> Math 418 : MAPLE Solution of Assignment #3 -- *Your NAME*

Problem 3: The RSA encription method

```
> restart; with(numtheory) :
```

(a) Program to encode a given message m using pulic key (n,e) :

```
> encode := proc(m, n, e) local M;
   M := Power(m, e) mod n;
   return(M); end;
      encode := proc(m, n, e) local M; M := Power(m, e) mod n; return M end proc     (1)
```

Notes: 1) Recall that the command "Power(m,e) mod n" is MAPLE's implementation of the power-mod algorithm which we learned in class. If had you used instead the naive command "modp(m^e, n)", then the computations take much longer (and you will get an overflow error for large values of m and e.

2) A shorter way to write the above program is as follows:

```
> encode := (m, n, e) → Power(m, e) mod n;
                         encode := (m, n, e) → Power(m, e) mod n          (2)
```

(b) Program to encode a message list mls with public key (n,e):

```
> encodelist := proc(mls, n, e) local i;
    return([seq(encode(mls[i], n, e), i = 1 ..nops(mls))]); end;
encodelist := proc(mls, n, e)                                              (3)
    local i; return [seq(encode(mls[i], n, e), i = 1 ..nops(mls))]
end proc
```

Alternate version (for advanced MAPLE users):

```
> encodelist := (mls, n, e) → map(encode, mls, n, e);
                    encodelist := (mls, n, e) → map(encode, mls, n, e)    (4)
```

(c) Test the above program on the message "this is top secret".
Recall from class that this message was translated into the following number list:

```
> ml := [20080919, 91900, 20151600, 19050318, 5200000];
              ml := [20080919, 91900, 20151600, 19050318, 5200000]        (5)
```

Note (for MAPLE experts): Instead of typing in these numbers, we could also use a conversion program to convert a given text message into a list of numbers. (See the comments at the end.)

The public key is:

```
> n := 2363612653; e := 123456023;
                              n := 2363612653
                              e := 123456023                              (6)
```

Thus, using this public key, the encoded message is:

```
> msg := encodelist(ml, n, e);
       msg := [2248458560, 766345103, 1580137287, 2326909722, 913149839]  (7)
```

Remark: It is always a good idea to test your programs on data for which you know the answer.
For example, if you test your program on the key used in class, then you get:

```
> encodelist(ml, 101284087, 1234567);
              [18463460, 81091624, 39290746, 47738594, 77028351]          (8)
```

Note that these values agree with those given on the overhead in class.

(d) Procedure to find the secret code (using ifactors):
- use ifactors(n) to factor n and to compute phi(n):

```
> crack := proc(n, e) local p, q, k, f, d, a;
    f := ifactors(n); p := f[2, 1, 1]; q := f[2, 2, 1];
    k := (p − 1)·(q − 1); igcdex(e, k, 'd', 'a');
    return(d); end;
crack := proc(n, e)
    local p, q, k, f, d, a;
    f := ifactors(n);
    p := f[2, 1, 1];
    q := f[2, 2, 1];
    k := (p − 1) * (q − 1);
    igcdex(e, k, 'd', 'a');
    return d
end proc                                                        (9)
```

Alternate version (using modp(..) in place of igcdex(..)):

```
> crack := proc(n, e) local p, q, k, f, d;
    f := ifactors(n); p := f[2, 1, 1]; q := f[2, 2, 1];
    k := (p − 1)·(q − 1); d := modp(1/e, k);
    return(d); end;
crack := proc(n, e)
    local p, q, k, f, d;
    f := ifactors(n);
    p := f[2, 1, 1];
    q := f[2, 2, 1];
    k := (p − 1) * (q − 1);
    d := modp(1 / e, k);
    return d
end proc                                                        (10)
```

(e) Find the secret code d for the example in part (c):

```
> d := crack(n, e);
                          d := 1427888927                        (11)
```

Use the secret code d to decode the encoded message msg:

```
> dmsg := encodelist(msg, n, d);
      dmsg := [20080919, 91900, 20151600, 19050318, 5200000]    (12)
```

Check that this is the same as the original message ml:

```
> evalb(dmsg = ml);
                              true                               (13)
```

(f) Deciphering the given encoded message M = [M1, ..., M6] using crack(...):
The encoded message M is:

```
> M := [159081856006493, 158985808365626, 81822460876379,
     117473563599273, 34836171969457, 359056461118547];
M := [159081856006493, 158985808365626, 81822460876379, 117473563599273,
    34836171969457, 359056461118547]                             (14)
```

The known public key is:

```
> n := 378108763156937; e := 9988776543211;
                    n := 378108763156937
                    e := 9988776543211                                  (15)
```

Step 1: Find the secret key d (using the program "crack"):

```
> d := crack(n, e);
                    d := 201785059759891                                (16)
```

Step 2: Decode the given encoded message:

```
> msg2 := encodelist(M, n, d);
msg2 := [9002301142000, 20150013050520, 251521000914, 200805000801,     (17)
    12120001062005, 18000312011919]
```

Step 3: Interpret (by hand) the above message:
Break each number into a sequence of numbers consisting of 2 digits (work from right to left):

9002301142000 = 9|00|23|01|14|20|00 = I*want*

where * represents a blank (space). Similarly, the other numbers are converted, padding with blanks, if necessary. Thus we get:

```
   9002301142000  =  9|00|23|01|14|20|00 = i*want*
  20150013050520 = 20|15|00|13|05|05|20 = to*meet
    251521000914 = 00|25|15|21|00|09|14 = *you*in
    200805000801 = 00|20|08|05|00|08|01 = *the*ha
  12120001062005 = 12|12|00|01|06|20|05 = ll*afte
  18000312011919 = 18|00|03|12|01|19|19 = r*class
```

and so the message is:

"i want to meet you in the hall after class".

Note (for MAPLE experts): We can also use MAPLE to translate our decoded message into words by using the following decoding procedure which converts a list of numbers (viewed as character blocks of length b) into a string:

```
> converttostring := proc(lis, b)
   local blocks, i, j, l, m, r, alphabet, str, bl;
   l := nops(lis); alphabet := `abcdefghijklmnopqrstuvwxyz`;  str := ``;
   for i  to l  do; m := lis[i]; bl := ``;
     for j from 1 to b do; r := modp(m, 100); m := (m-r) / 100;
       if r < 1 or r > 26  then bl := cat(` `, bl)
                      else bl :=
           cat(substring(alphabet, r..r), bl); fi; od;
       str := cat(str, bl); od;
    return(str); end:
```

However, before being able to use this program, we have to figure out what block-length was used. From the above output msg2 in Step 2 we see that all the numbers have 13 or 14 decimal digits, which means that b = 7. Applying the above program to msg2 we obtain:

```
> converttostring(msg2, 7);
                 i want to meet you in the hall after class             (18)
```

This, therefore, gives the same message as we had worked out by hand.

Note that if we had used the wrong block length, then we would have either lost letters or had inserted extra blanks:

```
> converttostring(msg2, 5); converttostring(msg2, 9);
                   want  meetou inhe ha afteclass
              i want   to meet   you in   the ha  ll afte  r class      (19)
```

Comment (for MAPLE experts): Similarly, we could use MAPLE to translate a given text into a sequence of number blocks by using the following conversion program "converttoblocks" (which uses the following program "converttonumberlist"):

(i) converttonumberlist: converts a string "str" to a list of numbers:

```
> converttonumberlist := proc(str) local lis, i; lis := [ ];
    for i to length(str) do;
      lis := [op(lis), searchtext(substring(str, i..i),
          abcdefghijklmnopqrstuvwxyz)]; od;
     return(lis); end:
```

For example, the text "This is top secret"which was used in class is converted as follows:

```
> converttonumberlist(`This is top secret`);
                [20, 8, 9, 19, 0, 9, 19, 0, 20, 15, 16, 0, 19, 5, 3, 18, 5, 20]          (20)
```

(ii) converttoblocks: converts a string into list of numbers corresponding to blocks of length b:

```
> converttoblocks := proc(str, b)
    local lis, blocks, i, j, l, r;
    lis := converttonumberlist(str);
    l := nops(lis); r := irem(l, b);
    if r ≠ 0 then for i to b-r do;
      lis := [op(lis), 0]; od; fi;
    l := nops(lis)/b; blocks := [ ];
    for i from 0 to l - 1 do;
      r := lis[b*i + 1];
      for j from 2 to b do; r := r*100 + lis[b*i + j]; od;
        blocks := [op(blocks), r]; od;
     return(blocks); end:
```

Thus, the message used in class is translated as follows:

```
> m := converttoblocks(`this is top secret`, 4);
                m := [20080919, 91900, 20151600, 19050318, 5200000]          (21)
```