Thm 2.13 = for prime p and integer k. $\phi(p^k) = p^{k-1}(p-1)$

Proof: Since p is prime, p^k is divisible by multiples of p. Therefore, any number that is multiple of p is not relatively prime to p^k.

$p, 2p, \quad . \quad , (p^{k-1} - 1)p$

Also 0 is not in $\mathbb{Z}^*_{p^k}$.

Therefore, $\phi(p^k) = p^k - p^{k-1}$

$= p^{k-1}(p-1)$

Thm 2.14: Given $n = p_1^{k_1} \ldots p_t^{k_t}$ $\quad \phi(n) = \phi(p_1^{k_1}) \cdot \phi(p_2^{k_2}) \cdots \phi(p_t^{k_t})$

$\phi(n) = \prod_{i=1}^{t} p_i^{k_i}(1 - \frac{1}{p_i}) = n \prod_{i=1}^{t} (1 - \frac{1}{p_i})$

Multiplicative order of $a \bmod n$ is the smallest of k such that $a^k \equiv 1 \bmod n$

$3^1 = 3 \bmod 7$ $\qquad 3^6 = 1 \bmod 7$

$3^2 = 2$ "

$3^3 = 6$ "

$3^4 = 4$ "

$3^5 = 5$

Thm 2-15: For any positive n and any a s.t $\gcd(a,n)=1$ then $a^{\Phi(n)} = 1 \bmod n$

Proof: Define $f: Z_n^* \to Z_n^*$

$f(b) = a \cdot b \bmod n$ $\quad f(\beta) = f(\beta')$

f is injective : $a\beta = a\beta' \bmod n \Rightarrow \beta = \beta' \bmod n$

f is surjective : because f is injective map onto itself and $|Z_n^*|$ is finite. (why finiteness is needed ?)

so $\prod_{b \in Z_n^*} b = \prod_{b \in Z_n^*} \overbrace{ab}^{f(b)} \Rightarrow a^{\Phi(n)} \prod_{b \in Z_n^*} b = \prod_{b \in Z_n^*} b$

$\Rightarrow a^{\Phi(n)} = 1$ ∎

How to calculate $a^k \bmod n$?

Naive way:

```
res=1;
for (i=1; i<=k; i++)
    res *= a mod n;
return res;
```

Requires $O(k)$ len(n)-bit multiplication

since each multiplication mod n is $O(\text{len}(n)^2)$

so Naive approach takes $O(k \cdot \text{len}^2(n))$ time

Faster Approach:

Let $k = (b_{l-1} \cdots b_0)$

```
res = 1
for (i = l-1; i >= 0; i--)
{   res = res * res mod n;
    if (bi == 1)
        res = res . a (mod n);
}
return res;
```

why it works? $a^5 \bmod n$

$5 = (101)_2$

res = 1

res = 1

res = a

res = a^2

res = a^4

res = a^5

Define $k_i = \lfloor k / 2^i \rfloor$ $= k_1 = 2$

$\beta_i = a^{k_i}$

$\beta_l = 1$ & $\beta_0 = a^e$

Note

$k_i = 2k_{i+1} + b_i$

$\beta_i = \beta_{i+1}^2 \cdot a^{b_i}$

Clearly repeated squaring

requires $O(\mathrm{len}(k) \cdot \mathrm{len}^2(n))$

Euclid's Algorithm

Goal: Calculate $\gcd(a, b)$!

Assume w.l.g. $a \geq b \geq 0$

if $b = 0 \Rightarrow \gcd(a, 0) = a$

if $b > 0$, basic idea is the following

$a = bq + r$ where $0 \leq r < b$

$\underline{\gcd(a, b) = \gcd(b, r)}$

Thm 4.1 : $a, b \in \mathbb{Z}$ & $a \geq b \geq 0$

Define r_i, q_i such that

$a = r_0$, $b = r_1$

$r_0 = r_1 q_1 + r_2 \quad (0 < r_2 < r_1)$

$\vdots$

$r_{i-1} = r_i q_i + r_{i+1} \quad (0 < r_{i+1} < r_i)$

$\vdots$

$r_{\ell-2} = r_{\ell-1} \cdot q_{\ell-1} + r_\ell \quad (0 < r_\ell < r_{\ell-1})$

$r_{\ell-1} = \underline{r_\ell} \cdot q_\ell + 0 \quad (r_{\ell+1} = 0)$

Then $r_\ell = \gcd(a, b)$. If $b > 0$ then

$\phi = \frac{(1+\sqrt{5})}{2}$ $\qquad \ell \leq \frac{\log b}{\log \phi} + 1$

$\boxed{\text{Proof:}}$

Note that

$$\gcd(a,b) = \gcd(r_0, r_1) = \gcd(r_\ell, r_{\ell+1})$$
$$= \gcd(r_\ell, 0) = r_\ell$$

Given $b>0$, we use induction

Clearly, $\ell \le \dfrac{\log b}{\log \phi} + 1$ if $\ell = 1$

Assume $\ell > 1$

$$\ell \le \frac{\log b}{\log \phi} + 1 \Rightarrow \ell \le \log_\phi b + 1$$
$$\Rightarrow \phi^{\ell-1} \le b \Rightarrow \phi^{\ell-1} \le r_1$$

so if we prove that $r_{\ell-i} \ge \phi^i$ for $i = 0, \dots, \ell$, we are done

$r_\ell \ge \phi^0 = 1$ (why?)

$r_{\ell-1} \ge r_\ell + 1 \ge 2 > \phi^1$

Using induction

$$r_{\ell-i} \ge r_{\ell-(i-1)} + r_{\ell-(i-2)} \quad \text{(why?)}$$
$$\ge \phi^{i-1} + \phi^{i-2} \quad \text{(why?)}$$
$$= \phi^{i-2}(1+\phi) = \phi^i \quad \text{(why?)}$$

$\square$

Example gcd(100, 35)

$100 = 35 \cdot 2 + 30$

$35 = 30 \cdot 1 + 5$

$30 = 5 \cdot 6$

so $\gcd(100, 35) = 5$

```
gcd (a, b)
{   r = a;
    r' = b;

    while (r' > 0)
    {   r'' = r mod r';
        r = r';
        r' = r'';
    }
    return r;
}
```

Thm 4.2. Euclid Algorithm runs in time $O(\mathrm{len}(a) \cdot \mathrm{len}(b))$

Proof: Running time is $O\left(\sum_{i=1}^{\ell} \mathrm{len}(r_i) \cdot \mathrm{len}(q_i)\right)$

$$\sum_{i=1}^{\ell} \mathrm{len}(r_i) \cdot \mathrm{len}(q_i) \le \mathrm{len}(b) \sum_{i=1}^{\ell} \mathrm{len}(q_i) \leftarrow$$

$$\leq \text{len}(b)\left(\sum_{i=1}^{l}\left(\lfloor \log_2 q_i \rfloor + 1\right)\right)$$

$$= \text{len}(b)\left(l + \log_2\left(\prod_{i=1}^{l} q_i\right)\right)$$

Note

$a = r_0 \geq r_1 q_1 \cdots \geq q_l \cdots q_1$

Also $\quad l \leq \dfrac{\log b}{\log \phi} + 1$

Running time $\leq \text{len}(b)\left(\dfrac{\log b}{\log \phi} + 1 + \log_2(a)\right)$

$O(\text{len}(b) \cdot \text{len}(a))$

Thm 4.3

Let $a, b, r_0, r_1 \ldots r_{l+1}$
$q_1, \ldots, q_l$

be defined as in Thm 4-1

Define $s_0, \ldots, s_{l+1}$ and $t_0, \ldots t_{l+1}$

$s_0 = 1 \quad , \quad t_0 = 0$ and for $i = 1, \ldots, l$, define
$s_1 = 0 \quad , \quad t_1 = 1$

$s_{i+1} = s_{i-1} - s_i q_i$
$t_{i+1} = t_{i-1} - t_i q_i$

Then

for $i = 0, \dots$ to $\ell + 1$

$\Rightarrow s_i a + t_i \cdot b = r_i \Rightarrow s_\ell a + t_\ell b = \gcd(a, b)$

plus

Proof:

For $i = 0$ & $i = 1$

$s_0 a + t_0 b = 1 \cdot a + 0 b = r_0 = a$ ✓

$s_1 a + t_1 b = 0 \cdot a + 1 \cdot b = r_1 = b$ ✓

Assume this is true for 0 up to $i - 1$

$$s_i a + t_i b = (s_{i-2} - s_{i-1} q_{i-1}) a + (t_{i-2} - t_{i-1} q_{i-1}) b \quad \text{(by def.)}$$

$$= (s_{i-2} a + t_{i-2} b) - (s_{i-1} a + t_{i-1} b) q_{i-1}$$

$$= r_{i-2} - r_{i-1} q_{i-1} \quad \text{(why ?)}$$

$$= r_i$$

$\square$

$100 = 35 \cdot 2 + 30 \rightarrow$

$s_2 = s_0 - s_1 q_1 = 1 - 0 = 1$

$t_2 = t_0 - t_1 q_1 = 0 - 2 = -2$

$100 \cdot 1 + -2 \cdot 35 = 30$

$35 = 30 \cdot 1 + 5$

$s_3 = s_1 - s_2 q_2 = 0 - 1$
$= -1$

$t_3 = t_1 - t_2 \cdot q_2 = 1 - (-2) \cdot 1$
$= 3$

$-1 \cdot 100 + 3 \cdot 35 = 5$

$30 = 5 \cdot 6 + 0$

ERROR: undefined
OFFENDING COMMAND:

STACK: