Mathematics 3360
Computing powers mod m

Recall that we can compute powers mod m by repeatedly squaring and multiplying, reducing mod m at every step to keep the numbers from getting too big. One way to organize this is to use the equations

$$a^{2k+1} = a^{2k} \cdot a, \qquad a^{2k} = (a^2)^k.$$

An algorithm for computing a^e mod m then takes the following form in pseudocode:

```
answer = 1;
while (e > 0)
  {
    if e is odd
      answer = (answer * a) mod m;
    e = e / 2 (with fractional part discarded);
    a = a^2 mod m;
  }
```

At the end of the loop the variable `answer` contains a^e mod m.

The next page contains a C program that implements this algorithm, with rudimentary input and output. It should handle a modulus of about 32 bits on most machines. You can adapt it to your favorite programming language if you want.

```
#include <stdio.h>

unsigned long long
power (unsigned long long a, unsigned long long e, unsigned long long m);

int
main ()
{
  unsigned long long a, e, m;
  printf ("This program computes a^e mod m.\n");
  while (1)
    {
      printf ("Enter m (or 0 to quit): ");
      scanf ("%llu", &m);
      if (m == 0)
        return 0;
      printf ("Enter a: ");
      scanf ("%llu", &a);
      printf ("Enter e: ");
      scanf ("%llu", &e);
      printf ("%llu\n", power (a, e, m));
    }
}

unsigned long long
power (unsigned long long a, unsigned long long e, unsigned long long m)
{
  unsigned long long ans = 1;
  while (e > 0)
    {
      if (e % 2 != 0)
        ans = (ans * a) % m;
      e /= 2;
      a = (a * a) % m;
    }
  return ans;
}
```