PULSE 2023 Fall

2주차 – 수학

목차

- PS 시 주의할 점
  - ❖자주 하는 실수-1
- 수학
  - ❖나머지 정리
  - ❖소수
  - ❖에라토스테네스의 체
  - ❖LCM/GCD

PS시 주의할 점

자주 하는 실수-1

PULSE
부산대학교 알고리즘 동아리

- 테스트 케이스가 여러 개인 문제에서 **초기화를 제대로 수행하지 않는** 실수
  - 예) 1부터 n까지의 합을 구하는 문제

```c
#include <stdio.h>
int main() {
  int n, sum = 0;
  int t;
  scanf("%d", &t);
  while (t--) {
    scanf("%d", &n);
    for (int i = 1; i <= n; ++i) {
      sum += i;
    }
    printf("%d\n", sum);
  }
}
```

→

```c
#include <stdio.h>
int main() {
  int t;
  scanf("%d", &t);
  while (t--) {
    int n, sum = 0;
    scanf("%d", &n);
    for (int i = 1; i <= n; ++i) {
      sum += i;
    }
    printf("%d\n", sum);
  }
}
```

- 변수의 스코프를 항상 고려할 것!

자주 하는 실수-1

- 문자열에서 마지막 **널문자를 고려하지 않고 배열 크기를 잡는** 실수

h	e	l	l	o		w	o	r	l	d	₩0	?	?	?

- 예) 입력으로 들어오는 문자열의 길이가 최대 50이라고 할 때

```
char str[50];
```

→

```
char str[51];
```

자주 하는 실수-1

- 변수 **<u>범위 오버플로우를</u>** 고려하지 않는 실수

C++ 기준

- int (32bit, -2^31 ~ 2^31-1)
- long long (64bit, -2^63 ~ 2^63-1)

```
int main() {
  int x = 1 << 16 // 2^16;
  int y = 1 << 16 // 2^16;
  int z1 = x * y; //틀린예1 (int는 -2^31 ~ 2^31-1)
  long long z2 = x * y; 틀린예2 // 틀린예2 (계산과정에서 오버플로우)
  long long z3 = (long long)x * y; // 옳은 예
}
```

자주 하는 실수-1

- 부동소수점 값을 **<u>직접</u>** 비교하는 실수
  - 부동소수점 타입의 두 값을 비교할 때는 두 값의 차이가 충분히 작은 값 이하일 때 같은 것으로 판단해야 함

```
for (double d = 0; d != 0.3; d += 0.1) {
  // infinite loop
  //
}
```

→

```
const double EPS = 1e-9;
for (double d = 0; fabs(d - 0.3) > EPS; d += 0.1) {
  // finite loop
  //
}
```

수학

나눗셈 정리

- 두 정수 m, n이 주어지고, n != 0 일 때, $m = nq + r$이고 $0 \le r \le |n|$ 를 만족하는 정수 q, r이 유일하게 존재한다.
- 여기서 정수 q, r을 각각 m을 n으로 나눈 몫과 나머지라고 한다.
- r이 0인 경우, m,n은 서로 배수 / 약수의 관계에 놓여 있다고 하며, n|m (m은 n의 배수이다)로 표기한다.

소수

- 약수가 1과 자기 자신만 있는 2 이상의 자연수
- 소수 판별
  1. 2부터 n-1까지 나누어 떨어지지 않으면 소수
     - 시간 복잡도: O(n)
  2. $\sqrt{n}$ 까지만 확인해도 소수인지 판별 가능함
     - $n = pq \ (p \le q)$ 일 때, $\mathrm{p} \le \sqrt{n}$ 만족
     - $\mathrm{p}, \mathrm{q} > \sqrt{n}$ 이면, $n < pq$
     - 시간 복잡도: O($\sqrt{n}$)

소인수 분해

- 양수 n의 소인수를 모두 출력하는 것
- n은 $\sqrt{n}$ 보다 큰 소인수를 중복 포함하여 1개만 가질 수 있음
  - ➢ $\sqrt{n}$ 이하의 소수로 모두 나눠보면 됨
  - 마지막 라인에 n != 1이면 n도 소인수
  - 시간 복잡도: O($\sqrt{n}$)

```
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 << i << "₩n";
}
```


https://rkm0959.tistory.com/category/PS/PS%20%EC%A0%95%EC%88%98%EB%A1%A0%20%EA%B0%80%EC%9D%B4%EB%93%9C?page=2

에라토스테네스의 체

- 1부터 n까지의 소수를 모두 구하기 위해 도입
- 1부터 진행하며, 소수 발견 시 그 배수에 해당 하는 모든 수 제외
  - 예) 2는 소수이므로, 2를 제외한 2의 배수는 모두 소수가 아니다.

- 시간 복잡도
  - n 이하의 소수 p에 대하여, n보다 작거나 같은 p의 배수를 제거함
  - Mertens' second theorem에 의해 O(n log log n)으로 알려져 있음.

https://rkm0959.tistory.com/category/PS/PS%20%EC%A0%95%EC%88%98%EB%A1%A0%20%EA%B0%80%EC%9D%B4%EB%93%9C?page=2

최소공배수 / 최대공약수

- 최소공배수(LCM, Least Common Multiple)
  - 두 수의 곱에서 최대공약수를 나누면 얻을 수 있음

- 최대공약수(GCD, Greatest Common Division)
  - 두 수의 공통된 소인수를 모두 곱하면 최대 공약수를 모두 곱하면 얻을 수 있음

- A, B의 최대공약수가 G, 최소공배수가 L이면, 다음과 같은 등식 성립

$$AB = LG$$

https://rkm0959.tistory.com/category/PS/PS%20%EC%A0%95%EC%88%98%EB%A1%A0%20%EA%B0%80%EC%9D%B4%EB%93%9C?page=2

최대공약수의 성질

PULSE
부산대학교 알고리즘 동아리

- $\gcd(a, b) = \gcd(|a|, |b|)$
  - 음수는 절댓값으로 처리
- $\gcd(a, 0) = |a|$
- $\gcd(a, b) = \gcd(b, a)$
  - 변환해도 결과는 같음
- $\gcd(a, b) = \gcd(a \pm b,\ b)$
  - ➢ $\gcd(a, b) = \gcd(a + nb,\ b)$
  - ➢ $\gcd(a, b) = \gcd(a \bmod b,\ b)$

유클리드 호제법

- 두 정수 m, n의 최대공약수를 구하는 과정
- $m, n \geq 0, m \geq n$으로 변형하여 진행 가능
- $\gcd(m, 0) = |m|$이므로 n이 0이면 종료
- $\gcd(m, n) = \gcd(m\ mod\ n,\ n)$
- 과정을 반복하여 n이 0이 될 때까지 반복

https://rkm0959.tistory.com/category/PS/PS%20%EC%A0%95%EC%88%98%EB%A1%A0%20%EA%B0%80%EC%9D%B4%EB%93%9C?page=2

참고자료

- PS/PS 정수론 가이드
  - https://rkm0959.tistory.com/177