수학 1

최백준 choi@startlink.io

나머지 연산

나머지 연산

Modular Arithmetic

- 컴퓨터의 정수는 저장할 수 있는 범위가 저장되어 있기 때문에, 답을 M으로 나눈 나머지를 출력하라는 문제가 등장한다.
- (A+B) mod M = ((A mod M) + (B mod M)) mod M
- (A×B) mod M = ((A mod M) × (B mod M)) mod M
- 나누기의 경우에는 성립하지 않는다. (Modular Inverse를 구해야 함)
- 뺄셈의 경우에는 먼저 mod 연산을 한 결과가 음수가 나올 수 있기 때문에 다음과 같이 해야 한다.
- (A-B) mod M = ((A mod M) – (B mod M) + M) mod M

나머지

https://www.acmicpc.net/problem/10430

- 첫째 줄에 (A+B)%C
- 둘째 줄에 (A%C + B%C)%C
- 셋째 줄에 (A×B)%C
- 넷째 줄에 (A%C × B%C)%C
- 를 출력하는 문제

나머지

https://www.acmicpc.net/problem/10430

- C++: https://gist.github.com/Baekjoon/c3dce55574f250368684

최대공약수

최대공약수

Greatest Common Divisor

- 최대공약수는 줄여서 GCD라고 쓴다.
- 두 수 A와 B의 최대공약수 G는 A와 B의 공통된 약수 중에서 가장 큰 정수이다.
- 최대공약수를 구하는 가장 쉬운 방법은 2부터 min(A, B)까지 모든 정수로 나누어 보는 방법
- 최대공약수가 1인 두 수를 서로소(Coprime)라고 한다.

```
int g = 1;
for (int i=2; i<=min(a,b); i++) {
    if (a % i == 0 && b % i == 0) {
        g = i;
    }
}
```

최대공약수

Greatest Common Divisor

- 앞 페이지에 있는 방법보다 빠른 방법이 있다.
- 유클리드 호제법(Euclidean algorithm)을 이용하는 방법이다.
- a를 b로 나눈 나머지를 r이라고 했을 때
- GCD(a, b) = GCD(b, r) 과 같다
- r이 0이면 그 때 b가 최대 공약수이다.
- GCD(24, 16) = GCD(16, 8) = GCD(8, 0) = 8

최대공약수

Greatest Common Divisor

- 재귀함수를 사용하지 않고 구현한 유클리드 호제법

```
int gcd(int x, int y) {
    while (y != 0) {
        int r = x%y;
        x = y;
        y = r;
    }
    return x;
}
```

최대공약수

Greatest Common Divisor

- 재귀함수를 사용해서 구현한 유클리드 호제법

```
int gcd(int x, int y) {
    if (y == 0) {
        return x;
    } else {
        return gcd(y, x%y);
    }
}
```

최대공약수

Greatest Common Divisor

- 세 수의 최대공약수는 다음과 같이 구할 수 있다.
- GCD(a, b, c) = GCD(GCD(a, b), c)
- 네 수, N개의 숫자도 위와 같은 식으로 계속해서 구할 수 있다.

최소공배수

Least Common Multiple

- 최소공배수는 줄여서 LCM이라고 한다.
- 두 수의 최소공배수는 두 수의 공통된 배수 중에서 가장 작은 정수
- 최소공배수는 GCD를 응용해서 구할 수 있다.
- 두 수 a, b의 최대공약수를 g라고 했을 때
- 최소공배수 l = g * (a/g) * (b/g) 이다.

최대공약수와 최소공배수

https://www.acmicpc.net/problem/2609

- 두 수의 최대공약수와 최소공배수를 구하는 문제

최대공약수와 최소공배수

https://www.acmicpc.net/problem/2609

- C++: https://gist.github.com/Baekjoon/daa4d9aa266f7401fdcd

최소공배수

https://www.acmicpc.net/problem/1934

- 두 수의 최소공배수를 구하는 문제

최소공배수

https://www.acmicpc.net/problem/1934

- C++: https://gist.github.com/Baekjoon/7d627f13b93fe6eb9037

최대공약수

https://www.acmicpc.net/problem/1850

- 모든 자리가 1로만 이루어져 있는 두 자연수 A, B의 최대공약수를 구하는 문제

최대공약수

https://www.acmicpc.net/problem/1850

- C++: https://gist.github.com/Baekjoon/86bde9c6f3560a392cb5

GCD 합

https://www.acmicpc.net/problem/9613

- 수 n개가 주어졌을 때, 가능한 모든 쌍의 GCD의 합을 구하는 문제

GCD 합

https://www.acmicpc.net/problem/9613

- C/C++: https://gist.github.com/Baekjoon/050e76a2bf8f0e2a9e0d

진법 변환

진법 변환

Base Conversion

- 10진법 수 N을 B진법으로 바꾸려면 N이 0이 될때 까지 나머지를 계속해서 구하면 된다.
- 11을 3진법을 바꾸는 방법
- 11/3 = 3 ··· 2
- 3/3 = 1 ... 0
- 1 / 3 = 0 ... 1
- 11은 3진법으로 102 이다.

진법 변환 2

https://www.acmicpc.net/problem/11005

- 10진법 수 N을 B진법으로 바꿔 출력하는 문제

진법 변환 2

https://www.acmicpc.net/problem/11005

- C/C++: https://gist.github.com/Baekjoon/c3d4eb33fdd3c7c71391

진법 변환

Base Conversion

- B진법 수 N을 10진수로 바꾸려면 B^k을 곱하면서 더해가면 된다.
- 3진법 수 102 = 1 * 3^2 + 0 * 3^1 + 2 * 3^0 = 11

진법 변환

https://www.acmicpc.net/problem/2745

- B진법 수 N을 10진법으로 바꾸는 문제

진법 변환

https://www.acmicpc.net/problem/2745

- C++: https://gist.github.com/Baekjoon/b55cef1dcd69f5406a85

2진수 8진수

https://www.acmicpc.net/problem/1373

- 2진수를 8진수로 바꾸는 문제

2진수 8진수

https://www.acmicpc.net/problem/1373

- C++: https://gist.github.com/Baekjoon/e5df08378a2b88b402d9

8진수 2진수

https://www.acmicpc.net/problem/1212

- 8진수를 2진수로 바꾸는 문제

8진수 2진수

https://www.acmicpc.net/problem/1212

- C++: https://gist.github.com/Baekjoon/ec205c33d6fdc6f53b83

-2진수

https://www.acmicpc.net/problem/2089

- N을 -2진수로 바꾸는 문제

-2진수

https://www.acmicpc.net/problem/2089

- 일반적인 진법 변환과 똑같이 변환을 하면 된다.
- 이 때, 나머지가 음수가 나오면 안된다는 점을 조심해서 코딩해야 한다.
- 총 2가지 경우로 나눠볼 수가 있다.
- 양수/-2
- 음수/-2
- 각각의 경우에서 양수가 2로 나누어 떨어지는 경우와
- 음수가 2로 나누어 떨어지는 경우로 나눌 수가 있다

-2진수

https://www.acmicpc.net/problem/2089

- 예시
- 6/2 = 3 ⋯ 0
- 7/2 = 3 ⋯ 1
- -6/2 = -3 ⋯ 0
- -7/2 = -4 ⋯ 1
- 음수 나눗셈의 경우를 조심하면서 구현해야 한다

-2진수

https://www.acmicpc.net/problem/2089

- C/C++: https://gist.github.com/Baekjoon/502537c4ecb459d75311

진법 변환

Base Conversion

- A진법을 B진법으로 바꾸려면
- A진법 -> 10진법 -> B진법
- 의 과정을 거치면 된다.

Base Conversion

https://www.acmicpc.net/problem/11576

- A진법 수를 B진법으로 바꾸는 문제

Base Conversion

https://www.acmicpc.net/problem/11576

- C++: https://gist.github.com/Baekjoon/194d80655163cdb2a726

소수

소수

Prime Number

- 소수: 약수가 1과 자기 자신 밖에 없는 수
- N이 소수가 되려면, 2보다 크거나 같고, N-1보다 작거나 같은 자연수로 나누어 떨어지면 안된다.
- 1부터 100까지 소수
- 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97

소수

Prime Number


```
bool prime(int x) {
    if (x < 2) {
        return false;
    }
    for (int i=2; i<=x-1; i++) {
        if (x % i == 0) {
            return false;
        }
    }
    return true;
}
```

소수

Prime Number

- 소수: 약수가 1과 자기 자신 밖에 없는 수
- N이 소수가 되려면, 2보다 크거나 같고, N/2보다 작거나 같은 자연수로 나누어 떨어지면 안된다.
- 이유: N의 약수 중에서 가장 큰 것은 N/2보다 작거나 같기 때문
- N = a × b로 나타낼 수 있는데, a가 작을수록 b는 크다.
- 가능한 a중에서 가장 작은 값은 2이기 때문에, b는 N/2를 넘지 않는다.

소수

Prime Number

```
bool prime(int x) {
    if (x < 2) {
        return false;
    }
    for (int i=2; i<=x/2; i++) {
        if (x % i == 0) {
            return false;
        }
    }
    return true;
}
```

소수

Prime Number

- 소수: 약수가 1과 자기 자신 밖에 없는 수
- N이 소수가 되려면, 2보다 크거나 같고, 루트N 보다 작거나 같은 자연수로 나누어 떨어지면 안된다.
- 이유: N이 소수가 아니라면, N = a × b로 나타낼 수 있다. ($a \le b$)
- a > b라면 두 수를 바꿔서 항상 $a \le b$로 만들 수 있다.
- 두 수 a와 b의 차이가 가장 작은 경우는 루트 N이다.
- 따라서, 루트 N까지만 검사를 해보면 된다.

소수

Prime Number


```
bool prime(int x) {
    if (x < 2) {
        return false;
    }
    for (int i=2; i*i<=x; i++) {
        if (x % i == 0) {
            return false;
        }
    }
    return true;
}
```

소수

Prime Number

- 컴퓨터에서 실수는 근사값을 나타내기 때문에, 루트 N과 같은 경우는 앞 페이지 처럼 나타내는 것이 좋다.
- 루트 i ≤ N은
- i ≤ N*N 과 같다.
- 어떤 수 N이 소수인지 아닌지 판별하는데 걸리는 시간 복잡도: O(루트N)

소수 찾기

https://www.acmicpc.net/problem/1978

- 입력으로 주어지는 N개의 소수 중에서 소수가 몇 개 인지 구하는 문제

소수 찾기

https://www.acmicpc.net/problem/1978

- C++: https://gist.github.com/Baekjoon/3597219897f6706f9bfb

소수

Prime Number

- 어떤 수 N이 소수인지 아닌지 알아내는데 걸리는 시간 복잡도는 O(루트N) 이었다.
- N = 백만인 경우: 루트N = 1,000
- N = 1억인 경우: 루트 N = 10,000
- 그럼, 1부터 1,000,000까지 모든 소수를 구하는데 걸리는 시간 복잡도는 몇일까?
- 각각의 수에 대해서 소수인지 아닌지 검사해야 한다.
- 각각의 수에 대해서 O(루트N)의 시간이 걸린다.
- 수는 총 N개이기 때문에, O(N루트N)이 걸린다.
- 1,000,000 * 1,000 = 1,000,000,000 = 10억 = 10초
- 너무 긴 시간이 필요하다.

에라토스테네스의 체

Sieve of Eratosthenes

- 1부터 N까지 범위 안에 들어가는 모든 소수를 구하려면 에라토스테네스의 체를 사용한다.

1. 2부터 N까지 모든 수를 써놓는다.
2. 아직 지워지지 않은 수 중에서 가장 작은 수를 찾는다.
3. 그 수를 지우고 소수로 저장한다.
4. 이제 그 수의 배수를 모두 지운다.

에라토스테네스의 체

Sieve of Eratosthenes

- 지워지지 않은 수 중에서 가장 작은 수는 2이다.
- 2는 소수이고 2의 배수를 모두 지운다.

	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
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

에라토스테네스의 체

Sieve of Eratosthenes

- 지워지지 않은 수 중에서 가장 작은 수는 2이다.
- 2는 소수이고 2의 배수를 모두 지운다.

	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
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

에라토스테네스의 체

Sieve of Eratosthenes

- 3의 배수를 지운다.

	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
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

에라토스테네스의 체

Sieve of Eratosthenes

- 5의 배수를 지운다.

	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
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

에라토스테네스의 체

Sieve of Eratosthenes

- 7의 배수를 지운다.

	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
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

에라토스테네스의 체

Sieve of Eratosthenes

- 11의 배수는 이미 지워져 있다.
- 2, 3, 5, 7로 인해서
- 11×11은 121로 100을 넘기 때문에
- 더 이상 수행할 필요가 없다.
- 남아있는 모든 수가 소수이다.

	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
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

에라토스테네스의 체

Sieve of Eratosthenes

```
int p[100]; // 소수 저장
int pn=0; // 소수의 개수
bool c[101]; // 지워졌으면 true
int n = 100; // 100까지 소수
for (int i=2; i<=n; i++) {
    if (c[i] == false) {
        p[pn++] = i;
        for (int j = i*i; j<=n; j+=i) {
            c[j] = true;
        }
    }
}
```

에라토스테네스의 체

Sieve of Eratosthenes

- 1부터 N까지 모든 소수를 구하는 것이 목표이기 때문에, 구현할 때는 바깥 for문 (i)를 N까지 돌린다.
- 안쪽 for문 (j)는 N의 크기에 따라서, i*i 또는 i*2로 바꾸는 것이 좋다.
- i = 백만인 경우 i*i는 범위를 넘어가기 때문

소수 구하기

https://www.acmicpc.net/problem/1929

- M이상 N이하 소수를 모두 출력하는 문제

소수 구하기

https://www.acmicpc.net/problem/1929

- C++: https://gist.github.com/Baekjoon/3247d67f7eb841d04e40

골드바흐의 추측

Goldbach's conjecture

- 2보다 큰 모든 짝수는 두 소수의 합으로 표현 가능하다.
- 위의 문장에 3을 더하면
- 5보다 큰 모든 홀수는 세 소수의 합으로 표현 가능하다.
- 로 바뀐다.
- 아직 증명되지 않은 문제
- 10^{18} 이하에서는 참인 것이 증명되어 있다.

골드바흐의 추측

https://www.acmicpc.net/problem/6588

- 백만 이하의 짝수에 대해서 골드 바흐의 추측을 검증하는 문제

골드바흐의 추측

https://www.acmicpc.net/problem/6588

- C++: https://gist.github.com/Baekjoon/86c79ab80265f6ca440d

소인수분해

소인수분해

Prime Factorization

- 정수 N을 소수의 곱으로 분해
- 소수를 구하지 않고도 해결할 수 있다.
- N을 소인수분해 했을 때, 나타날 수 있는 인수 중에서 가장 큰 값은 루트N이다.
- 따라서, 2부터 루트 N까지 for문을 돌면서
- N을 나눌 수 있으면, 나눌 수 없을 때 까지 계속해서 나누면 된다.

소인수분해

Prime Factorization

```c
for (int i=2; i*i <= n; i++) {
    while (n%i == 0) {
        printf("%d\n",i);
        n /= i;
    }
}
if (n > 1) {
    printf("%d\n",n);
}
```

소인수분해

https://www.acmicpc.net/problem/11653

- 정수 N을 소인수분해해서 출력하는 문제

소인수분해

https://www.acmicpc.net/problem/11653

- C++: https://gist.github.com/Baekjoon/8aa1fc90d7a50d79445f

팩토리얼

팩토리얼

Factorial

- $N! = 1 \times 2 \times \cdots \times N$
- 팩토리얼은 매우 큰 값
- $6! = 720$
- $8! = 40320$
- $10! = 3628800$

팩토리얼

https://www.acmicpc.net/problem/10872

- N!을 출력하는 문제

팩토리얼

https://www.acmicpc.net/problem/10872

- C++: https://gist.github.com/Baekjoon/cf577a5205f82b741575

팩토리얼 0의 개수

https://www.acmicpc.net/problem/1676

- $N! = 1 \times 2 \times \cdots \times N$
- 의 0이 몇 개 인지 알아내는 문제
- 10! = 36288**00**
- 10!이 0이 2개인 이유는 10!을 소인수분해 해보면 알 수 있다.
- $10! = 1 \times 2 \times 3 \times 4 \times 5 \times 6 \times 7 \times 8 \times 9 \times 10$
- $10! = 1 \times 2 \times 3 \times 2^2 \times 5 \times 2 \times 3 \times 7 \times 2^3 \times 3^2 \times 2 \times 5$
- $10! = 2^8 \times 3^4 \times 5^2 \times 7$
- $10! = 2^6 \times 3^4 \times 7 \times (2^2 \times 5^2) = 2^6 \times 3^4 \times 7 \times 10^2$

팩토리얼 0의 개수

https://www.acmicpc.net/problem/1676

- $N! = 1 \times 2 \times \cdots \times N$
- 의 0이 몇 개 인지 알아내려면 N!을 소인수분해 했을 때, 2와 5가 몇 개 나오는지 알아야 한다.
- 5의 개수가 항상 2의 개수 보다 적기 때문에, 5의 개수만 세어주면 된다.
- N! 0의 개수 = $\lfloor N/5 \rfloor + \lfloor N/5^2 \rfloor + \lfloor N/5^3 \rfloor + \cdots$

팩토리얼 0의 개수

https://www.acmicpc.net/problem/1676

- 100!의 경우
- 인수로 5가 들어가는 것을 찾아보자.

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
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

팩토리얼 0의 개수

https://www.acmicpc.net/problem/1676

- 100!의 경우
- 인수로 5가 들어가는 것을 찾아보자.
- 여기서 25, 50, 75, 100은
- 25*1, 25*2, 25*3, 25*4 =
- 5*5*1, 5*5*2, 5*5*3, 5*5*4로
- 5가 두 개씩 들어간다.

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
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

팩토리얼 0의 개수

https://www.acmicpc.net/problem/1676

- 100/5를 했을 때 세는 5의 개수
- = 20개
- 25, 50 ,75, 100도 5의 개수를 1개로 센다
- 따라서 100/25를 한 번 더 해서
- 5의 개수를 한 번 더 세어줘야 한다.

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
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

팩토리얼 0의 개수

https://www.acmicpc.net/problem/1676

- 100/25 = 4
- 25, 50, 75, 100
- 따라서 ,100!의 0의 개수는 20+4 = 24개이다.

93326215443944152681699238856266700490715968264381621468592963895217599993229915608941463976156518286253697920827223758251185210916864000000000000000000000000

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
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

팩토리얼 0의 개수

https://www.acmicpc.net/problem/1676

- C++: https://gist.github.com/Baekjoon/2b1b9ff4f46ef2a1546c

조합 0의 개수

https://www.acmicpc.net/problem/2004

- nCm 의 0의 개수를 구하는 문제

조합 0의 개수

https://www.acmicpc.net/problem/2004

- 팩토리얼은 2의 개수가 5의 개수 보다 항상 많기 때문에, 5의 개수만 세어줬는데
- 조합은 어떻게 될 지 모르기 때문에, 2의 개수와 5의 개수를 동시에 세어줘야 한다.

조합 0의 개수

https://www.acmicpc.net/problem/2004

- C++: https://gist.github.com/Baekjoon/de23cab681709bc96680