CIS 3362 10/20/23

① Factoring Alg

② Fast Modular Exponentiation

Fermat Factoring

$$x^2 - y^2 = (x+y)(x-y)$$

p q

$$N = x^2 - y^2 = (x+y)(x-y) \qquad x^2 > N$$

$$x^2 = N + y^2 \quad (\text{solve for } x^2) \rightarrow x > \sqrt{N}$$

Idea: Start plugging in possible values of x, $x > \sqrt{N}$.

$\rightarrow x^2 - N = y^2$ (test if $x^2 - N$ is a perfect square)

$N = 209 \qquad \sqrt{209} > 14$

x	$x^2 - N$	Is perfect sq?
15	16	yes ✓

$x = 15$, $y = \sqrt{16} = 4 \rightarrow (15+4)(15-4)$

19×11 ✓

$N = 9\overset{4}{\cancel{3}}9$ $\quad \sqrt{939} > 30$

x	$x^2 - N$	Is perfect sq?
31	$961 - 949 = \cancel{22}\ 12$	NO
32	$1024 - 949 = 75$	NO
33		NO
⋮		
43	$1849 - 949 = 900$	YES

$$\begin{array}{r} 43 \\ \times\ 43 \\ \hline 129 \\ 172 \\ \hline 1849 \\ -949 \\ \hline 900 \end{array}$$

$x = 43,\ y = \sqrt{900} = 30 \rightarrow (43+30)(43-30)$

73×13

GOOD ⇒ When #s are "close to each other"

BAD ⇒ far apart

Pollard-Rho

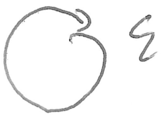
a = 2, 5, 26, 677, X

b = 2, 26, X

Imagine $n = p \times q$

Imagine mod p

mod q




```
a = 2
b = 2
while (true) {
    a = (a*a+1) % n;
    b = (b*b+1) % n;
    b = (b*b+1) % n;
    d = gcd(a-b, n);
    if (d>1 && d<n) return [d, n/d];
    if (d == n) return failed;
}
```

Fast Modular Exponentiation

$base^{exp}$ % mod

Slow alg

```
int slowmodexpo(int b, int e, int m) {
    int res = b % m;
    for (int i = 0; i < e; i++)
        res = (res * b) % m;
    return res;
}
```

to make mod expo chart

SLOW O(e)

How to improve? → Same

$b^{1000} = b^{500} \times b^{500}$

if e is even just calc $b^{e/2}$ % m.
Then square this! $b^{1001} = b^{1000} \times b$

```
int fastmodexpo(int b, int e, int m) {
    if (e == 0) return 1 % m;
    if (e % 2 == 0) {
        int tmp = fastmodexpo(b, e/2, m);
        return (tmp * tmp) % m;
    }
    return (b * fastmodexpo(b, e-1, m)) % m;
}
```

1001 → 1000 → 500 → 250 → 125 → 124 → 62 → 31 → ...
→ 15 → 14 → 7 → 6 → 3 → 2 → 1

WORST CASE 2 step → div 2
$\boxed{\le 2\log_2 exp}$

BUILT-IN python pow(base, exp, mod)

BUILT-IN <u>Java BigInteger</u>

base.modPow(exp, mod)