

# 완전 탐색 0

최백준 [choi@startlink.io](mailto:choi@startlink.io)

---

# 비트마스크

---

# 비트마스크

Bitmask

- 비트(bit) 연산을 사용해서 부분 집합을 표현할 수 있다.

# 비트 연산

bitwise operation

- $\&$  (and),  $|$  (or),  $\sim$  (not),  $\wedge$  (xor)

A	B	$\sim A$	$A \& B$	$A   B$	$A \wedge B$
0	0	1	0	0	0
0	1	1	0	1	1
1	0	0	0	1	1
1	1	0	1	1	0

# 비트 연산

bitwise operation

- 두 수 A와 B를 비트 연산 하는 경우에는 가장 뒤의 자리부터 하나씩 연산을 수행하면 된다.
- A = 27, B = 83인 경우
- $A = 11011_2, B = 1010011_2$
- $A \& B = 19, A \mid B = 91, A \wedge B = 73$

0	0	1	1	0	1	1		0	0	1	1	0	1	1		0	0	1	1	0	1	1	
&	1	0	1	0	0	1	1		1	0	1	0	0	1	1	^	1	0	1	0	0	1	1
-----								-----								-----							
0	0	1	0	0	1	1		1	0	1	1	0	1	1		1	0	0	1	0	0	0	

# 비트 연산

bitwise operation

6

- not 연산의 경우에는 자료형에 따라 결과가 달라진다.
- $A = 83 = 1010011_2$
- $\sim A = 10101100_2$  (8비트 자료형인 경우)
- $\sim A = 11111111\ 11111111\ 11111111\ 10101100_2$  (32비트 자료형인 경우)
- 또, unsigned, signed에 따라서 보여지는 값은 다르다.

# 비트 연산

bitwise operation

- shift left (<<) 와 shift right (>>) 연산이 있다.
- $A \ll B$  (A를 왼쪽으로 B비트만큼 민다.)
- $1 \ll 0 = 1$
- $1 \ll 1 = 2 \ (10_2)$
- $1 \ll 2 = 4 \ (100_2)$
- $1 \ll 3 = 8 \ (1000_2)$
- $1 \ll 4 = 16 \ (10000_2)$
- $3 \ll 3 = 24 \ (11000_2)$
- $5 \ll 10 = 5120 \ (101000000000000_2)$

$$\begin{aligned} & \times 2^k \\ & A \ll B \\ & A \cdot 2^B \end{aligned}$$

$$\begin{aligned} & A \gg B \\ & A / 2^B \end{aligned}$$

# 비트 연산

bitwise operation

- shift left (<<) 와 shift right (>>) 연산이 있다.
- $A \gg B$  (A를 오른쪽으로 B비트만큼 민다.)
- $1 \gg 0 = 1$
- $1 \gg 1 = 0 \text{ } (0_2)$
- $10 \gg 1 = 5 \text{ } (101_2)$
- $10 \gg 2 = 2 \text{ } (10_2)$
- $10 \gg 3 = 1 \text{ } (1_2)$
- $30 \gg 1 = 15 \text{ } (1111_2)$
- $1024 \gg 10 = 1 \text{ } (1_2)$



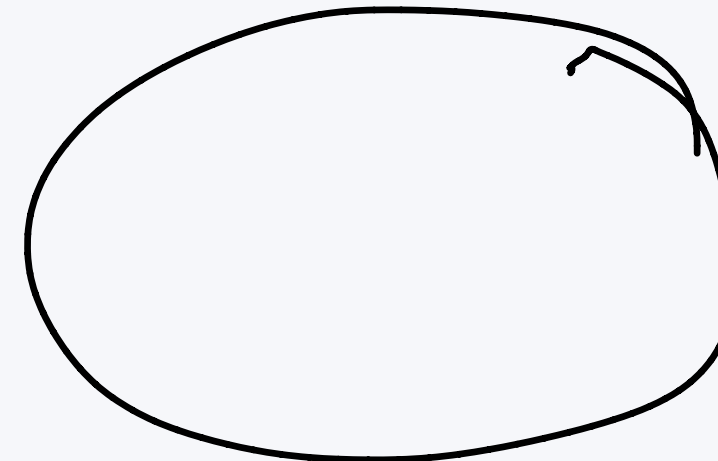
# 비트 연산

bitwise operation

- $A \ll B$ 는  $A \times 2^B$ 와 같다.
- $A \gg B$ 는  $A / 2^B$ 와 같다.
- $(A + B) / 2$ 는  $(A+B) \gg 1$ 로 쓸 수 있다.
- 어떤 수가 홀수 인지 판별하는  $\text{if } (N \% 2 == 1)$  은  $\text{if } (N \& 1)$ 로 줄여 쓸 수 있다.

N이 2의 제곱꼴 또는 0

$$\text{if}(N \& (N-1) == 0)$$

$$\begin{array}{r} N = 10000000 \\ \& N-1 = 01111111 \\ \hline \end{array}$$


# 비트마스크

Bitmask

10

$$570 = 1000111010_2$$

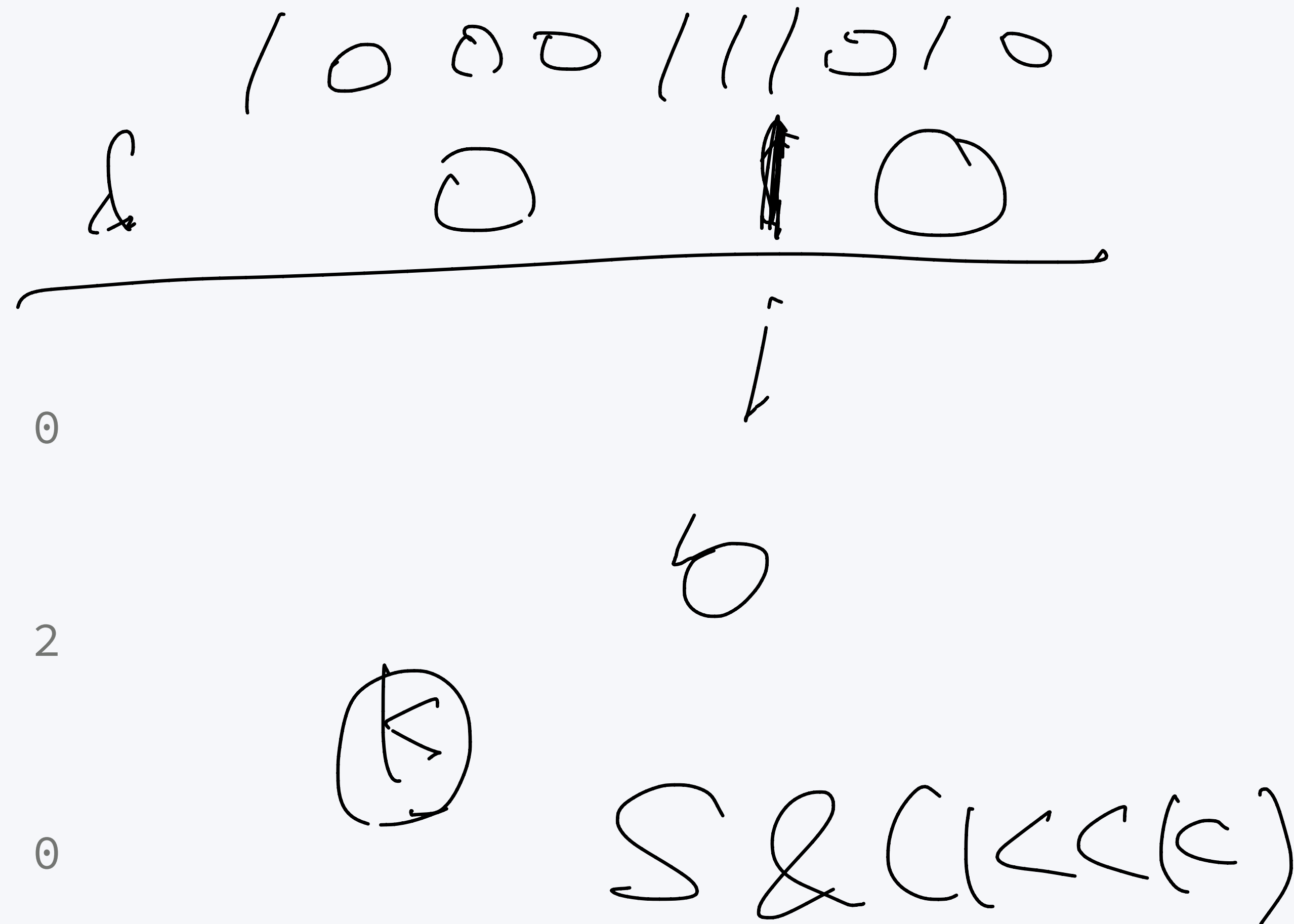
- 정수로 집합을 나타낼 수 있다.

$$\{1, 3, 4, 5, 9\} = 570 = 2^1 + 2^3 + 2^4 + 2^5 + 2^9$$

# 비트마스크

Bitmask

- $\{1, 3, 4, 5, 9\} = 570$
- 0이 포함되어 있는지 검사
  - $570 \ \& \ 2^0 = 570 \ \& \ (1 \ll 0) = 0$
- 1이 포함되어 있는지 검사
  - $570 \ \& \ 2^1 = 570 \ \& \ (1 \ll 1) = 2$
- 2이 포함되어 있는지 검사
  - $570 \ \& \ 2^2 = 570 \ \& \ (1 \ll 2) = 0$
- 3이 포함되어 있는지 검사
  - $570 \ \& \ 2^3 = 570 \ \& \ (1 \ll 3) = 8$



# 비트마스크

Bitmask

- $\{1, 3, 4, 5, 9\} = 570$

- 1 추가하기

$$\bullet 570 \mid 2^1 = 570 \mid (1 \ll 1) = 570 \quad (1000111010_2)$$

- 2 추가하기

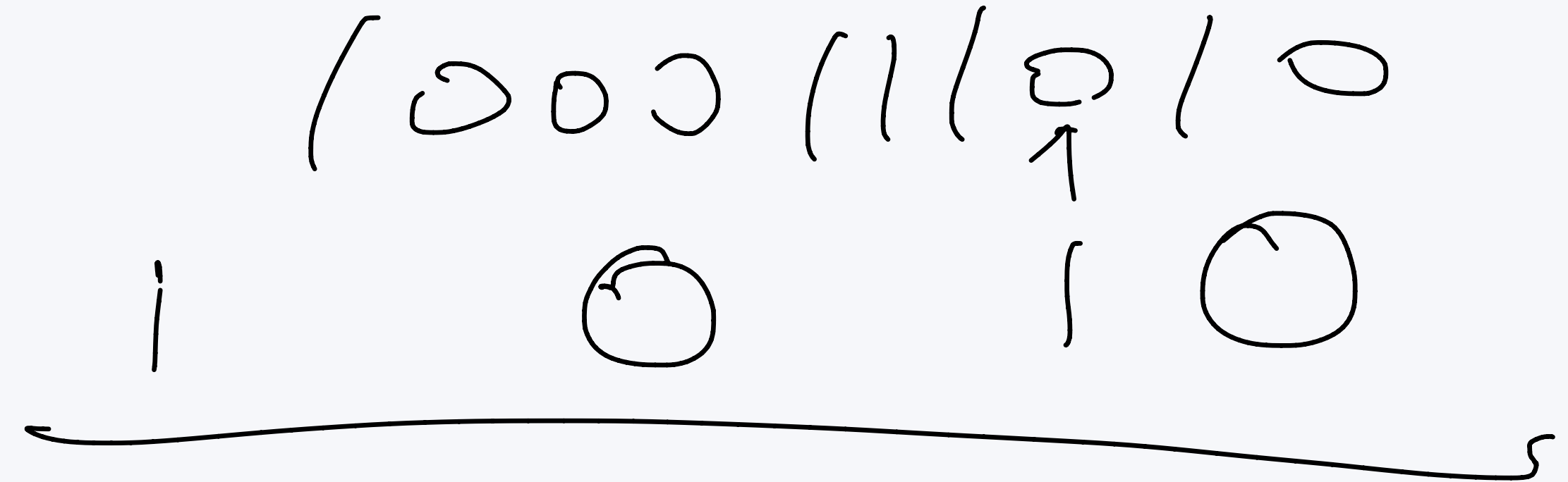
$$\bullet 570 \mid 2^2 = 570 \mid (1 \ll 2) = 574 \quad (1000111110_2)$$

- 3 추가하기

$$\bullet 574 \mid 2^3 = 570 \mid (1 \ll 3) = 570 \quad (1000111010_2)$$

- 4 추가하기

$$\bullet 574 \mid 2^4 = 570 \mid (1 \ll 4) = 570 \quad (1000111010_2)$$



$S \mid (1 \ll k)$

# 비트마스크

Bitmask

- $\{1, 3, 4, 5, 9\} = 570$

- 1 제거하기

- $570 \ \& \sim 2^1 = 570 \ \& \sim (1 \ll 1) = 568 \ (1000111000_2)$

5 & ~(<<1)

- 2 제거하기

- $570 \ \& \sim 2^2 = 570 \ \& \sim (1 \ll 2) = 570 \ (1000111010_2)$

- 3 제거하기

- $562 \ \& \sim 2^3 = 562 \ \& \sim (1 \ll 3) = 562 \ (1000110010_2)$

- 4 제거하기

- $562 \ \& \sim 2^4 = 562 \ \& \sim (1 \ll 4) = 546 \ (1000101010_2)$

5  
 ( 0 0 0 1 1 1 0 1 0 )  
 1 1 1 1 1 0 1 1 1 )

# 비트마스크

Bitmask

- 전체 집합

- $(1 \ll N) - 1$

- 공집합

- 0

(N개)

$$0 \sim N-1$$

14

$$2^N - 1$$

$$(1 \ll N) = 2^N$$

$$1 \ll N - 1$$

$$\textcircled{1} 1 \ll (N-1)$$

$$\textcircled{2} (1 \ll N) - 1$$

# 비트마스크

15

## Bitmask

- 현재 집합이 S일때
- i를 추가
  - $S \mid (1 \ll i)$
- i를 검사
  - $S \& (1 \ll i)$
- i를 제거
  - $S \& \sim(1 \ll i)$
- i를 토글 (0을 1로, 1을 0으로)
  - $S \wedge (1 \ll i)$

# 집합

<https://www.acmicpc.net/problem/11723>

- 비트마스크를 연습해보는 문제




# 집합

<https://www.acmicpc.net/problem/11723>

- C++: <https://gist.github.com/Baekjoon/3503aaa55c03cdde9df51b1bd5155486>
- Java: <https://gist.github.com/Baekjoon/2da9289baa79449207eed6c2013f8c41>

# 비트마스크

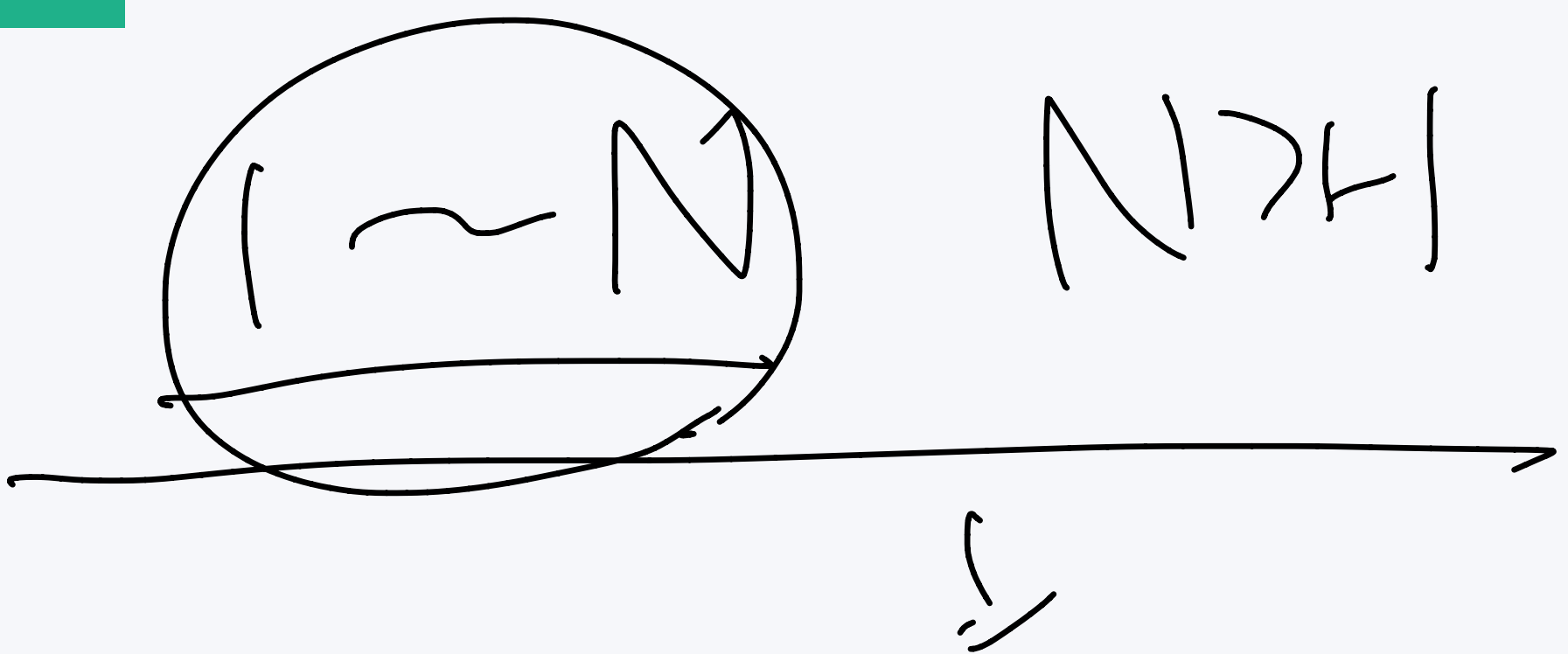
## Bitmask

- 물론 배열을 사용하는 것이 더욱 편리하지만, 비트마스크를 사용하는 이유는
- 집합을 배열의 인덱스로 표현할 수 있기 때문이다.
- 상태  다이نام릭을 할 때 자주 사용하게 된다.

# bitset

bitset

- 비트마스크는 STL의 `bitset`을 이용해서 더 쉽게 나타낼 수 있다.



↓  
H3242 N개의 수

N개의 수

# 순열

# 순열

## Permutation

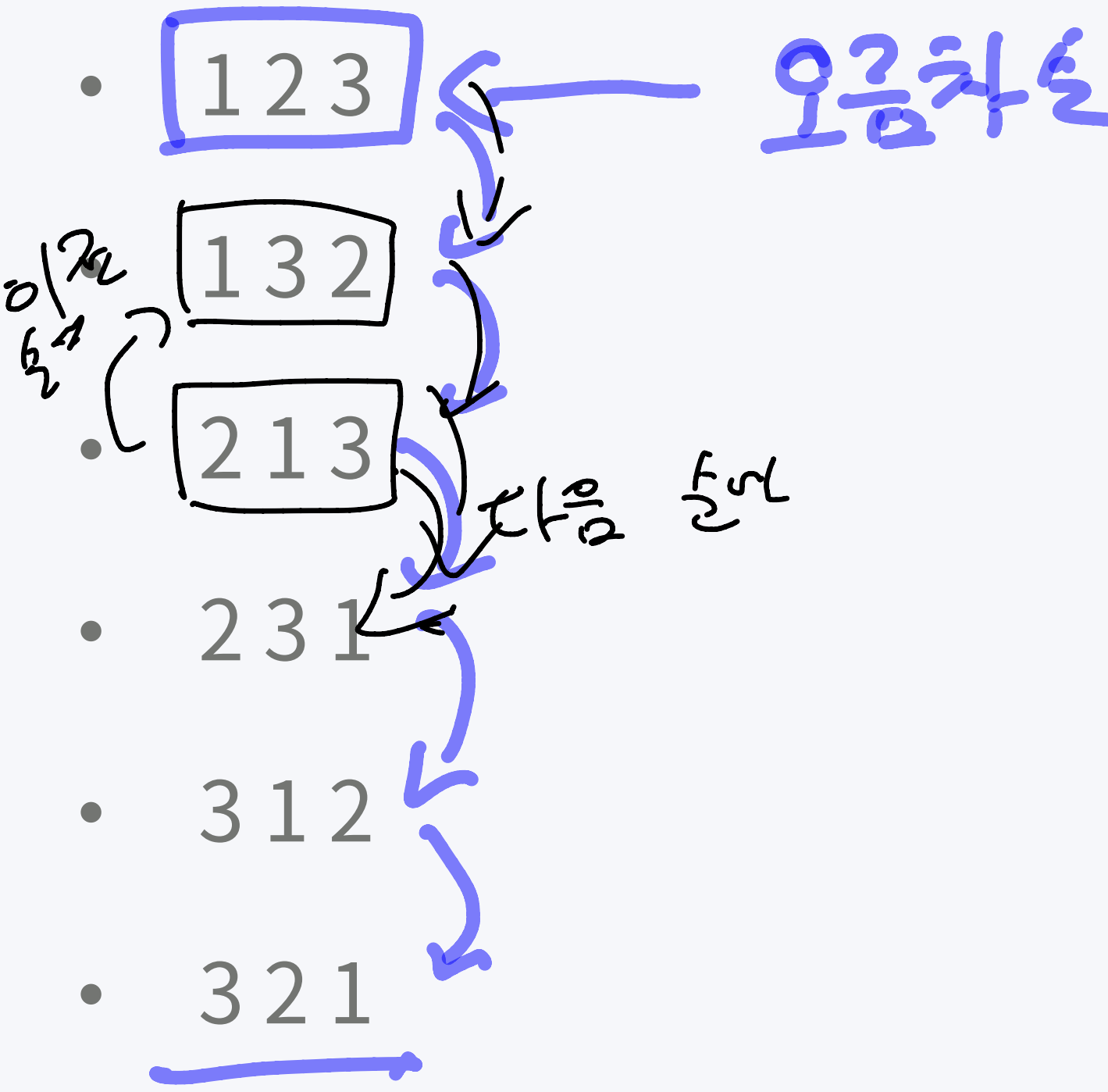
21

- 1 ~ N 까지로 이루어진 수열
- 1 2 3
- 4 1 3 2
- 5 4 2 3 1
- 6 5 1 2 3 4
- 크기는 항상 N이 되어야 하고, 겹치는 숫자가 존재하지 않음

# 순열

Permutation

- 크기가 N인 순열은 총 N!개가 존재한다
- 순열을 사전순으로 나열했을 때
- N = 3인 경우에 사전순은 다음과 같다



다음 순열  $O(N)$   
전체 순열  $O(N \cdot N!)$

5 -2 7  
-2 5 7  
-2 7 5

# 다음 순열

## Next Permutation

- 순열을 사전순으로 나열했을 때, 사전순으로 다음에 오는 순열과 이전에 오는 순열을 찾는 방법
- C++ STL의 algorithm에는 이미 next\_permutation과 prev\_permutation이 존재하기 때문에 사용하면 된다

# 다음 순열

## Next Permutation

1.  $A[i-1] < A[i]$  를 만족하는 가장 큰  $i$ 를 찾는다
2.  $j \geq i$  이면서  $A[j] > A[i-1]$  를 만족하는 가장 큰  $j$ 를 찾는다
3.  $A[i-1]$ 과  $A[j]$ 를 swap 한다
4.  $A[i]$ 부터 순열을 뒤집는다



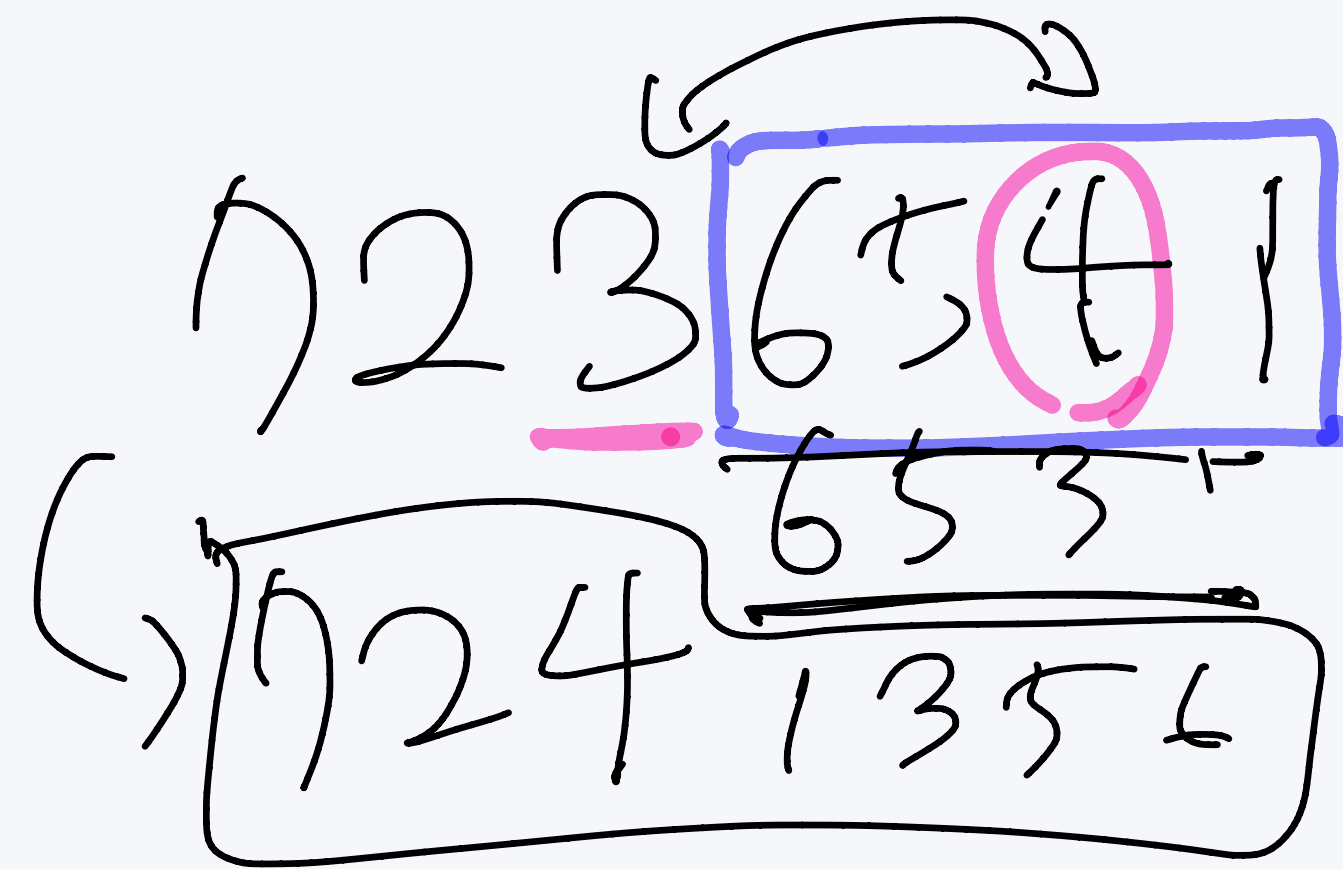
# 다음 순열

Next Permutation

- 순열: 7 2 3 6 5 4 1

- $A[i-1] < A[i]$  를 만족하는 가장 큰  $i$  를 찾는다
- 즉, 순열의 마지막 수에서 끝나는 가장 긴 감소수열을 찾아야 한다

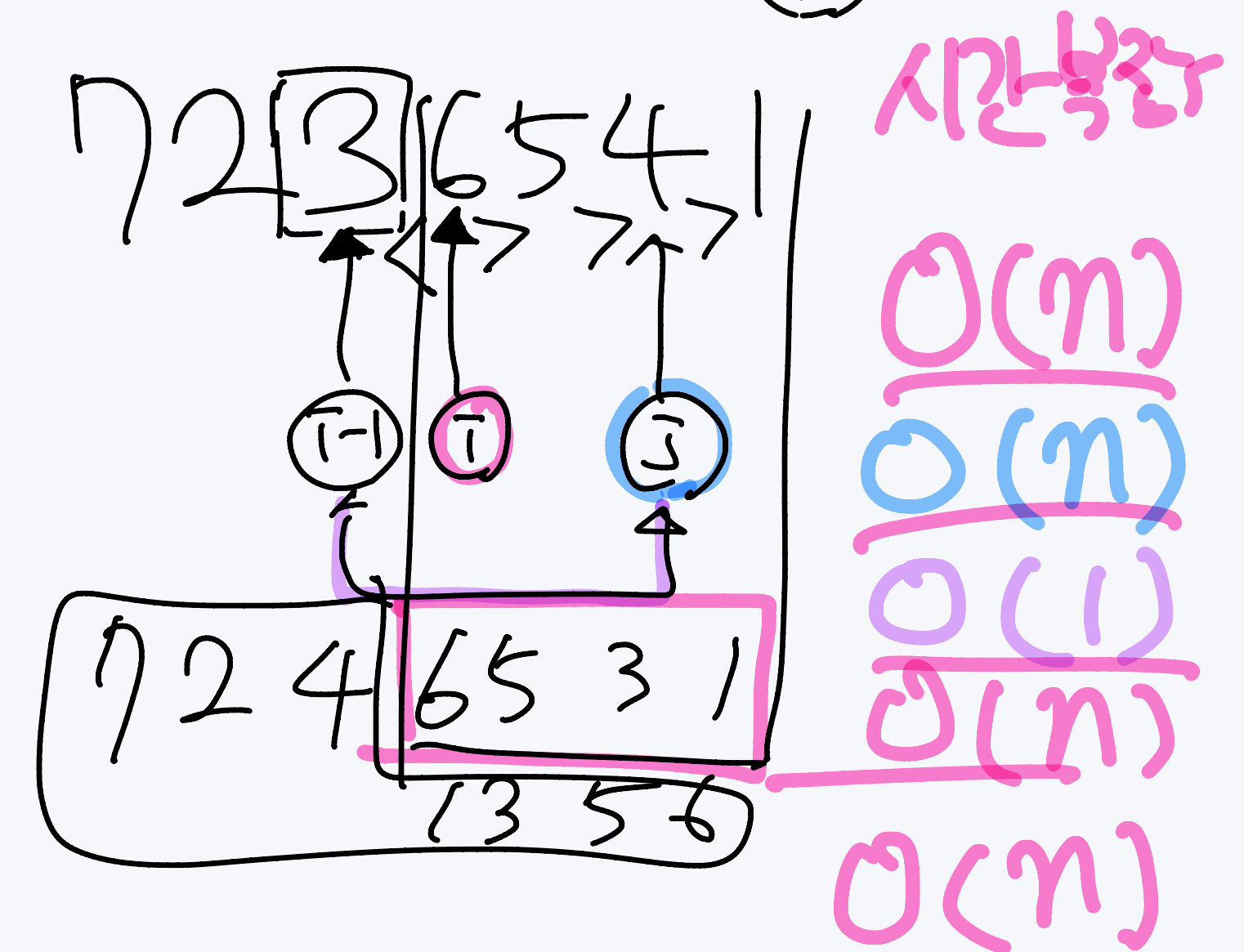
- 순열: 7 2 3 **6 5 4** 1



7 2 3 시작하는  
순열 중 가장 마지막

25

7 2 4 시작 첫  
( > )



# 다음 순열

Next Permutation

26

- 순열: 7 2 3 6 5 4 1
- $j \geq i$  이면서  $A[j] > A[i-1]$  를 만족하는 가장 큰  $j$ 를 찾는다
- 순열: 7 2 3 6 5 4 1

# 다음 순열

Next Permutation

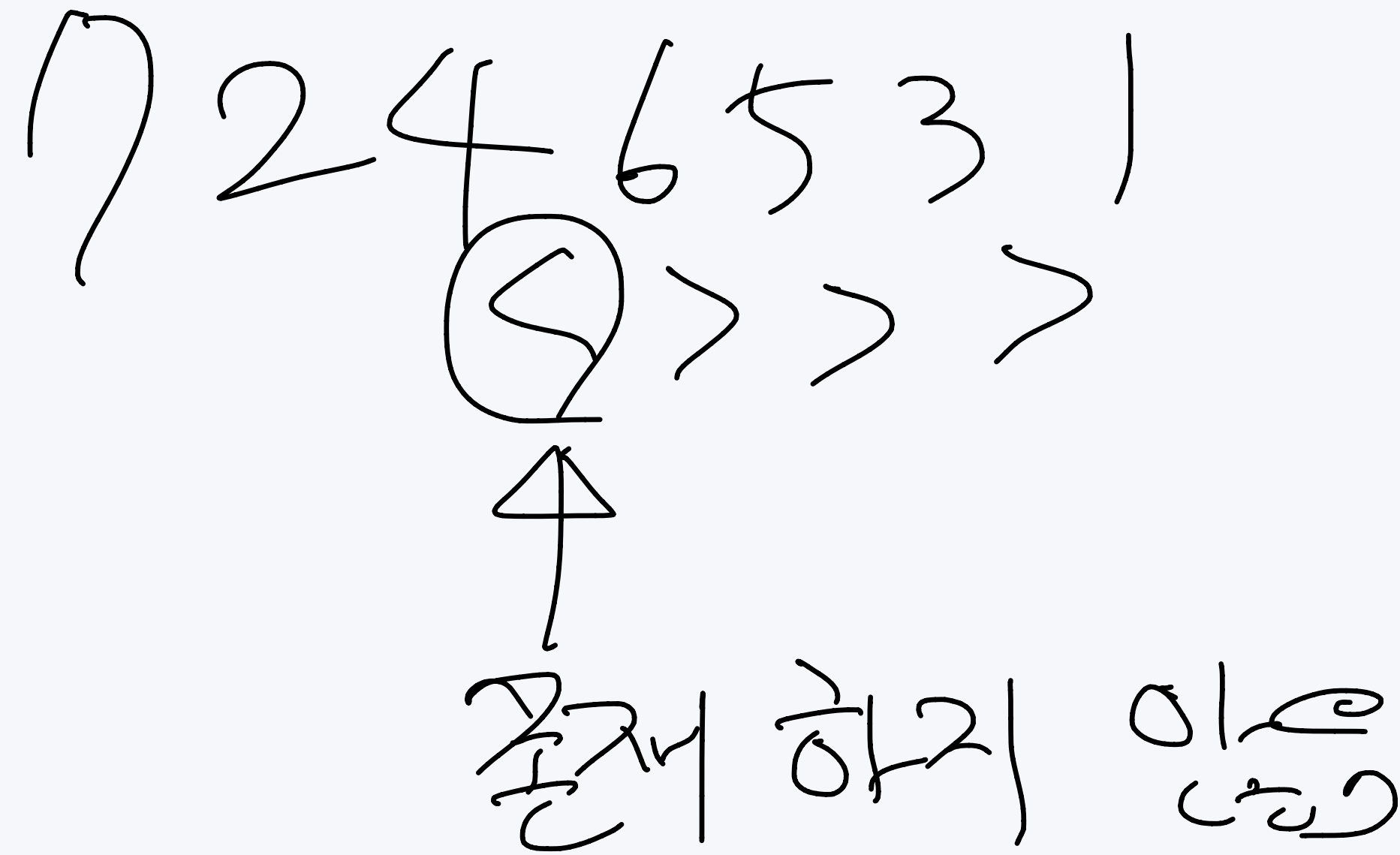
27

- 순열: 7 2 3 6 5 4 1
- $A[i-1]$ 과  $A[j]$ 를 swap 한다
- 순열: 7 2 4 6 5 3 1

# 다음 순열

Next Permutation

- 순열: 7 2 4 6 5 3 1
- A[i]부터 순열을 뒤집는다
- 순열: 7 2 4 1 3 5 6



# 다음 순열

Next Permutation

```
bool next_permutation(int *a, int n) {
```

```
    int i = n-1;
```

```
    while (i > 0 && a[i-1] >= a[i]) i -= 1; ← ①
```

```
    if (i <= 0) return false; // 마지막 순열
```

```
    int j = n-1;
```

```
    while (a[j] <= a[i-1]) j -= 1; ← ②
```

```
    swap(a[i-1], a[j]); ← ③
```

```
    j = n-1;
```

```
    while (i < j) {
```

```
        swap(a[i], a[j]);
```

```
        i += 1; j -= 1;
```

```
    }
```

```
    return true;
```

```
}
```

← ④

# 다음 순열

30

<https://www.acmicpc.net/problem/10972>

- 다음 순열을 구하는 문제

# 다음 순열

<https://www.acmicpc.net/problem/10972>

- C++: <https://gist.github.com/Baekjoon/d51fbc6f75332cfc6ab9>
- C++ (next\_permutation 구현): <https://gist.github.com/Baekjoon/f8d9765ccde7262744b5>
- Java: <https://gist.github.com/Baekjoon/c307fc69373a74a730c0>

# 이전 순열

32

<https://www.acmicpc.net/problem/10973>

- 이전 순열을 구하는 문제



# 이전 순열

<https://www.acmicpc.net/problem/10973>

- C++: <https://gist.github.com/Baekjoon/2db36900d5b1f37b2397>
- C++ (prev\_permutation 구현): <https://gist.github.com/Baekjoon/c3b6e4a24b3841575dc9>
- Java: <https://gist.github.com/Baekjoon/37eda7e437c1d14092aa>

# 모든 순열

34

<https://www.acmicpc.net/problem/10974>

- 모든 순열을 구하는 문제

# 모든 순열

<https://www.acmicpc.net/problem/10974>

- C++: <https://gist.github.com/Baekjoon/8c1c89872b713e45d45d>
- Java: <https://gist.github.com/Baekjoon/bb4679d85dd726fd3456>

# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- 1부터 N까지의 수를 임의로 배열한 순열은 총  $N! = N \times (N-1) \times \cdots \times 2 \times 1$  가지가 있다.
- 임의의 순열은 정렬을 할 수 있다.
- 예를 들어  $N=3$ 인 경우  $\{1, 2, 3\}, \{1, 3, 2\}, \{2, 1, 3\}, \{2, 3, 1\}, \{3, 1, 2\}, \{3, 2, 1\}$ 의 순서로 생각할 수 있다
- 첫 번째 수가 작은 것이 순서상에서 앞서며, 첫 번째 수가 같으면 두 번째 수가 작은 것이, 두 번째 수도 같으면 세 번째 수가 작은 것이...
- N이 주어지면, 아래의 두 소문제 중에 하나를 풀어야 한다.
- k가 주어지면 k번째 순열을 구하고, 임의의 순열이 주어지면 이 순열이 몇 번째 순열인지를 출력하는 프로그램을 작성하시오.

# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- 총 2개의 문제로 이루어져 있다.
- 어떤 순열이 몇 번째 순열인지
- 몇 번째 순열이 뭔지

# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- 어떤 순열이 몇 번째 순열인지 구하는 방법
- $N = 7$ , 순열 = 5, 2, 4, 7, 6, 3, 1
- 1, ?, ?, ?, ?, ?, ? = 6! 개
- 2, ?, ?, ?, ?, ?, ? = 6! 개
- 3, ?, ?, ?, ?, ?, ? = 6! 개
- 4, ?, ?, ?, ?, ?, ? = 6! 개

# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- 어떤 순열이 몇 번째 순열인지 구하는 방법
- $N = 7$ , 순열 = 5, 2, 4, 7, 6, 3, 1
- 5, 1, ?, ?, ?, ?, ? = 5! 개
- 5, 2, 1, ?, ?, ?, ? = 4! 개
- 5, 2, 3, ?, ?, ?, ? = 4! 개
- 5, 2, 4, 1, ?, ?, ? = 3! 개
- 5, 2, 4, 3, ?, ?, ? = 3! 개
- 5, 2, 4, 6, ?, ?, ? = 3! 개

# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- 어떤 순열이 몇 번째 순열인지 구하는 방법
- $N = 7$ , 순열 = 5, 2, 4, 7, 6, 3, 1
- 5, 2, 4, 7, 1, ?, ? = 2! 개
- 5, 2, 4, 7, 3, ?, ? = 2! 개
- 5, 2, 4, 7, 6, 1, ? = 1! 개
- 5, 2, 4, 7, 6, 3, 1
- $6! \times 4 + 5! \times 1 + 4! \times 2 + 3! \times 3 + 2! \times 2 + 1! = 3,071$  번째 순열 (0부터 시작했을 때)



# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- 몇 번째 순열이 어떤 순열인지 찾는 방법
- 3071번째 순열
- $1, ?, ?, ?, ?, ?, ? = 6! = 720$
- $3071 - 720 = 2351$ 번째 순열
- $2, ?, ?, ?, ?, ?, ? = 6! = 720$
- $2351 - 720 = 1631$  번째 순열
- $3, ?, ?, ?, ?, ?, ? = 6! = 720$
- $1631 - 720 = 911$ 번째 순열
- $4, ?, ?, ?, ?, ?, ? = 6! = 720$

# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- 몇 번째 순열이 어떤 순열인지 찾는 방법
- 3071번째 순열
- $911 - 720 = 191$ 번째 순열
- $5, 1, ?, ?, ?, ?, ? = 5! = 120$
- $191 - 120 = 71$ 번째 순열
- $5, 2, 1, ?, ?, ?, ? = 4! = 24$
- $71 - 24 = 47$ 번째 순열
- $5, 2, 3, ?, ?, ?, ? = 4! = 24$
- $47 - 24 = 23$ 번째 순열

# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- 몇 번째 순열이 어떤 순열인지 찾는 방법
- 3071번째 순열
- $47 - 24 = 23$ 번째 순열
- $5, 2, 4, 1, ?, ?, ? = 3! = 6$
- $23 - 6 = 17$ 번째 순열
- $5, 2, 4, 3, ?, ?, ? = 3! = 6$
- $17 - 6 = 11$ 번째 순열
- $5, 2, 4, 6, ?, ?, ? = 3! = 6$
- $11 - 6 = 5$ 번째 순열

# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- 몇 번째 순열이 어떤 순열인지 찾는 방법
- 3071번째 순열
- $11-6 = 5$ 번째 순열
- 5, 2, 4, 7, 1, ?, ? =  $2! = 2$
- $5-2 = 3$ 번째 순열
- 5, 2, 4, 7, 3, ?, ? =  $2! = 2$
- $3-2 = 1$ 번째 순열
- 5, 2, 4, 7, 6, 1, ? =  $1! = 1$
- $1-1 = 0$ 번째 순열

# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- 몇 번째 순열이 어떤 순열인지 찾는 방법
- 3071번째 순열
- $1-1 = 0$ 번째 순열
- 5, 2, 4, 7, 6, 1, 3

# 순열의 순서

<https://www.acmicpc.net/problem/1722>

- C/C++: <https://gist.github.com/Baekjoon/1b94fe874444c2287fe5>
- Java: <https://gist.github.com/Baekjoon/51c4e11e610c81b493a2d0aa7c30c65a>