MAS3008 NUMBER THEORY – OPTIONAL MAPLE EXERCISES

This work will not be assessed or taken in for marking, but model solutions will be placed in the Work Room and on the web after the Christmas Vacation. I am happy to discuss these questions with individuals.

An executable MAPLE worksheet, with implementations of the Binary Powering Algorithm, Miller-Rabin Primality Test, and Pollard's $p-1$ and Rho Methods, can be accessed from the Module Page for this module on the SMS website. You can use the programs there to do the exercises below, or, if you prefer, you can write your own.

1. This exercise compares several methods of computing $a^k \bmod n$.

(a) Set `p:=nextprime(100000)` and compute first `2^(p-1) mod p` and then `2&^(p-1) mod p`. Notice how much faster the second one is; here Maple reduces mod p as it goes along. If both seem fast to you, increase p to `nextprime(1000000)`. Of course, the answers should all be 1 by Fermat's Little Theorem.

(b) Use the procedure `powermod(a,k,n)` from the MAPLE worksheet to evaluate $a^k \bmod n$ for some suitable values of a, k and n. A good test is to evaluate `powermod(a,p-1,p)` for various primes p and various bases a; the answer should of course be 1 (as long as $\gcd(a,p)=1$). You should be able to compute `powermod(2,p-1,p)` for the next prime above 2^{20}, 2^{30} or even 2^{50}.

(c) Now write a procedure `powermod2(a,k,n)` of your own to evaluate a^k mod n by a simple loop, using multiplication by a mod n at each stage . Try this out on (some of) the same examples as before. What do you notice about the speed of the two methods?

2. Fermat primality test. Using the `powermod` function, (or MAPLE's built-in function `a&^k mod n`, which uses the same method) you can test some numbers for primality using the Fermat primality test with various bases. Test the numbers $n = 2212374139$ and $n = 19707683773$ with various small bases by computing `powermod(a,n-1,n)`. What do your results tell you? You can verify your conclusion using the Maple functions `isprime` and `ifactor`.

Find ten 10-digit numbers which pass the Fermat test to bases 2, 3, 5 and 7. Are they all prime (as indicated by `isprime`)? The number 1001152801 passes to all these bases, but is not prime. Find the factorisation of this number. Is it a Carmichael number, or is there another base which the Fermat test could use to prove its compositeness?

3. Experiment with the program `spsp(n,a)`, which implements the Miller-Rabin test to determine whether n is a strong pseudoprime to base a. Try it out on the same numbers as in Exercise 2. Does the number 1001152801 pass this test?

4. (a) The procedure `pollard(n,a0,kmax)` attempts to factorise n by Pollard's $p-1$ method, using base a_0 and carrying out up to `kmax` steps. If the method succeeds in finding a factor g of n, it prints out g, n and the number of steps required. Try it out on the following numbers: $n = 403$; $n = 1891$; $n = 5157437$ (these examples were discussed in lectures). Try it out also on the Fermat numbers $F_5 = 2^{32} + 1$ and $F_6 = 2^{64} + 1$. Any success with F_7?

(b) Try to factorise all the Mersenne numbers $M_p = 2^p - 1$ for primes $p < 100$. (Eliminate those which are prime first, using `isprime`.) With $a_0 = 2$ there are no successes. Why? With $a_0 = 3$ and `kmax`=200 you should be able to factorise all with four exceptions. Which are they? Two of these "hard cases" can be factorised by increasing `kmax`, but the other two cannot. Can you explain why?

Repeat with $a_0 = 5$. The results are almost identical. Why?

(c) Try to factorise the numbers $111\ldots111 = (10^k - 1)/9$ for all $k \leq 30$. (These numbers are sometimes called "repunits".) On which of these does your procedure fail when $a0 = 2$ and `kmax`=50. Can you factorise these cases by varying the parameters?

5. Work through the same examples as in Exercise 4, but using the procedure `rho` (which implement Pollard's rho algorithm) instead of the $p-1$ algorithm.

Nigel Byott
November 2002