Modular Arithmetic

- Addition: O(n)
- Multiplication: O(n^2) *(naive)*
- Multiplication: O(nlogn) *(FFT)*
- # 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 $r < 1$

Formal Limit Proof

$\lim_{n\to\infty} \frac{f(n)}{g(n)}$:

≥ 0 (∞) $\Rightarrow$ 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 $a > 0, b > 1, d \geq 0$

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