2022-1학기 스터디 #3

나정휘
https://justiceHui.github.io/

목차

용어 정의

용어 정의

- 약수, 배수, 최대공약수, 최소공배수, 서로소
  - 정수 a, b에 대해 $b = an$을 만족하는 정수 n이 존재하면 a는 b의 약수, b는 a의 배수
    - $a|b$: a가 b를 나눈다, b는 a로 나누어진다.
  - $a = b = 0$이 아닌 정수 a, b에 대해, $g|a, g|b$를 만족하는 가장 큰 자연수 g : 최대공약수
  - $a|l, b|l$를 만족하는 가장 작은 자연수 l: 최소공배수
  - a와 b의 최대공약수가 1이면 a와 b는 서로소
  - $ab = gl$
    - $a = ga', b = gb'$라고 하면 a'과 b'은 서로소
    - $l = ga'b'$이므로 $ab = g^2a'b' = gl$임
    - $l = ab/g$

용어 정의

- 몫, 나머지, 합동
  - 정수 a와 0이 아닌 정수 b가 있을 때, $a = bq + r, 0 \le r < |b|$를 만족하는 정수 q, r은 유일함
    - q는 몫, r은 나머지
    - 주의: C/C++에서 음수 나눗셈은 나머지가 음수가 나올 수 있음
      - 5 % 3 = 2, -5 % -3 = -2
      - -5 % 3 = -2, 5 % -3 = 2
  - 정수 a, b와 0이 아닌 정수 n이 있을 때, $n|(a - b)$이면 a와 b가 $(mod\ n)$에서 합동
    - $a \equiv b\ (mod\ n)$
    - a와 b를 n으로 나눈 나머지가 동일

용어 정의

- 합동
  - 반사성, 대칭성, 추이성
    - $a \equiv a(mod\ n)$
    - $a \equiv b(mod\ n)$이면 $b \equiv a(mod\ n)$
    - $a \equiv b(mod\ n)$이고 $b \equiv c(mod\ n)$이면 $a \equiv c(mod\ n)$
  - 사칙연산
    - $a \equiv b(mod\ n)$이면
    - 정수 c에 대해, $a \pm c \equiv b \pm c(mod\ n)$
    - 0이 아닌 정수 c에 대해, $ac \equiv bc(mod\ n)$
    - 나눗셈은 성립 안 함

용어 정의

- 소수
  - 약수가 1과 자기 자신밖에 없는 2 이상의 자연수
  - 소수는 무한히 많음
    - 소수가 유한하다고 가정하고 귀류법 사용하면 증명 가능
  - 임의의 자연수 n에 대해, 소수가 등장하지 않는 길이 n인 구간 존재
    - $2 \le k \le n+1$일 때 $(n+1)!+k$는 k의 배수이므로 소수가 아님
  - 임의의 자연수 n에 대해, $n < p \le 2n$인 소수 p가 존재
    - 베르트랑 공준
  - $\pi(x)$를 x이하 소수의 개수라고 하면, $\lim_{n\to\infty} \frac{\pi(x)\log x}{x} = 1$이 성립함
    - x이하 소수의 개수는 $O(\frac{x}{\log x})$개
    - 소수 정리

용어 정의

- 2 이상의 자연수 n을 소수들의 곱으로 표현하는 것: 소인수분해
  - $n = {p_1}^{e_1}{p_2}^{e_2} \cdots {p_k}^{e_k}$
  - 소수들의 순서만 다른 경우를 같은 표현으로 보면, 소인수분해의 결과는 유일하게 존재
    - 2개 이상이라고 가정하고 귀류법 사용하면 증명 가능
  - n의 약수는 $p1^{f_1}{p_2}^{f_2} \cdots {p_k}^{f_k}$꼴 (단, $0 \le f_i \le e_i$)

질문?

거듭제곱

거듭제곱

- 목표: 음이 아닌 정수 a, b, c에 대해 $a^b (mod\ c)$를 구하는 것
  - $b = 0 \;\Rightarrow a^b = 1$
  - $2 \mid b \;\Rightarrow a^b = (a^{b/2})^2$
  - $2 \nmid b \;\Rightarrow\; a^b = a(a^{(b-1)/2})^2$
  - $a < c$이면 계산 과정 도중에 나오는 값은 c^2미만
  - a대신 $a\%c$를 사용하면 됨

거듭제곱

- BOJ 1629 곱셈
  - 2147483647($= 2^{31} - 1$) 이하의 자연수 a, b, c가 주어졌을 때
  - $a^b \ mod \ c$를 구하는 문제
  - 계산 도중에 int 범위를 넘어갈 수 있으므로 long long 사용
  - 시간 복잡도: $O(\log b)$

```cpp
#include <bits/stdc++.h>
using namespace std;

long long Pow(long long a, long long b, long long c){
    if(b == 0) return 1;
    long long half = Pow(a, b/2, c);
    if(b % 2 == 0) return half * half % c;
    else return a * half % c * half % c;
}

int main(){
    long long a, b, c;
    cin >> a >> b >> c;
    cout << Pow(a%c, b, c);
}
```

거듭제곱

- BOJ 1629 곱셈
  - $b = 5$ 일 때 b를 이진법으로 나타내 보면 $101_{(2)}(2^2 + 2^0)$
  - a^5는 $a^4 \times a^1$로 나타낼 수 있음
  - $a^1, a^2, a^4, a^8, \cdots$ 를 알고 있다면 a^b를 구할 수 있음
  - 시간 복잡도: $O(\log b)$

```
long long Pow(long long a, long long b, long long c){
    long long res = 1;
    while(b > 0){
        if(b % 2 == 1) res = res * a % c;
        b /= 2;
        a = a * a % c;
    }
    return res;
}
```

질문?

소수 판별

소수 판별

- 목표: 양수 n이 주어졌을 때 n이 소수이면 1, 소수가 아니면 0 반환하는 함수 작성
  - $n = 1$이면 0 반환
  - 소수의 약수는 1과 자기 자신밖에 없으므로 $2,3,\cdots,n-1$중 하나로 나누어 떨어지면 0 반환
  - 시간 복잡도: $O(n)$

  - 사실 $\sqrt{n}$ 이하만 확인해도 됨
    - $n = pq\ (\mathrm{p} \le q)$일 때 $p \le \sqrt{n}$ 을 만족함
    - $p, q > \sqrt{n}$ 이면 $n < pq$
    - 동일한 방식으로 n의 모든 약수를 찾을 수 있음
  - 시간 복잡도: $O(\sqrt{n})$

```
int IsPrime(int n){
    if(n < 2) return 0;
    for(int i=2; i*i<=n; i++){
        if(n % i == 0) return 0;
    }
    return 1;
}
```

소인수분해

소인수분해

- 목표: 양수 n의 소인수를 모두 출력하는 것
  - n은 $\sqrt{n}$보다 큰 소인수를 중복을 포함해 최대 한 개 가질 수 있음
  - $\sqrt{n}$이하의 소수로 모두 나눠보면 됨
  - 만약 마지막에 $n \neq 1$이라면 n도 소인수
  - 시간 복잡도: $O(\sqrt{n})$
    - 실제로 n를 나누는 건 $O(\log n)$번만 하면 됨

```
void Factorize(int n){
    for(int i=2; i*i<=n; i++){
        while(n % i == 0){
            cout << i << "\n";
            n /= i;
        }
    }
    if(n != 1) cout << n << "\n";
}
```

질문?

에라토스테네스의 체

에라토스테네스의 체

- 목표: 1부터 n까지의 소수를 모두 구하는 것
  - 2는 소수, 2의 배수를 전부 제거
  - 3은 소수, 3의 배수를 전부 제거
  - 4는 이미 제거됨
  - 5는 소수, 5의 배수를 전부 제거
  - 6은 이미 제거됨
  - 7은 소수, 7의 배수를 전부 제거
  - …

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30

에라토스테네스의 체

- 시간 복잡도
  - n이하의 소수 p에 대해, n보다 작거나 같은 p의 배수를 제거
  - $\sum \frac{n}{p} = n \sum \frac{1}{p}$
  - $\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} = O(\ln n)$
    - 1/x 의 적분을 생각해보자.
  - $n \sum \frac{1}{p} \leq n \sum_{i=1}^{n} \frac{1}{i} = O(n \log n)$

  - 사실 $\sum \frac{1}{p} = O(\ln \ln n)$ 이라서 실제 시간 복잡도는 $O(n \log \log n)$
    - Mertens’ second theorem

에라토스테네스의 체

- BOJ 15965 K번째 소수
  - K ≤ 500’000 번째 소수를 출력하는 문제
  - 소수 정리에 의해 K번째 소수는 약 O(K ln K)
    - 800만 이하에서 나옴

  - 사실 j는 i*i부터 시작해도 됨
    - 2i, 3i, 4i, … , (i-1)i는 이미 앞에서 지워짐

```
#include <bits/stdc++.h>
using namespace std;

int Check[8080808];
vector<int> Primes;

void Sieve(int n){
    for(int i=2; i<=n; i++){
        if(Check[i]) continue;
        Primes.push_back(i);
        for(int j=i+i; j<=n; j+=i) Check[j] = 1;
    }
}

int main(){
    Sieve(8000000);
    int K; cin >> K;
    cout << Primes[K-1];
}
```

질문?

에라토스테네스의 체

- 가장 작은 소인수
  - 소수 p의 배수를 지우는 과정을 다시 생각해보자.
  - 만약 x가 p에 의해서 처음으로 지워졌으면
  - p는 x의 가장 작은 소인수

  - 소인수분해
    - x의 가장 작은 소인수를 sp[x]라고 하면
    - while(x > 1) x /= sp[x];
    - 소인수분해를 O(log x)에 할 수 있음
      - x는 매번 절반 이상 감소

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30

```cpp
#include <bits/stdc++.h>
using namespace std;

int SP[5050505];
void Sieve(int n){
    for(int i=2; i<=n; i++){
        if(SP[i]) continue;
        SP[i] = i;
        for(int j=i+i; j<=n; j+=i) if(!SP[j]) SP[j] = i;
    }
}

int main(){
    ios_base::sync_with_stdio(false); cin.tie(nullptr);
    Sieve(5000000);
    int N; cin >> N;
    for(int i=1; i<=N; i++){
        int X; cin >> X;
        while(X > 1){
            cout << SP[X] << " ";
            X /= SP[X];
        }
        cout << "\n";
    }
}
```

질문?

유클리드 호제법

유클리드 호제법

- 최대공약수의 성질
  - $\gcd(a, b) = \gcd(|a|, |b|)$
  - $\gcd(a, 0) = |a|$
  - $\gcd(a, b) = \gcd(b, a)$
  - $\gcd(a, b) = \gcd(a \pm b, b)$
    - $d|a \text{ and } d|b \Leftrightarrow d|(a+b) \text{ and } d|b$
    - 이므로 (a, b)의 공약수 집합과 (a+b, b)의 공약수 집합 동일함
    - a-b도 동일하게 증명 가능
  - $\gcd(a, b) = \gcd(a + nb, b)$
    - 위의 결과에서 수학적 귀납법 적용
  - $\gcd(a, b) = \gcd(a \bmod b, b)$
    - $a \bmod b = r$ 이라고 하면 $a = nb + r$ 인 정수 n 존재
    - a에서 b를 여러 번 뺀다고 생각해도 됨

유클리드 호제법

- 유클리드 호제법
  - 두 정수 a, b의 최대공약수를 구하는 과정
    - 음수인 경우 절댓값을 취하면 되므로 $a, b \geq 0$ 인 경우만 생각
    - $\gcd(a, b) = \gcd(b, a)$ 이므로 $a \geq b$ 인 경우만 생각
    - $\gcd(a, 0) = a$ 이므로 $a \geq b \geq 1$ 인 경우만 생각, $b = 0$ 이면 알고리즘 종료
  - $\gcd(a, b) = \gcd(a\ mod\ b, b)$
    - $a = bq + r$ 이라고 하면 $\gcd(a, b) = \gcd(b, r)$, $a \geq b > r$
    - (a, b)를 (b, r)로 축소
    - 이대로 b를 0까지 끌고 내려가면 됨
      - $r \leq a/2$ 이므로 $br \leq ab/2$
        - $q \geq 1$이므로 $2r \leq (q + 1)r = qr + r \leq qb + r = a$
      - $O(\log ab)$번의 축소를 거치면 $b = 0$

유클리드 호제법

- BOJ 2609 최대공약수와 최소공배수
  - 두 자연수의 최대공약수와 최소공배수를 출력하는 문제
  - lcm(a, b) = a * b / gcd(a, b)
    - 계산 과정에서 나올 수 있는 최댓값은 ab
  - lcm(a, b) = a / gcd(a, b) * b
    - 계산 과정에서 나올 수 있는 최댓값은 lcm ≤ ab

```
int gcd(int a, int b){
    return b ? gcd(b, a % b) : a;
}
int lcm(int a, int b){
    return a / gcd(a, b) * b;
}
```

질문?

과제

- 필수
  - 17466 N! mod P (1)
  - 1629 곱셈
  - 1978 소수 찾기
  - 15965 K번째 소수
  - 16563 어려운 소인수분해
  - 2609 최대공약수와 최소공배수
- 심화
  - 1990 소수인 팰린드롬
  - 2824 최대공약수
  - 11690 LCM(1, 2, ..., n)