Pragmatic Programming

Session 6 - Geometry, Number Theory, and Probability

Max Nilsson

November 30th

Combinatorics

- The principle of inclusion and exclusion:
  - $\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{\emptyset \neq J \subset \{1,\ldots,n\}} (-1)^{|J|+1} \left|\bigcap_{j \in J} A_j\right|$
  - Example: Calculate how many numbers in $\{1, \ldots, n\}$ which are relatively prime to every prime in some set $\mathcal{P}$ consisting of primes.
- Compute Binomial coefficients $\binom{m}{n}$ modulo a prime p.
  - Lucas's Theorem: Write $m = \sum_{r=0}^{k} m_r p^r, n = \sum_{r=0}^{k} n_r p^r$ then

$$\binom{m}{n} \equiv_{\mathbb{Z}_p} \prod_{r=0}^{k} \binom{m_r}{n_r}$$

Probability

- I have really nothing to add here since I am not very good at these types of problems.
- Sometimes they reduce to a `dp` formulation, where the states contain `double`. Then you have to pray to Kattis that multiplying a bunch of `double` or `long double` does not ruin your final answer. See Memory for such a problem.
- Other times you have to use basic probability methods, such as Bayes' Theorem

$$\mathbb{P}(A|B)\mathbb{P}(B) = \mathbb{P}(B|A)\mathbb{P}(A)$$

and expected values

$$\mathbb{E}[X] = \int X \mathrm{d}\mathbb{P}.$$

. See Infection Estimation for such a problem.

Number Theory

- Primality test of a lot of small ($\leq 10^6$) numbers - Use Sieve of Eratosthenes with complexity $\mathcal{O}(n \log \log n)$ (Eratosthenes was a Greek mathematician born 276 BC and was the first international grandmaster on codeforces).
- Primality test of some large ($\leq 10^{18}$) numbers - Use Miller-Rabin with complexity $\mathcal{O}(k \log^3 n)$ where k is dependent on the input.
- Factor numbers - Use a preprocessed Sieve of Eratosthenes or Pollard's Rho algorithm with complexity $\mathcal{O}(n^{1/4})$.
- For finding the greatest common divisor, use __gcd, and for finding a solution (x_0, y_0) to the linear Diophantine equation

$$ax + by = \gcd(a, b)$$

use the Euclidean algorithm with time complexity $\mathcal{O}(\log n)$.

Geometry

This is probably the deepest of our topics today. Most of the geometry will be in $\mathbb{R}^2$ or even $\mathbb{Z}^2$. Always avoid `double`'s as much as possible!

- Many simple problems can actually be quite tricky to figure out. Example: given two segments in $\mathbb{R}^2$, decide whether they intersect or not.
- Computing the convex hull in $\mathbb{R}^2$ can be done in $\mathcal{O}(n \log n)$ with either a divide and conquer method or a Graham scan.
- The convex hull in $\mathbb{R}^3$ can be done with an incremental convex hull algorithm in $\mathcal{O}(n^2)$.
- The area of a (non-convex) polygon can be found by

```
T polygonArea2(vector<Point<T>>& v) {
    T a = v.get_back().cross(v[0]);
    rep(i,0,sz(v)-1) a += v[i].cross(v[i+1]);
    return a;}
```

- The minimum circle that encloses a collection of points in $\mathbb{R}^2$ can be found with `dp` and with complexity $\mathcal{O}(n)$!

This Week

- Skim read chapter 15 and 16 in Sannemo[1] and chapter 13 in Laaksonen[2].
- Also check out this resource for a text on competetive computational geometry.
- Solve half of this weeks 10 problems.

[1] Johan Sannemo. "Principles of Algorithmic Problem Solving". In: *Draft version* (2018).

[2] Antti Laaksonen. *Guide to competitive programming*. Springer, 2020.