Modular Arithmetic

- · Addition: O(n)
- · Multiplication: O(n^2) *(naive)*
- · Multiplication: O(nlogn) *(FFT)*
- · Euclid's Rule: gcd(x, y) = gcd(x mod y, y)
- · # of bits in x^y = $y\log_2 x \leq 2^n \times n$
- · $\sum_{i=0}^{\infty} r^i = \frac{1}{1-r}$, if r < 1
- · $\sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$
- · $\sum_{i=0}^{n} \frac{1}{i} = O(\log_2 n)$

Extended Euclid's GCD(x,y)

O(n^3); gcd(x,y) = d = xi + yb; x ≥ y; # mod x

```
ext-gcd(x,y):
if y == 0:  return (x, 1, 0)
else:
  (d, a, b) = ext-gcd(y, x mod y)
  return (d, b, a-x/y·b)
```

#	X	Y	X/Y	X%Y	#	d	a	b
1.	26	15	1	11	6.	1	1	0
2.	15	11	1	4	5.	1	0	1-(3*0)
3.	11	4	2	3	4.	1	1	0-(1*1)
4.	4	3	1	1	3.	1	-1	1-(2*-1)
5.	3	1	3	0	2.	1	3	-1-(1*3)
6.	1	0			1.	1	-4	3-(1*-4)

Fermat's Little Theorem

if p is prime, then $\forall\ 1 \leq a < p$
$a^{p-1} = 1$ mod p

Proof: *Start by listing first p-1 positive multiples of a:*
$S = \{a,\ 2a,\ 3a,\ \cdots\ (p-1)a\}$
Suppose that ra and sa are the same mod p, ⇒ r = s mod p
∴ set S of p-1 multiples of a are distinct and nonzero, that is, they must be congruent to 1, 2, 3, ⋯ p-1 after being sorted.
Multiply all congruences together and we find
$a\cdot 2a\cdot 3a\cdots(p-1)\cdot a = 1\cdot 2\cdot 3\cdots(p-1)$ (mod p) or better,
$a^{(p-1)}(p-1)! = (p-1)!$ mod p. Divide both side by (p-1)! ■

Modular Exponentiation

x^y mod N → start with repeated squaring mod N
x mod N → x^2 mod N → $(x^2)^2 \cdots x^{log_2 y}$ mod N
each step takes O($\log^2 N$) times to compute and
there are $\log_2$y steps, ∴ ∈ O(n^3),
where n is the # of bits in N

Formal Limit Proof

$\lim_{n\to\infty} \frac{f(n)}{g(n)}$:
$\geq 0\ (\infty) \Rightarrow f(n) \in \Omega(g(n))$
$< \infty\ (0) \Rightarrow f(n) \in O(g(n))$
$= c_{|0<c<\infty} \Rightarrow f(n) \in \Theta(g(n))$

Logarithm Tricks

$\log_b x^p = p\log_b x$
$\frac{ln(x)}{ln(m)} = \log_m x$
$x^{log_b y} = y^{log_b x}$

Complexity Hierarchy

Exponential
Polynomial
Logarithmic
Constant

Master's Theorem

T(n) = aT($\frac{n}{b}$) + O(n^d), if a >0, b>1, $d \geq 0$

$$T(n) = \begin{cases} O(n^d) \text{ if d} > \log_b a \\ O(n^d log_b n) \text{ if d} = \log_b a \\ O(n^{log_b n}) \text{ if d} < \log_b a \end{cases}$$
