CS 97SI: INTRODUCTION TO PROGRAMMING CONTESTS

Jaehyun Park

Today's Lecture

- Algebra
- Number Theory
- Combinatorics
- (non-computational) Geometry

- Emphasis on "how to compute"

Sum of Powers

- $\sum_{k=1}^{n} k^2 = \frac{1}{6} n(n+1)(2n+1)$
- $\sum k^3 = (\sum k)^2 = \left(\frac{1}{2} n(n+1)\right)^2$

- Pretty useful in many random situations
- Memorize above!

Fast Exponentiation

- $a^n =$
  - 1, if $n = 0$
  - a, if $n = 1$
  - $\left(a^{n/2}\right)^2$, if n is even
  - $\left(a^{(n-1)/2}\right)^2 \cdot a$, if n is odd

- Can be computed recursively

Implementation (recursive)

```
double pow(double a, int n) {
     if(n == 0) return 1;
     if(n == 1) return a;
     double t = pow(a, n/2);
     return t * t * pow(a, n%2);
}
```

- Running time: $O(\log n)$

Implementation (non-recursive)

```
double pow(double a, int n) {
      double ret = 1;
      while(n) {
            if(n%2 == 1) ret *= a;
            a *= a; n /= 2;
      }
      return ret;
}
```

- You should understand how it works

Linear Algebra

- Gaussian Elimination
  - Solve a system of linear equations
  - Invert a matrix
  - Find the rank of a matrix
  - Compute the determinant of a matrix
- Cramer's Rule

Greatest Common Divisor (GCD)

- $\gcd(a, b)$: greatest integer divides both a and b
- Used very frequently in number theoretical problems

- Some facts:
  - $\gcd(a, b) = \gcd(a, b - a)$
  - $\gcd(a, 0) = a$
  - $\gcd(a, b)$ is the smallest positive number in $\{ax + by \mid x, y \in Z\}$

Euclidean Algorithm

- Repeated use of $\gcd(a, b) = \gcd(a, b - a)$
- $\gcd(1989, 867)$

$$\begin{aligned}
&= \gcd(1989 - 2 \times 867, 867) \\
&= \gcd(255, 867) \\
&= \gcd(255, 867 - 3 \times 255) \\
&= \gcd(255, 102) \\
&= \gcd(255 - 2 \times 102, 102) \\
&= \gcd(51, 102) \\
&= \gcd(51, 102 - 2 \times 51) \\
&= \gcd(51, 0) \\
&= 51
\end{aligned}$$

Implementation

```
int gcd(int a, int b) {
    while(b){int r = a % b; a = b; b = r;}
    return a;
}
```

- Running time: $O(\log(a+b))$
- Be careful: a % b follows the sign of a
  - `5 % 3 == 2`
  - `-5 % 3 == -2`

Congruence & Modulo Operation

- $x \equiv y \pmod{n}$ means x and y have the same remainder when divided by n

- Multiplicative inverse
  - x^{-1} is the inverse of x modulo n if $xx^{-1} \equiv 1 \pmod{n}$
  - $5^{-1} \equiv 3 \pmod{7}$ because $5 \cdot 3 \equiv 15 \equiv 1 \pmod{7}$
  - May not exist (e.g. Inverse of 2 mod 4)
  - Exists iff $\gcd(x, n) = 1$

Multiplicative Inverse

- All intermediate numbers computed by Euclidean algorithm are integer combinations of a and b
  - Therefore, $\gcd(a, b) = ax + by$ for some integers x, y
  - If $\gcd(a, n) = 1$, then $ax + ny = 1$ for some x, y
  - Taking modulo n gives $ax \equiv 1 \pmod{n}$

- We will be done if we can find such x and y

Extended Euclidean Algorithm

- Main idea: keep the original algorithm, but write all intermediate numbers as integer combinations of a and b

- Exercise: implementation!

Chinese Remainder Theorem

- Given a, b, n, m such that n and m are coprime
- Find x such that $x \equiv a \pmod{m}$, $x \equiv b \pmod{n}$

- Solution:
  - Let n^{-1} be the inverse of n modulo m
  - Let m^{-1} be the inverse of m modulo n
  - Set $x = ann^{-1} + bmm^{-1}$ (check this yourself)
- Extension: solving for more simultaneous equations

Binomial Coefficients

- $\binom{n}{k}$ is the number of ways to choose k objects out of n distinguishable objects
  - or, the coefficient of $x^k y^{n-k}$ in the expansion of $(x+y)^n$
- Appears everywhere in combinatorics

Computing $\binom{n}{k}$

- Solution 1: Compute using the following formula

$$\binom{n}{k} = \frac{n(n-1)\cdots(n-k+1)}{k!}$$

- Solution 2: Use Pascal's triangle

- Case 1: Both n and k are small
  - Use either solution
- Case 2: n is big, but k or $n-k$ is small
  - Use Solution 1 (carefully)

Fibonacci sequence

- Definition:
  - $F_0 = 0, F_1 = 1$
  - $F_n = F_{n-1} + F_{n-2}$, where $n \geq 2$

- Appears in many different contexts

Closed Form

- $F_n = \frac{\varphi^n - \overline{\varphi}^n}{\sqrt{5}}$, where $\varphi = \frac{1+\sqrt{5}}{2}$

- Bad because φ and $\sqrt{5}$ are irrational
- Cannot compute the exact value of F_n for large n

- There is a more stable way to compute F_n
  - … and any other recurrence of a similar form

Better Closed Form

- $$\begin{bmatrix} F_{n+1} \\ F_n \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} F_n \\ F_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n \begin{bmatrix} F_1 \\ F_0 \end{bmatrix}$$
- Use fast exponentiation to compute the matrix power
- Can be extended to support any linear recurrence with constant coefficients

Geometry

- In theory: not that hard
- In programming contests: avoid if you can…

- Will cover basic stuff today
  - Computational geometry in week 9

When Solving Geometry Problems

- Precision, precision, precision!
  - If possible, don't use floating-point numbers
  - If you have to, always use `double` and never use `float`
  - Avoid division whenever possible
  - Introduce small constant ϵ in (in)equality tests
    - e.g. Instead of `if(x == 0)`, write `if(abs(x) < EPS)`
- No hacks!
  - In most cases, randomization, probabilistic methods, small perturbations won't help

2D Vector Operations

- Have a vector (x, y)
- Norm (distance from the origin): $\sqrt{x^2 + y^2}$
- Counterclockwise rotation by θ:
  Left-multiply by $\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$
  - Make sure to use correct units (degrees, radians)
- Normal vectors: $(y, -x)$, $(-y, x)$
- Memorize all of them!

Line-Line Intersection

- Have two lines: $ax + by + c = 0, dx + ey + f = 0$
- Write in matrix form:
  - $\begin{bmatrix} a & b \\ d & e \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = -\begin{bmatrix} c \\ f \end{bmatrix}$
- Invert the matrix on the left using the following:
  - $\begin{bmatrix} a & b \\ d & e \end{bmatrix}^{-1} = \frac{1}{ae-bd} \begin{bmatrix} e & -b \\ -d & a \end{bmatrix}$
  - Memorize this!
- Edge case: $ae = bd$

Circumcircle of a Triangle

- Have three points A, B, C
- Want to compute P that is equidistance from A, B, C

- Don't try to solve the system of quadratic equations!
- Instead, do the following:
  - Find the bisectors of AB and BC
  - Compute their intersection

Area of a Triangle

- Have three points A, B, C
- Use cross product: $2 \cdot area = |(B - A) \times (C - A)|$

- Cross product:
  - $(x_1, y_1) \times (x_2, y_2) = \begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix} = x_1 y_2 - x_2 y_1$
  - Very important in computational geometry. Memorize!

Area of a Simple Polygon

- Given $P_1, P_2, \dots, P_n$ around perimeter of polygon P
- If P is convex, we can decompose P into triangles:
  - $2 \cdot area = \left|\sum_{i=2}^{n-1} (P_{i+1} - P_1) \times (P_i - P_1)\right|$
- It turns out that the formula above works for non-convex polygons too
  - Area is the absolute value of the sum of "signed area"
- Alternative formula:
  - $2 \cdot area = \left|\sum_{i=1}^{n} (x_i y_{i+1} - x_{i+1} y_i)\right|$

Conclusion

- No need to look for one-line closed form solutions
- Multi-step algorithms are good enough
  - This is a CS class, after all

- Have fun with the exercise problems
  - … and come to the practice contest if you can!