i una `decript` que desencripti, a partir d'unes claus públiques i privades arbitràries (és a dir, entre els arguments hi ha d'haver els nombres `n`, `e` i `d`). (Utilizeu `Power(a,e)` per a calcular potències amb combinació amb `mod`).

Podeu comprovar si les vostres funcions fan la feina calculant en alguns exemples

```
> encript(a,e,n);
> decript(%,d,n);
```

Nota: Recordeu que un nombre a s'encripta amb la clau pública (e, n) considerant a^e mòdul n i que si tenim un nombre b encriptat podem recuperar l'original fent b^d mòdul n.

18.1 Codificació d'un missatge

Si es vol utilitzar el sistema RSA per a enviar missatge autèntics, és a dir enviar una frase escrita com *"Hola, bon dia!"* a algú altre, cal codificar aquest missatge i convertir-lo en un nombre que pugui ser encriptat pel sistema. Aquesta part del procés és una altra de les parts que han de ser conegudes per tots els usuaris del mateix sistema. A continuació es proposa un sistema de codificació i descodificació de missatges bastant simple basat en interpretar els codis ASCII dels caràcters que componen el missatge com els dígits d'un nombre escrit en base 256.

Primer de tot guardem en una variable la cadena de text que es vol codificar (el missatge)

```
> miss:= "Hola, bon dia!";
```

La comanda `convert` té una opció per a convertir un text en la llista dels codis ASCII de cada un dels caràcters que el composen. Basta fer

```
> llascii:= convert(miss,bytes);
```

La mateixa comanda també fa el procés invers. És a dir, si tenim una llista de nombres entre 1 i 256 ens torna els caràcters que tenen aquests nombres com a codi ASCII

```
> convert(llascii,bytes);
```

Interpretar aquesta llista de codis ASCII com a dígits d'un nombre escrit en base 256 vol dir que el nombre que representa el missatge serà la suma

```
> misscod:=sum(llascii[i+1]*256^i,i=0..nops(llascii)-1);
```

I a aquest nombre és al que li podem aplicar els algoritmes d'encriptació.

El procés per a recuperar un missatge codificat també es pot fer sense gaire dificultat. Si tenim un nombre com `misscod`, el primer que podem fer és buscar la llista dels seus dígits si el volguéssim representar en base 256. La comanda `convert` també té les opcions necessàries per a obtenir aquest resultat amb

```
> llascii2:= convert(misscod, base,256);
```

Noteu ara que si feu escriure els caràcters que tenen els codis ASCII de la llista `llascii2` obteniu un altre cop el missatge original.

```
> printf(convert(llascii2,bytes));
```

Ara ja es pot fer un parell de procediments que realitzin tota la feina d'un sol cop.

Exercici 18.4

Feu un procediment `codif`, que tingui com argument una cadena de caràcters com `miss` i que doni com a resultat el nombre resultant de la forma de codificar que acabeu de veure, i un procediment `decod`, que realitzi la feina inversa, és a dir prengui un nombre i retorni la cadena de caràcters que representa d'acord amb la codificació anterior (utilitzeu `printf` per a representar el resultat, dóna els millors resultats de visualització).

Exercici 18.5

Els nombres que es poden encriptar be amb el sistema RSA són tots els que són menors que `n`. Quants caràcters pot arribar a tenir un missatge per a poder codificar-lo amb els `n` que hem generat al principi?