

ICPC Sinchon



# 2022 Winter Algorithm Camp

## 7. 이분탐색 & 분할정복

서강대학교 임지환

2022 Winter Algorithm Camp

# 목차

1. Introduction
2. 이분탐색
3. 좌표 압축
4. 매개변수 탐색
5. 분할정복
6. 분할정복을 이용한 거듭제곱
7. Appendix

2022 Winter Algorithm Camp

# Introduction – 쪼개다는 것

\* 문제를 쪼개다

문제의 크기가 커짐에 따라 난이도가 어려워진다.

문제를 작은 부분 문제로 나누어 푸는 기법?

⇒ 동적 계획법, 그리디

# Introduction – 컴퓨터공학에서의 2의 의미

- 컴퓨터에서 사용되는 수 체계
  - 이진법
- 대상 전체를 둘로 나누는 논리적 방법 (True / False) - 이분법
- 이 외에도 다양한 자료구조/알고리즘/테크닉에서 둘로 나누는 컨셉이 이용됨

# 이분탐색 - 도입

## \* 이분탐색(Binary Search)

- 정렬된 리스트에서 특정한 값을 찾는 알고리즘  
탐색 구간의 중앙값과의 대소 비교를 통해 다음 탐색구간 설정
- 탐색 연산이 반복적으로 요구될 때 사용

```
1 binary_search(a[], key):  
2     l <- 0  
3     r <- |a| - 1  
4  
5     while l <= r:  
6         mid <- (l + r) / 2  
7  
8         if key == a[mid]:  
9             return True  
10        else if a[mid] < key:  
11            r <- mid - 1  
12        else:  
13            l <- mid + 1  
14  
15    return False
```

# 이분탐색 - 도입

To find 13:

L  
↓

R  
↓

0	1	2	3	4	5	6	7	8
2	3	5	7	11	13	17	19	23

2022 Winter Algorithm Camp

# 이분탐색 - 도입

To find 13:

L				Mid				R
↓				↓				↓
0	1	2	3	4	5	6	7	8
2	3	5	7	11	13	17	19	23

2022 Winter Algorithm Camp

# 이분탐색 - 도입

To find 13:

					L	Mid		R
					↓	↓		↓
0	1	2	3	4	5	6	7	8
2	3	5	7	11	13	17	19	23



2022 Winter Algorithm Camp

# 이분탐색 - 도입

To find 13:



0	1	2	3	4	5	6	7	8
2	3	5	7	11	13	17	19	23

2022 Winter Algorithm Camp

# 이분탐색 - 도입

\* Time Complexity

탐색 구간을 새로 설정할 때마다 길이가 절반씩 감소

최악의 경우 탐색 구간의 길이가  $1 (= 2^0)$ 일 때 탐색 중지

리스트의 크기를  $n = 2^m$  이라 할 때,  $m (= \log_2 n)$ 번의 반복 끝에 존재여부 확인 가능

# 이분탐색 – lower\_bound & upper\_bound

## \* lower\_bound

key 이상의 값이 처음 나오는 위치

ex)  $a = [2, 3, 5, 7, 11, 11]$ ,  $key = 8$ ,

$lower\_bound(a, a + 6, key) - a = 4$

## \* upper\_bound

key 초과 값이 처음 나오는 위치

ex)  $a = [2, 3, 5, 8, 8, 11]$ ,  $key = 8$ ,

$upper\_bound(a, a + 6, key) - a = 5$

```

1 lower_bound(a[], key):
2 // returns index of lb key
3   l <- 0
4   r <- |a| - 1
5   ret <- n
6   while l <= r:
7       m <- (l + r) / 2
8       if arr[m] >= v:
9           r <- m - 1
10          ret <- m
11      else:
12          l <- m + 1
13  return ret

```

```

1 upper_bound(a[], key):
2 // returns index of ub key
3   l <- 0
4   r <- |a| - 1
5   ret <- n
6   while l <= r:
7       m <- (l + r) / 2
8       if arr[m] > v:
9           r <- m - 1
10          ret <- m
11      else:
12          l <- m + 1
13  return ret

```

2022 Winter Algorithm Camp

## 10816. 숫자 카드 2

정수가 적혀 있는 카드가  $N$ 개 ( $|a_i| \leq 10^7, 1 \leq N \leq 500,000$ )

정수가  $M$  ( $1 \leq M \leq 500,000$ )개에 대해, 이 수가 적혀있는 숫자 카드가 몇 개 있는가?

2022 Winter Algorithm Camp

# 10816. 숫자 카드 2

To find 11:



				L					R				
				↓					↓				
2	3	3	7	11	11	11	13	17					

2022 Winter Algorithm Camp

## 10816. 숫자 카드 2

To find 11, with lower\_bound && upper\_bound:

$\text{cnt} = \text{upper\_bound}(a, a + n, 11) - \text{lower\_bound}(a, a + n, 11) = 3$

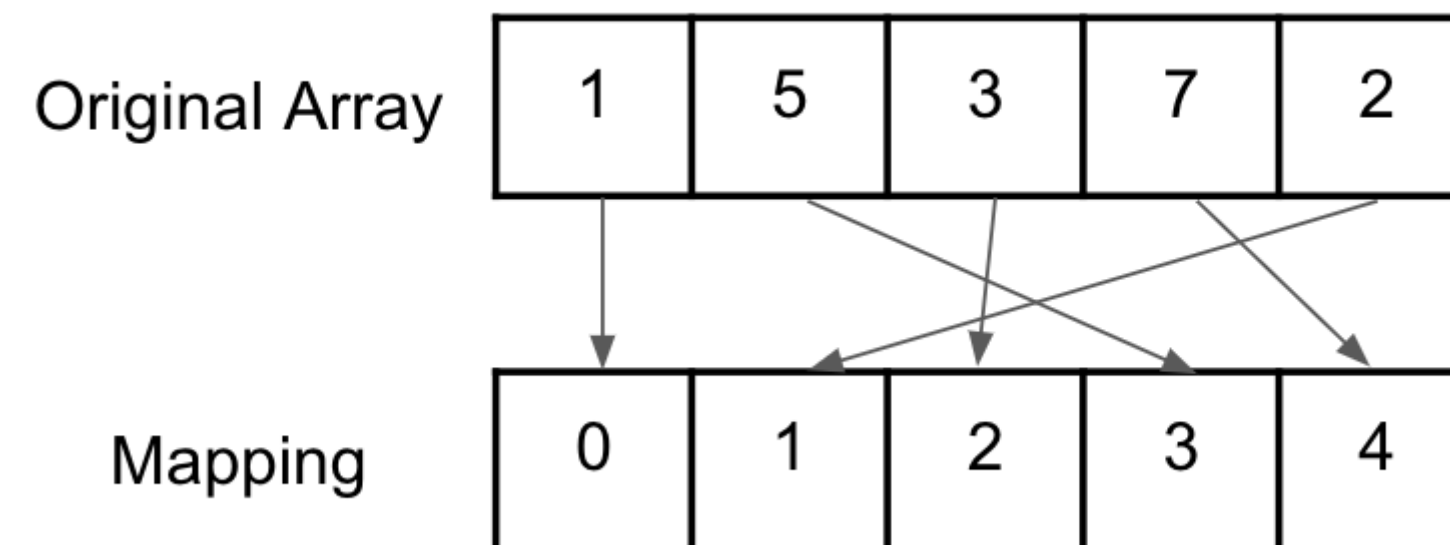
lower_bound				upper_bound				
								
2	3	3	7	11	11	11	13	17

# 좌표압축

## \* 좌표 압축(Coordinate Compression)

간격 또는 중복 정보를 제거하여 많은 점 집합을 더 작은 범위에 mapping하는 기법

- 실제 간격의 정보가 덜 중요할 때
- 중복값이 많고 상대적인 순서만 알아도 될 때
- (좌표 상에서) range는 크지만 실제 점 개수(좌표 정보의 개수)는 적을 때



출처: <https://medium.com/algorithms-digest/coordinate-compression-2fff95326fb>

## 2022 Winter Algorithm Camp

## 좌표압축

labeling with integer

```

1 // Compression with set & map
2
3 int a[mxn];
4 set<int> xlist;
5 map<int, int> xs;
6
7 cin >> N;
8 for (int i = 0; i < N; i++) {
9     cin >> a[i];
10    xlist.insert(a[i]);
11 }
12 int idx = 1;
13 for (int x : xlist) xs[x] = idx++;
14 for (int i = 0; i < N; i++)
15     a[i] = xs[a[i]];

```

맞았습니다!!

10728  
KB

812 ms

```

1 //Compression with vector unique & erase
2
3 int a[mxn];
4 vector<int> xlist;
5
6 cin >> N;
7 for (int i = 0; i < N; i++) {
8     cin >> a[i];
9     xlist.push_back(a[i]);
10 }
11 sort(xlist.begin(), xlist.end());
12 xlist.erase(unique(xlist.begin(), xlist.end()), xlist.end());
13 for (int i = 0; i < N; i++)
14     a[i] = 1 + (lower_bound(xlist.begin(), xlist.end(), a[i]) - xlist.begin());

```

맞았습니다!!

4744 KB

396 ms



2022 Winter Algorithm Camp

## 2370. 시장 선거 포스터

N개의 포스터, 포스터들의 양 끝 위치 정보  $l_i, r_i$  ( $1 \leq n \leq 10,000, 1 \leq l_i \leq r_i \leq 10^8$ ), 편의상 높이는 동일  
기존에 포스터가 붙어 있던 위치는 그 위에 덮어서 붙일 수 있다.  
입력된 순서대로 포스터를 붙인 후 보이는 포스터의 총 수를 구해보자.

2022 Winter Algorithm Camp

# 2370. 시장 선거 포스터

1. 주어진 정보를 있는 그대로 적용해보자.

ex) (1, 4), (2, 6), (8, 10), (3, 4), (7, 10)

0	1	2	3	4	5	6	7	8	9	10
	1	1	1	1						
		2	2	2	2	2				
								3	3	3
			4	4						
							5	5	5	5
↓										
	1	2	4	4	2	2	5	5	5	5

2022 Winter Algorithm Camp

## 2370. 시장 선거 포스터

1. 주어진 정보를 있는 그대로 적용해보자.

실제로 붙이는 것처럼 시뮬레이션을 하여 배열 등에 마킹한 후 실제 칸에 칠해져 있는 서로 다른 원소의 개수를 구하면 된다.

But, 실제 칸은  $1 \sim 10^8$ 까지의 값을 가질 수 있다..

2022 Winter Algorithm Camp

## 2370. 시장 선거 포스터

### 2. 문제의 성질 찾아보기

포스터의 실제 길이가 중요할까?

1	1	1	1		
	2	2	2	2	2
1	2	2	2	2	2



1	1	
	2	2
1	2	2

2022 Winter Algorithm Camp

## 2370. 시장 선거 포스터

3. 입력 제한과 범위를 살펴보자.

- 범위는  $1 \sim 10^8$
- 포스터의 개수는 10,000개 이하. 나올 수 있는 서로 다른 좌표는 20,000가지

좌표가 나올 수 있는 범위는 매우 큰데 실제 등장하는 좌표 개수는 적다.

실제 간격이 중요하지 않고 상대적인 크기만 중요하다.



좌표압축을 써서 각 포스터의  $l, r$ 을 상대적인 순서로 대응시켜보자.

2022 Winter Algorithm Camp

## 2370. 시장 선거 포스터

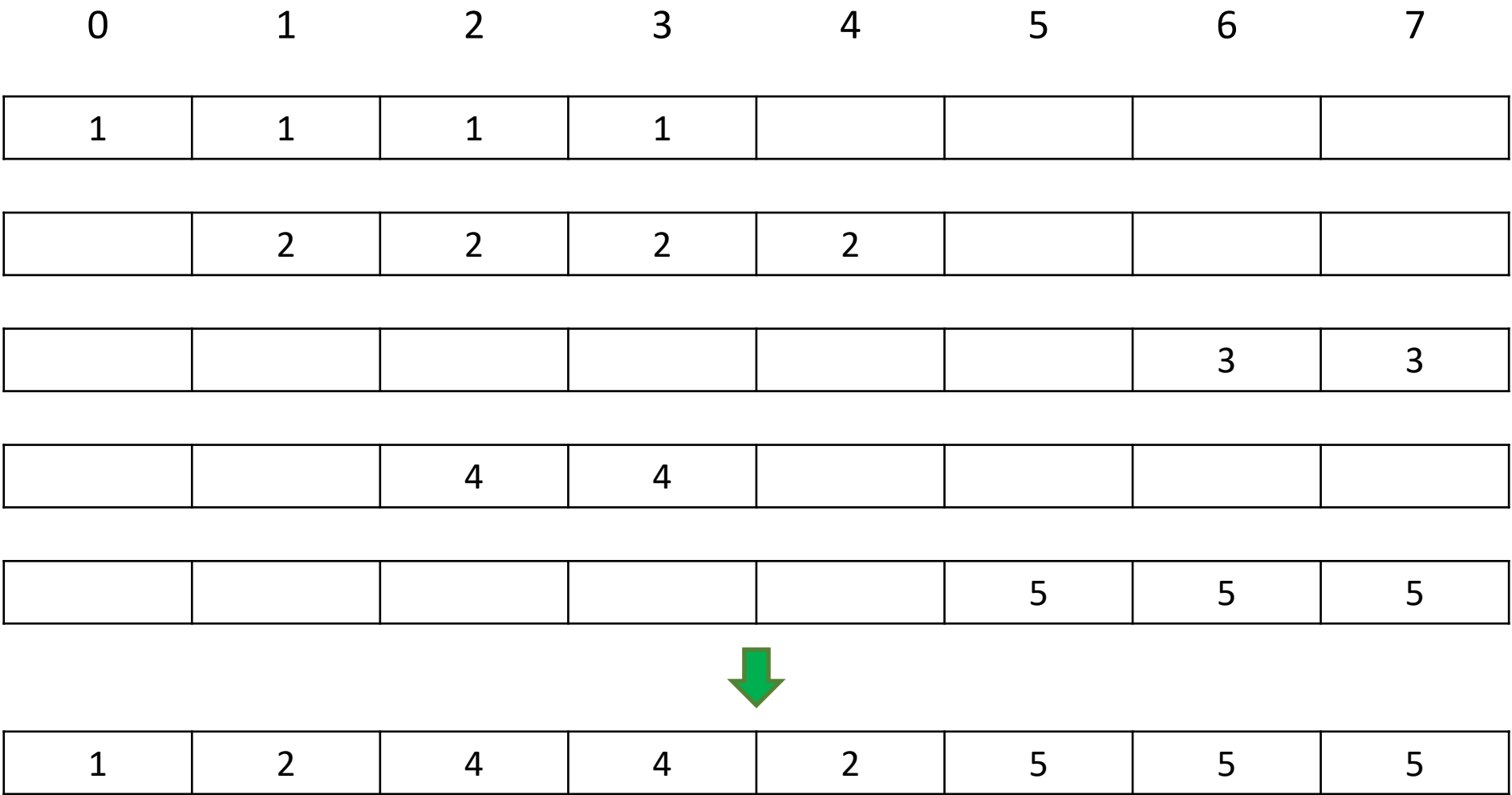
ex) (1, 4), (2, 6), (8, 10), (3, 4), (7, 10)

real	1	2	3	4	6	7	8	10
compressed	0	1	2	3	4	5	6	7

(1, 4), (2, 6), (8, 10), (3, 4), (7, 10)    =>    (0, 3), (1, 4), (6, 7), (2, 3), (5, 7)

# 2370. 시장 선거 포스터

ex) (0, 3), (1, 4), (6, 7), (2, 3), (5, 7)



2022 Winter Algorithm Camp

# 매개변수 탐색 – Introduction

\* 정보과학에서의 문제의 분류

- 결정 문제 (Decision Problem)
- 탐색 문제 (Search Problem)
- 계산 문제 (Counting Problem)
- 최적화 문제 (Optimization Problem)
- 함수 문제 (Function Problem)

결정 문제의 난이도  $\leq$  최적화 문제의 난이도



2022 Winter Algorithm Camp

# 매개변수 탐색 – Introduction

\* 매개변수 탐색 (Parametric Search)

결정 문제로 최적화 문제를 푸는 기법

특정 파라미터를 기준으로 조건을 만족하는 부분과 만족하지 않는 경계점 찾기

2022 Winter Algorithm Camp

## 16564. 히오스 프로그래머

$N$ 개의 캐릭터, 각 캐릭터의 레벨  $X_i$  ( $1 \leq N \leq 10^6, 1 \leq X_i \leq 10^9$ )

레벨을 최대 총합  $K$  ( $1 \leq K \leq 10^9$ )만큼 올릴 수 있다.

팀 목표 레벨  $T = \min(X_i)$  ( $1 \leq i \leq N$ )일 때 만들 수 있는 팀 목표 레벨의 최댓값을 구해보자.

2022 Winter Algorithm Camp

# 16564. 히오스 프로그래머

## 1. 답 후보군 설정

- 1) 최초 레벨들 중 최솟값보다 작을 수는 없다.  
-> 레벨은 올라가기만 할 것이다.
- 2) 가능한 최댓값은 캐릭터가 하나일 때,  $X_i, K$ 가 최대일 경우이다.

$\text{range} = [\min(X_i), 2 \times 10^9]$

# 16564. 히오스 프로그래머

## 2. 성질 찾기

목표 레벨을 임의의 자연수  $T'$  ( $\min(X_i) < T'$ )로 만들 수 있다면:

$T' - 1$  ( $\min(X_i) \leq T' - 1$ )로도 만들 수 있을까?

$T' - 2$  ( $\min(X_i) \leq T' - 2$ )로도 가능할까?



$T'$ 이하의 값으로는 항상 충족을 하니 그보다 큰 경우에 대해서 가능한지를 찾아주자.

True				False		
$\min(X_i)$	...	$T - 1$	$T$	$T + 1$	$T + 2$	...

2022 Winter Algorithm Camp

# 분할정복 - 도입

## \* 분할정복 (Divide & Conquer)

문제를 (비슷한 크기의) 둘 이상의 부분 문제로 나눈 뒤	(Divide)
각 문제에 대한 답을 계산하고	(Conquer)
원래 문제에 대한 답으로 병합	(Merge)

## \* 동적계획법 (Dynamic Programming)과의 차이

중복되는 부분 문제(Overlapping Subproblems) 여부

2022 Winter Algorithm Camp

# 분할정복 - 도입

\* Design Technique

```
1 void dnc(int l, int r){  
2     int m = (l + r) / 2;  
3     dnc(l, m);  
4     dnc(m + 1, r);  
5  
6     //sth merge solution  
7 }
```

\* Time Complexity

구간 길이  $N$ 에 대한 처리 비용을 이라  $T(N)$ 하고 문제를  $k$ 분할했을 때

$$T(N) = kT\left(\frac{N}{k}\right) + C(N)$$

2022 Winter Algorithm Camp

# 분할정복 – Example

\* 병합 정렬 (Merge sort)

수열을 절반의 크기로 나눈 뒤

(Divide)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---

1	4	2	8	5
---	---	---	---	---

7	4	9	3	0
---	---	---	---	---

2022 Winter Algorithm Camp

# 분할정복 – Example

\* 병합 정렬 (Merge sort)

나뉜 수열들끼리 정렬을 수행한 뒤 (Conquer)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---

1	2	4	5	8	0	3	4	7	9
---	---	---	---	---	---	---	---	---	---



# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---



0	3	4	7	9
---	---	---	---	---

⋮

⋮

--	--	--	--	--	--	--	--	--	--

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---



0	3	4	7	9
---	---	---	---	---

⋮

⋮

0									
---	--	--	--	--	--	--	--	--	--

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---



0	3	4	7	9
---	---	---	---	---

⋮

⋮

0	1								
---	---	--	--	--	--	--	--	--	--

2022 Winter Algorithm Camp

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---



0	3	4	7	9
---	---	---	---	---

⋮

⋮

0	1	2							
---	---	---	--	--	--	--	--	--	--

2022 Winter Algorithm Camp

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---



0	3	4	7	9
---	---	---	---	---

⋮

⋮

0	1	2	3						
---	---	---	---	--	--	--	--	--	--

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---



0	3	4	7	9
---	---	---	---	---

⋮

⋮

0	1	2	3	4					
---	---	---	---	---	--	--	--	--	--

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---



0	3	4	7	9
---	---	---	---	---

⋮

⋮

0	1	2	3	4	4				
---	---	---	---	---	---	--	--	--	--

2022 Winter Algorithm Camp

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---



0	3	4	7	9
---	---	---	---	---

⋮

⋮

0	1	2	3	4	4	5			
---	---	---	---	---	---	---	--	--	--



2022 Winter Algorithm Camp

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---

0	3	4	7	9
---	---	---	---	---

⋮

⋮

0	1	2	3	4	4	5	7		
---	---	---	---	---	---	---	---	--	--

2022 Winter Algorithm Camp

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---

0	3	4	7	9
---	---	---	---	---

⋮

⋮

0	1	2	3	4	4	5	7	8	
---	---	---	---	---	---	---	---	---	--

2022 Winter Algorithm Camp

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---



1	2	4	5	8
---	---	---	---	---

0	3	4	7	9
---	---	---	---	---

⋮

⋮

0	1	2	3	4	4	5	7	8	9
---	---	---	---	---	---	---	---	---	---

2022 Winter Algorithm Camp

# 분할정복 - Example

\* 병합 정렬 (Merge sort)

부분적으로 정렬된 수열들의 결과를 종합 (Merge)

1	4	2	8	5	7	4	9	3	0
---	---	---	---	---	---	---	---	---	---

1	2	4	5	8	0	3	4	7	9
---	---	---	---	---	---	---	---	---	---

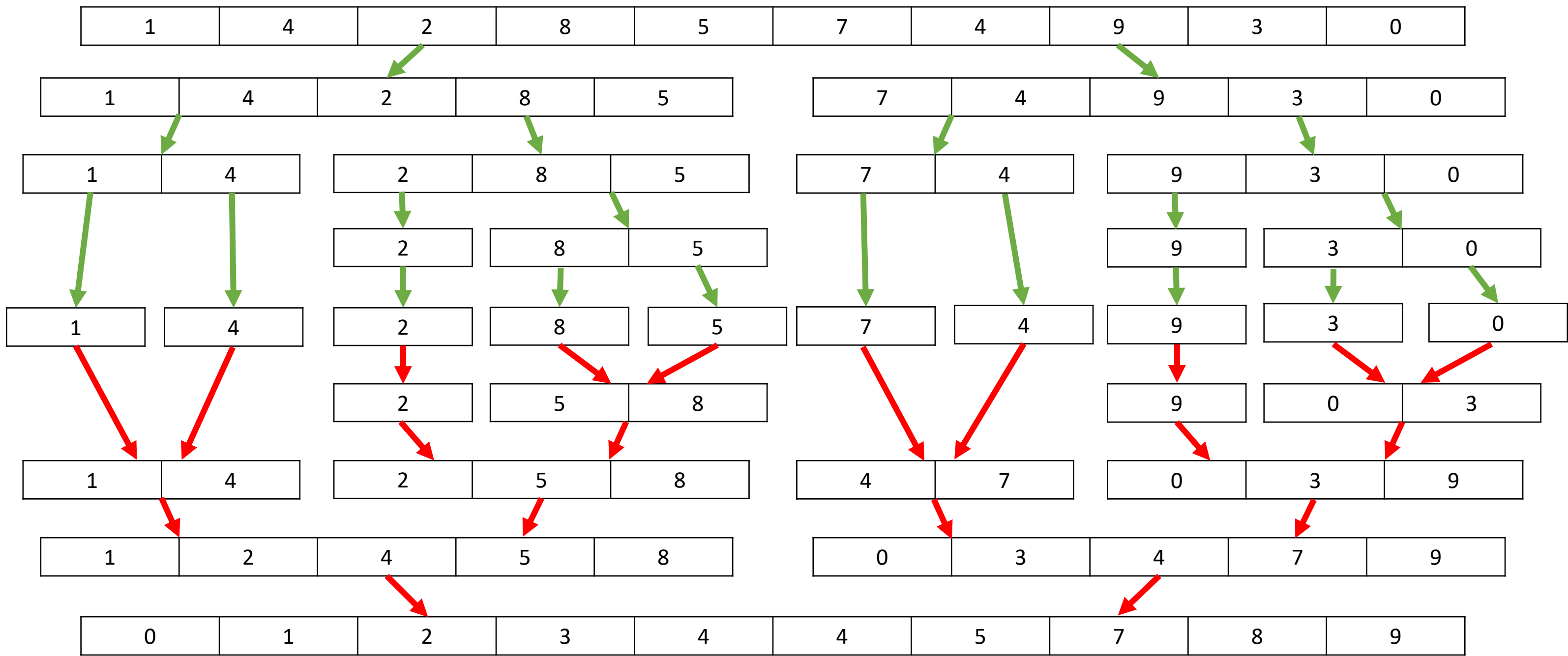
⋮

⋮

--	--	--	--	--	--	--	--	--	--

# 분할정복 - Example

\* 병합 정렬 (Merge sort)



# 분할정복 – Example

\* 병합 정렬 (Merge sort) – Time complexity

N개의 원소를 정렬하는데 드는 비용을  $T(N)$ 이라 할 때,

$$T(N) = 2T\left(\frac{N}{2}\right) + N,$$

$$\begin{aligned} \text{let } N = 2^m, T(N) = T(2^m) &= 2^1 T(2^{m-1}) + 2^m \\ &= 2^1 (2^1 T(2^{m-2}) + 2^{m-1}) + 2^m = 2^2 T(2^{m-2}) + 2 \cdot 2^m \\ &= 2^2 (2T(2^{m-3}) + 2^{m-2}) + 2 \cdot 2^m = 2^3 T(2^{m-3}) + 3 \cdot 2^m \\ &= \dots = 2^m T(2^{m-m}) + m \cdot 2^m = 2^m T(0) + m \cdot 2^m \\ &= (m + 1)2^m = \log N \cdot N = N \log N \end{aligned}$$

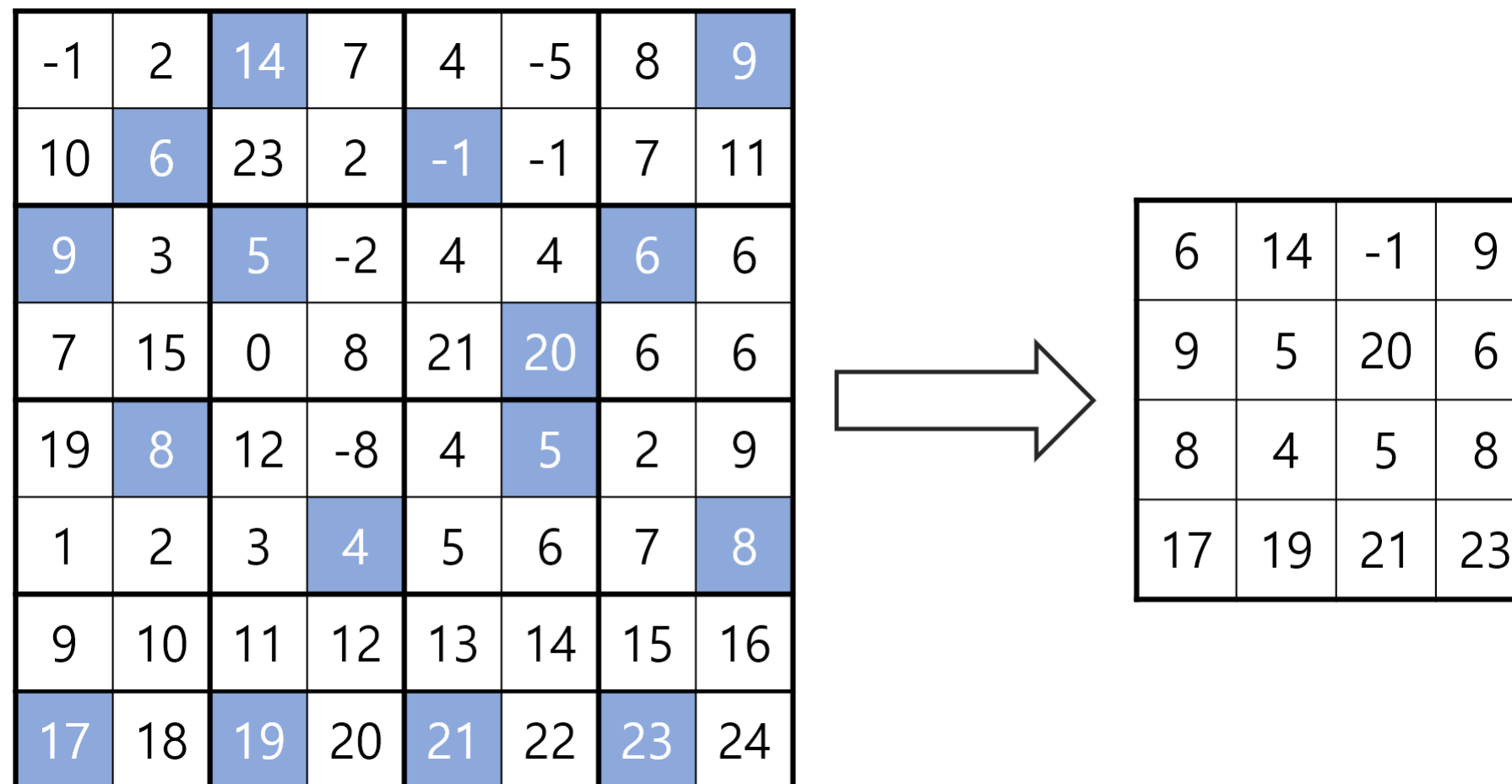
2022 Winter Algorithm Camp

# 17829. 222-폴링

$N \times N$  크기의 행렬 ( $N = 2^K, 1 \leq K \leq 10, |a_{ij}| \leq 10,000$ )

행렬을  $2 \times 2$  정사각형으로 나누고 각 정사각형에서 2번째로 큰 수만 남기는 행위를 222-폴링이라 한다.

222-폴링을 반복해서 크기가  $1 \times 1$  이 되었을 때 어떤 값이 남아있을 까?



2022 Winter Algorithm Camp

## 17829. 222-폴링

## 1. 분할 기준 설정

전체 문제는 아래 그림과 같은 4개의 정사각형들로부터 나온 단일 값들로부터 구해진다.

따라서 변의 길이를 절반씩 줄이는 방식으로 수행

-1	2	14	7	4	-5	8	9
10	6	23	2	-1	-1	7	11
9	3	5	-2	4	4	6	6
7	15	0	8	21	20	6	6
19	8	12	-8	4	5	2	9
1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24



2022 Winter Algorithm Camp

# 17829. 222-폴링

## 2. 분할이 되지 않는 최소 단위

크기가  $2 \times 2$ 인 경우 분할을 할 필요가 없다.

```
1 int dnc(int len, int r, int c){
2     if(len == 2){
3         // base case, return sth
4     }
5
6     int x[2][2];
7
8     //sth merge solution with recursive call
9 }
```

2022 Winter Algorithm Camp

# 분할정복을 이용한 거듭제곱

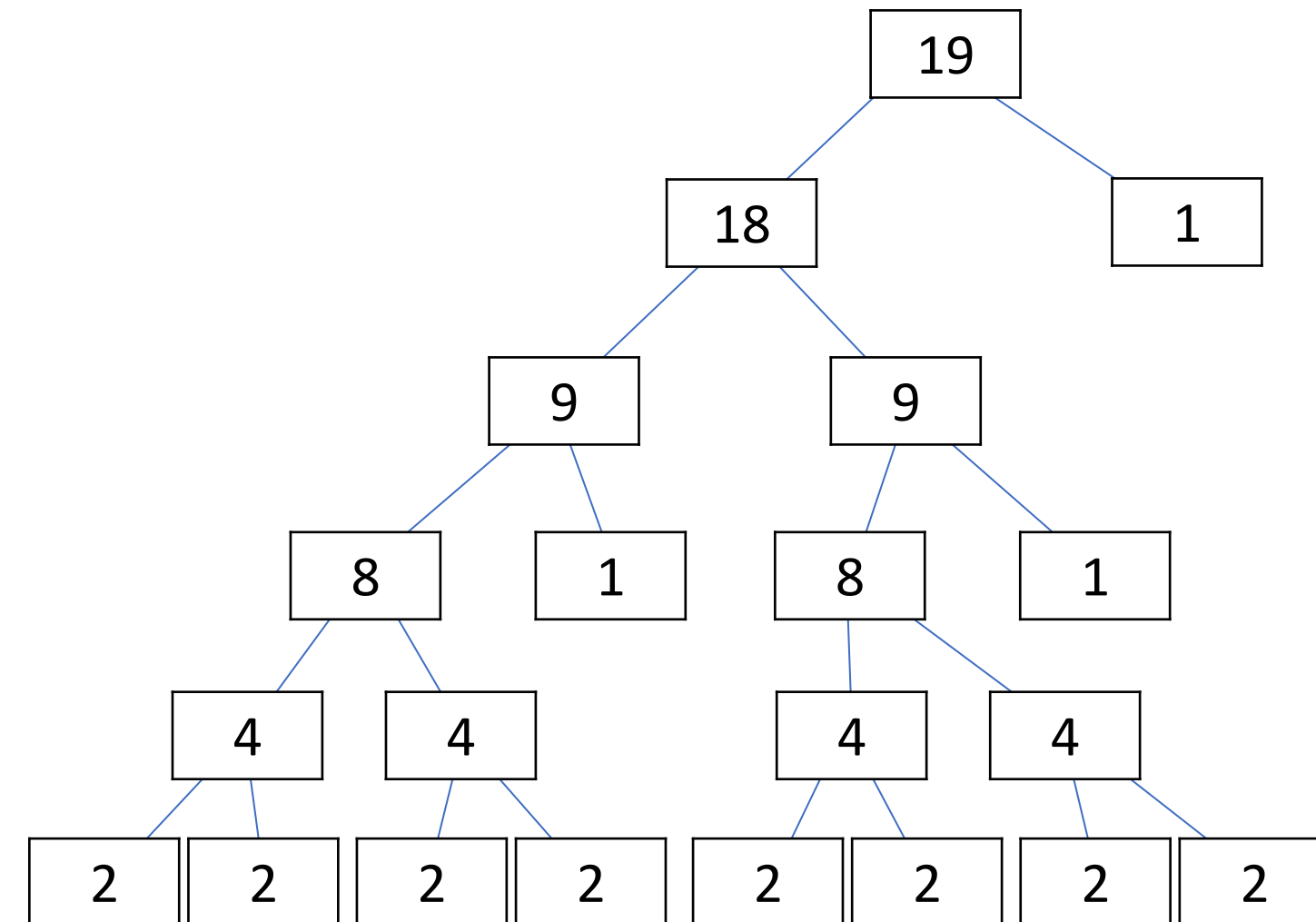
\* 분할정복을 이용한 거듭제곱 (Exponentiation by Squaring)

- $a^n = a \times a \times \cdots \times a$
- $O(n)$ 을  $O(\log n)$ 으로 줄여보자.

# 분할정복을 이용한 거듭제곱

\* Example

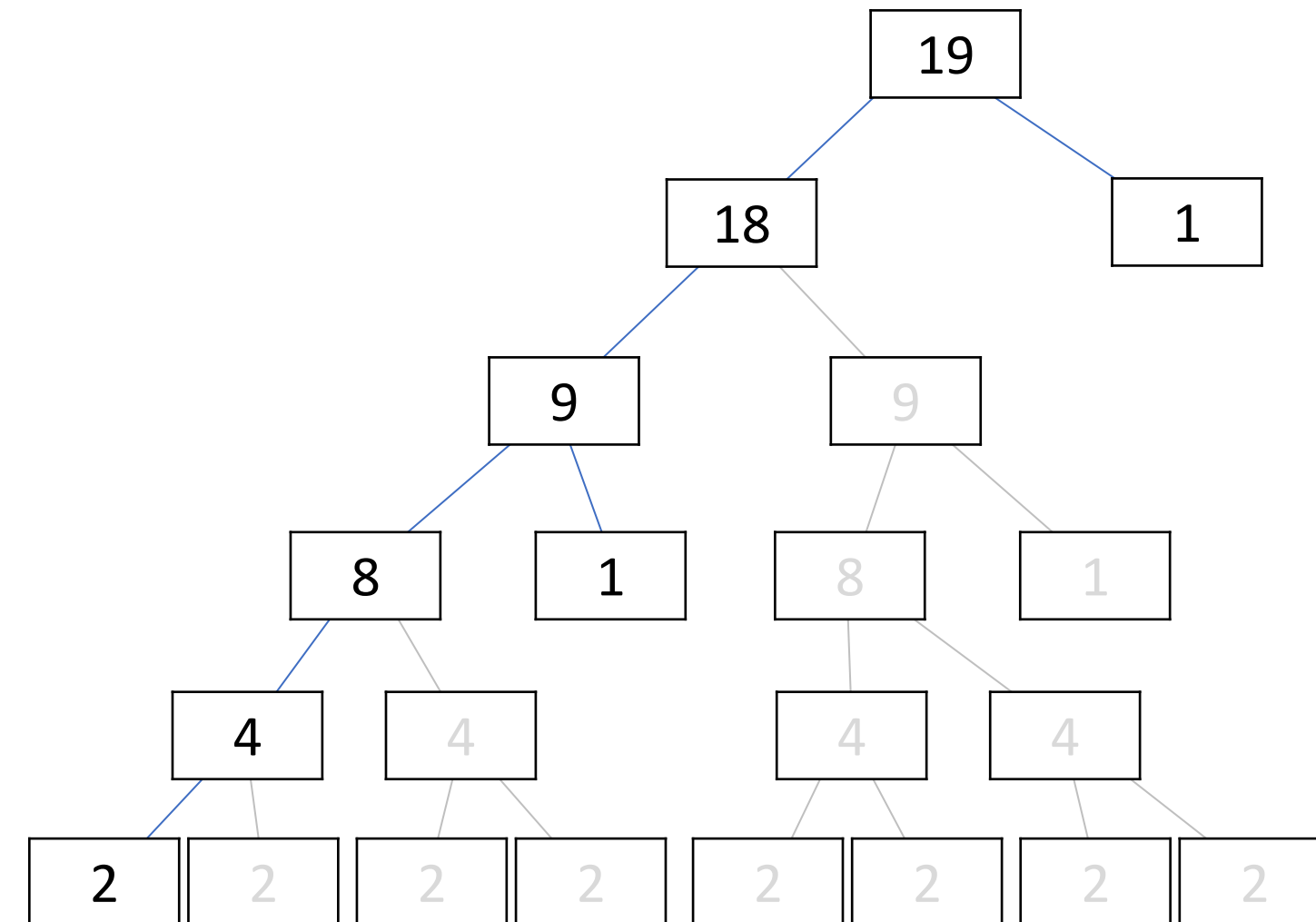
$$\begin{aligned}
 x^{19} &= x^{18+1} \\
 &= x^{9+9} \times x^1 = x^9 \times x^9 \times x^1 \\
 &= x^{8+1} \times x^{8+1} \times x^1 = x^8 \times x^8 \times x^3 \\
 &= x^{4+4} \times x^{4+4} \times x^3 = x^4 \times x^4 \times x^4 \times x^4 \times x^3 \\
 &= x^{2+2} \times \dots \times x^{2+2} \times x^3 = x^2 \times \dots \times x^2 \times x^3
 \end{aligned}$$



# 분할정복을 이용한 거듭제곱

\* Example

$$\begin{aligned}
 x^{19} &= x^{18+1} \\
 &= x^{9+9} \times x^1 = x^9 \times x^9 \times x^1 \\
 &= x^{8+1} \times x^{8+1} \times x^1 = x^8 \times x^8 \times x^3 \\
 &= x^{4+4} \times x^{4+4} \times x^3 = x^4 \times x^4 \times x^4 \times x^4 \times x^3 \\
 &= x^{2+2} \times \dots \times x^{2+2} \times x^3 = x^2 \times \dots \times x^2 \times x^3
 \end{aligned}$$



# 분할정복을 이용한 거듭제곱

\* Method

- base case:  $b=0$
- 홀수인 경우: -1해서 짝수로 만들고 a 곱하기
- 짝수인 경우: 그 절반에 해당하는 값 구하고 두번 곱하기

```
1 pow(a, b):  
2     if b == 0:  
3         return 1  
4     if b % 2 == 1:  
5         return pow(a, b-1) * a  
6  
7     half <- pow(a, b/2)  
8     return half * half
```

# 분할정복을 이용한 거듭제곱 - 도입

\* 행렬 곱셈

두 행렬  $A, B$ 의 크기가 각각  $m \times n, n \times p$ 일 때

$$\mathbf{A} = \begin{pmatrix} \overline{a_{11} & a_{12} & \cdots & a_{1n}} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} \overline{b_{11}} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{np} \end{pmatrix}$$

행렬곱  $C = AB$ 의 크기:  $m \times p$ ,

$$\mathbf{C} = \begin{pmatrix} \overline{a_{11}b_{11} + \cdots + a_{1n}b_{n1}} & a_{11}b_{12} + \cdots + a_{1n}b_{n2} & \cdots & a_{11}b_{1p} + \cdots + a_{1n}b_{np} \\ a_{21}b_{11} + \cdots + a_{2n}b_{n1} & a_{21}b_{12} + \cdots + a_{2n}b_{n2} & \cdots & a_{21}b_{1p} + \cdots + a_{2n}b_{np} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}b_{11} + \cdots + a_{mn}b_{n1} & a_{m1}b_{12} + \cdots + a_{mn}b_{n2} & \cdots & a_{m1}b_{1p} + \cdots + a_{mn}b_{np} \end{pmatrix}$$

# 분할정복을 이용한 거듭제곱 - 도입

\* 선형 결합 (linear combination)

$$a_1x_1 + a_2x_2 + \dots + a_nx_n$$

\* 선형 방정식과 행렬 곱셈

$$\begin{cases} ax + by + cz = p \\ dx + ey + fz = q \\ gx + hy + iz = r \end{cases} \Leftrightarrow \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} p \\ q \\ r \end{pmatrix}$$

# 분할정복을 이용한 거듭제곱 - 행렬 거듭제곱

\*  $N \times N$  크기 행렬  $A$ 의 거듭제곱  $A^m$

정수의 거듭제곱과 컨셉은 동일

차이점: base case,

$$a^0 = 1, A^0 = I, (I: \text{identity matrix})$$

\* Time Complexity

$O(N^3)$  for matrix multiplication,

$O(\log m)$  for exponentiation,

$O(N^3 \log m)$  total

```

1 // I = 1 for (i,i),
2 //   = 0 for (i,j), i!=j
3
4 matrix_pow(A, m):
5     if m == 0:
6         return I
7     if m % 2 == 1:
8         return pow(A, m-1) * A
9
10    half <- pow(A, m/2)
11    return half * half

```



2022 Winter Algorithm Camp

# 분할정복을 이용한 거듭제곱 - 도입

\* 관계식이 선형 결합인 경우

$$\begin{aligned} a_n &= \sum_{k=1}^m c_k \cdot a_{n-k} \\ &= c_1 \cdot a_{n-1} + c_2 \cdot a_{n-2} + \cdots + c_m \cdot a_{n-m} \end{aligned}$$

2022 Winter Algorithm Camp

# 14440. 정수 수열

모든  $n \geq 2$ 에 대해서,  $A_n = xA_{n-1} + yA_{n-2}$  ( $1 \leq x, y \leq 99, 0 \leq n < 10^8$ )

$A_0, A_1$ 이 주어졌을 때  $A_n$ 의 마지막 두 자리를 구해보자.

2022 Winter Algorithm Camp

# 14440. 정수 수열

1. 그냥 반복문으로 해도 되지 않나요?

사실 맞습니다..  $n$ 값이 작고 연산량도 적어서 그냥 돌려도 됩니다.....

2022 Winter Algorithm Camp

# 14440. 정수 수열

## 2. 선형 방정식의 행렬화

$$A_n = xA_{n-1} + yA_{n-2}$$

$A_{n-1}, A_{n-2}$  두개의 항에 대한 관계로 결정된다.

$v_n = \begin{bmatrix} A_n \\ A_{n-1} \end{bmatrix}$ 이라 하면 어떤  $2 \times 2$ 크기의 행렬  $W$ 에 대하여  $W \times v_n = v_{n+1}$ 이 되게 하는  $W$ 를 정의해보자.

2022 Winter Algorithm Camp

# 14440. 정수 수열

## 2. 선형 방정식의 행렬화

$$A_n = xA_{n-1} + yA_{n-2}$$

$$W \begin{bmatrix} A_n \\ A_{n-1} \end{bmatrix} = \begin{bmatrix} A_{n+1} \\ A_n \end{bmatrix}$$

2022 Winter Algorithm Camp

# 14440. 정수 수열

## 3. 일반화

$$A_n = xA_{n-1} + yA_{n-2}$$

$$Wv_1 = v_2$$

$$W^2v_1 = W \cdot Wv_1 = W \cdot v_2 = v_3$$

$$W^{n-1}v_1 = v_n$$

# Appendix – Additional Topics

## \* 실수 범위에서의 이분탐색 혹은 parametric search

실수 범위에서는 절반값을 구해가는 과정에서 우리가 의도한 값과 동일해지는 것을 기대하기 힘든 경우가 있습니다. 부동 소수점 방식에 의한 실수 오차가 나기 때문인데, 이를 다루는 것은 힘든 일입니다.

두가지 방식으로 간접적인 해결을 할 수 있습니다.

### 1) 구간 탐색 횟수 제한하기

앞서 배웠던 방식에서 구간이 절반씩 작아지는 과정이 최대  $O(\log N)$ 번 일어났음을 기억해본다면, 문제에서 요구하는 실수오차 범위(ex:  $10^{-3}$ 까지의 오차를 허용한다)에 다다르기 위해서 추가로 돌려야 하는 반복횟수를 생각해볼 수 있습니다. 경험적으로 60~100번 사이를 돌려보고 있습니다.

### 2) 구간 범위의 epsilon 설정하기

구간 설정이 반복적으로 일어남에 따라  $[L, R]$ 의 길이가 절반씩 줄어들게 되고, 일정 이하로 줄어들게 된다면 구간 길이가 문제에서 요구하는 실수오차 범위 내로 들어오게 됩니다. 이 경우 추가적인 반복을 하지 않아도 됩니다.

# Appendix – Additional Topics

\* LIS(Longest Increasing Subsequence) with lower\_bound

앞서 동적계획법에서 배웠던 방식으로는 LIS를  $O(N^2)$ 만에 구할 수 있었습니다.

lower\_bound를 이용하면 실제 LIS의 구성을 구하는데는 추가 작업이 필요하지만, 단순히 길이만 구한다면  $O(N \log N)$ 에 문제를 해결할 수 있습니다.

수열의 각 원소를 순차적으로 보면서 list를 append하거나 increasing한 성질을 깨지 않는 것을 지켰을 때 어느 위치에 들어가는지를 찾아서 넣는 방식으로 만들어진 list의 길이가 LIS의 길이가 됩니다.

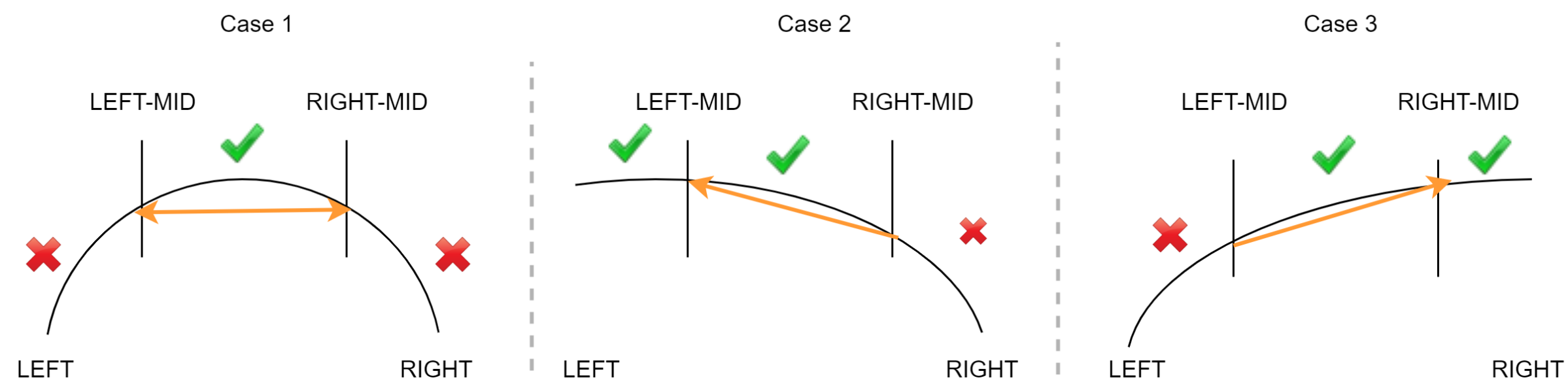


# Appendix – Additional Topics

## \* 삼분 탐색 (Ternary Search)

단조 증가(혹은 감소)의 조건에서 해를 찾는 것이 이분탐색이라면, 파라미터 값의 증가에 따라 함수값이 볼록(혹은 오목)한 형태를 보이는 경우에는 삼분탐색을 사용하여 해를 구할 수 있습니다.

구간을 삼등분하고 분점에서의 함수값을 비교하여 그 값을 기준으로 범위를 재설정합니다.



### Ternary search

# Appendix – Additional Topics

## \* 구간 트리 (Segment Tree)

분할정복에서 절반씩 나뉜 구간들의 정보를 저장(혹은 caching)하여 관리할 수 있는 자료구조입니다.

상위 레벨의 노드에서는 하위 두 노드에 대한 구간 정보를 포함하게끔 구성하면 분할정복에서와 마찬가지로 분할의 최대 깊이가  $\log N$ 이 되고, 구간들 정보를 표현하기 위해 필요한 노드의 개수는  $2N$ 개 이하가 됩니다.

분할정복에서 했던 것보다 좀더 일반화된 구간 범위 및 정보에 대한 처리를  $\log$ 시간에 수행할 수 있습니다.

[1, 16]															
[1, 8]								[9, 16]							
[1, 4]				[5, 8]				[9, 12]				[13, 16]			
[1, 2]		[3, 4]		[5, 6]		[7, 8]		[9, 10]		[11, 12]		[13, 14]		[15, 16]	
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

# Appendix – Problems

## 필수문제

10816	숫자 카드 2
2370	시장 선거 포스터
16564	히오스 프로그래머
17829	222-폴링
1802	종이 접기
1629	곱셈

## 연습문제

1920	수 찾기	2630	색종이 만들기
20551	Sort 마스터 배지훈의 후계자	1992	쿼드트리
10975	데크 소트 2	5904	Moo 게임
11997	Load Balancing (Silver)	18222	투에-모스 문자열
11973	Angry Cows (Silver)	14731	謎紛芳索紀 (Large)
16766	Convention	14440	정수 수열