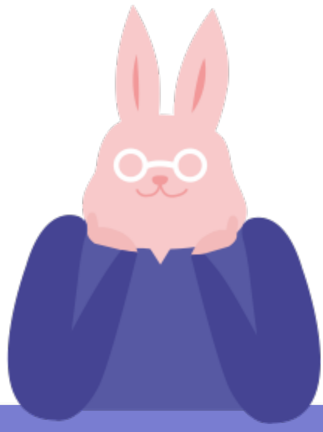
Dynamic Programming 1

2016. 12. 6.

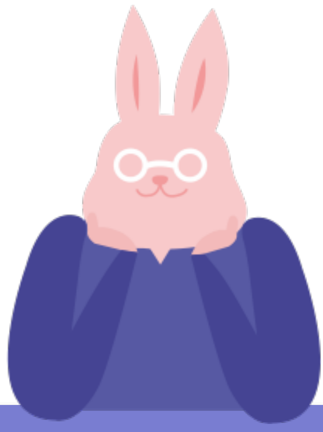
신현규



피드백

- 지난주 수업이 좋았습니다.
  - 우리에게 풀 시간을 주었다는 점
  - 다른 사람의 생각을 들어볼 수 있었다는 점

- 많은 피드백은 **여러분께 더 알맞는** 강의를 만듭니다



지난 시간 요약

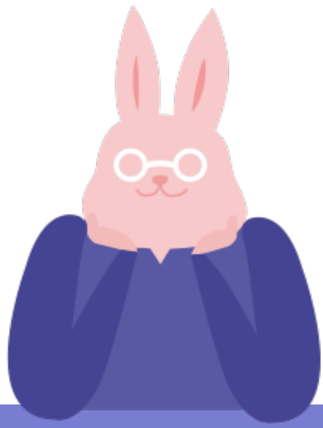
- 왜 ICPC인가?
- 분할정복법
  - Merge sort
  - Quick sort
  - 연속부분 최대합
- 탐욕적 기법
  - Interval Scheduling
  - 기울기의 최댓값 구하기

Machine Learning 을 배울 준비

- 알고리즘 디자인 Training 을 위해서
  - 이미 디자인 되어있는 여러 알고리즘을 배우고,
  - 이를 직접 구현해보자

- 정확성을 증명할 수 없는 문제를 풀기 위해서
  - 일단 정답이 존재하는 문제부터 제대로 증명하고 풀어보자

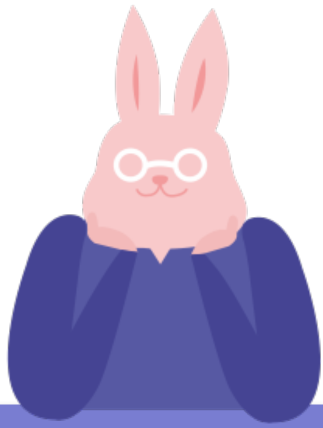
- 제한시간 내에 정답을 낸다는 것을 보이기 위해서
  - 간단한 알고리즘의 시간복잡도부터 계산해보자



Merge sort

- O(n log n) 정렬의 가장 대표적인 예

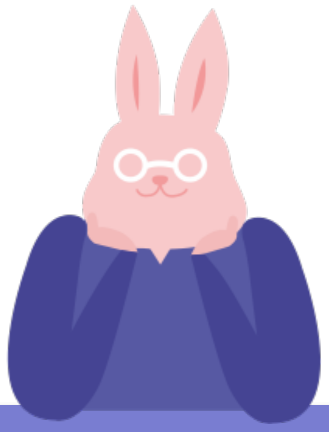
3	5	7	2	5	9	13	11	24	11	23	1	4	5	3	2



Merge sort

- O(n log n) 정렬의 가장 대표적인 예

3	5	7	2	5	9	13	11	24	11	23	1	4	5	3	2

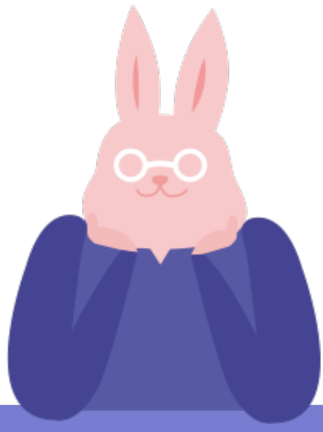


Merge sort

- O(n log n) 정렬의 가장 대표적인 예

3	5	7	2	5	9	13	11

24	11	23	1	4	5	3	2



Merge sort

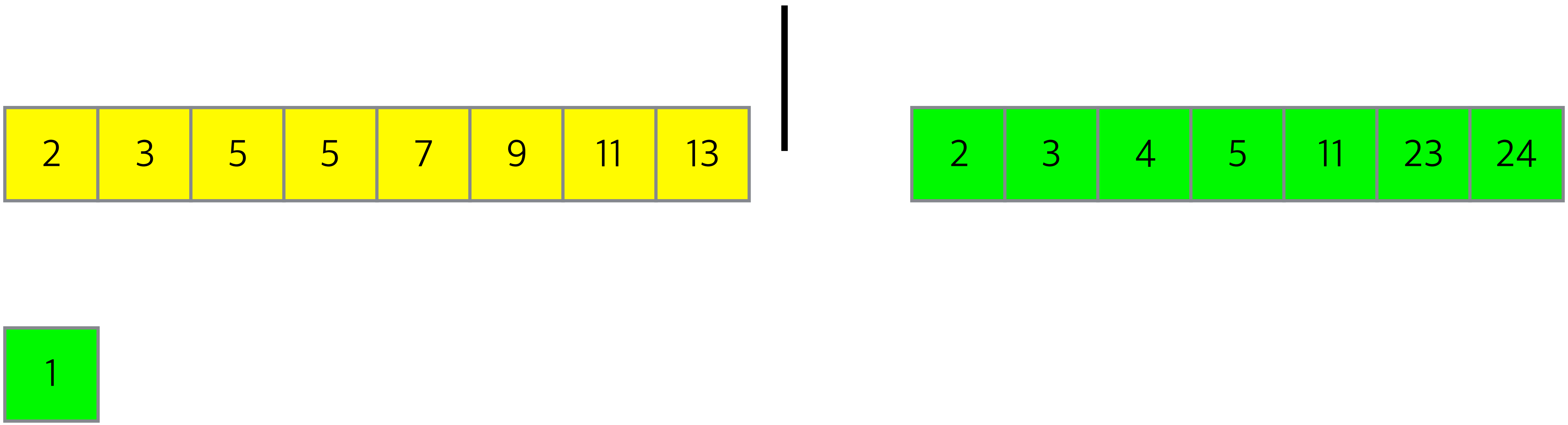
- O(n log n) 정렬의 가장 대표적인 예

2	3	5	5	7	9	11	13

1	2	3	4	5	11	23	24

Merge sort

- O(n log n) 정렬의 가장 대표적인 예



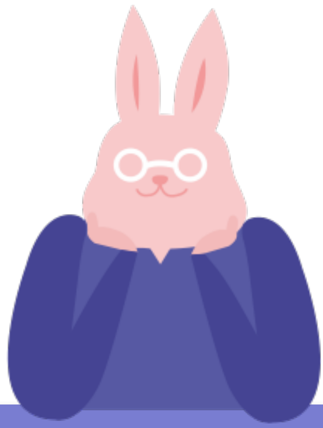
Merge sort

- O(n log n) 정렬의 가장 대표적인 예

3	5	5	7	9	11	13

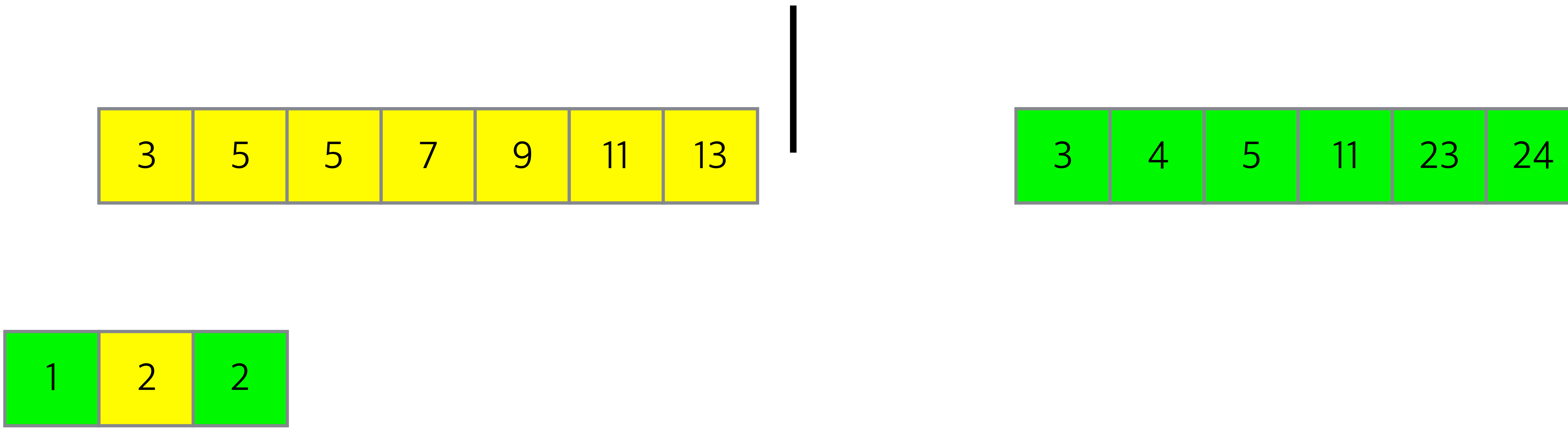
2	3	4	5	11	23	24

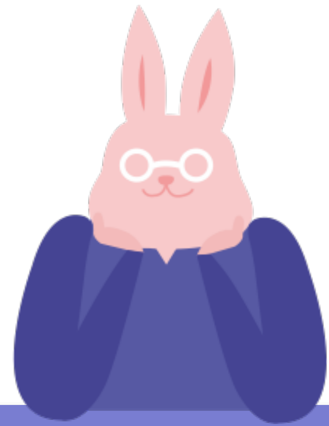
1	2



Merge sort

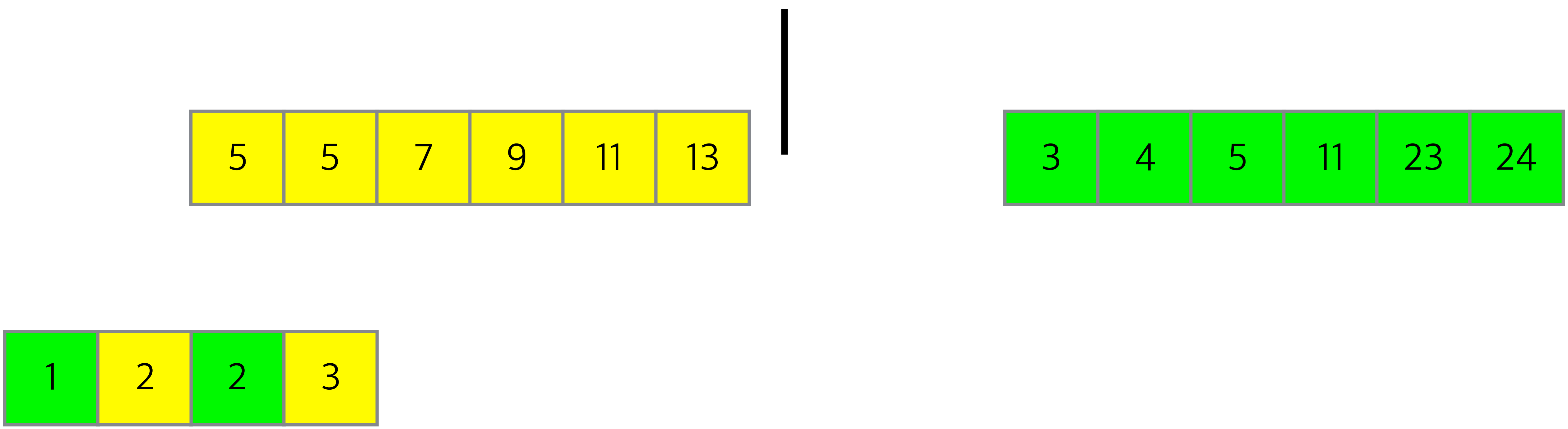
- O(n log n) 정렬의 가장 대표적인 예





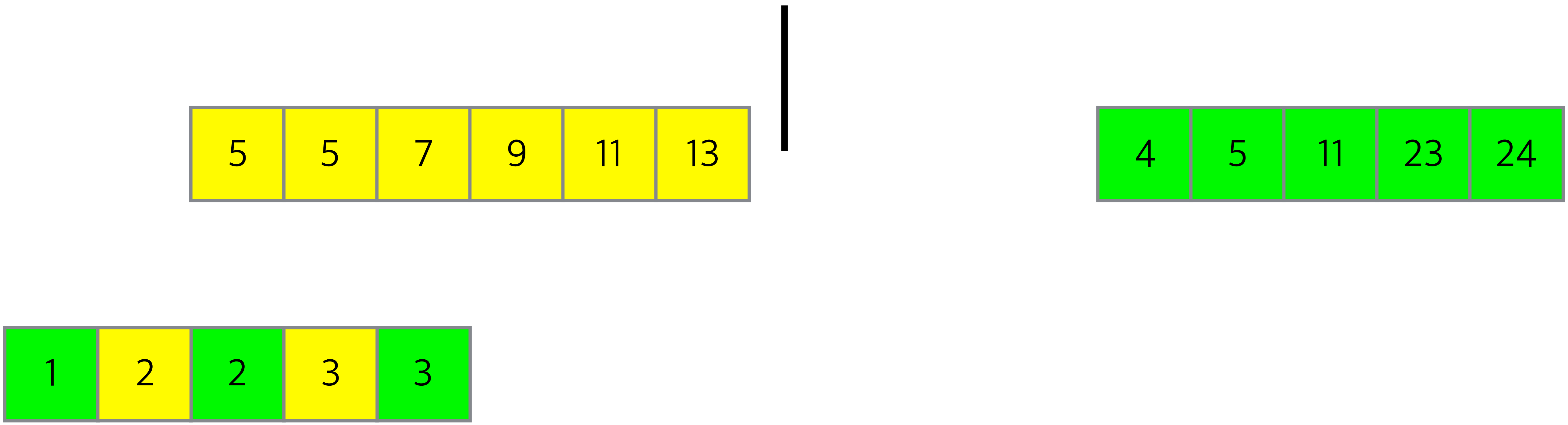
Merge sort

- O(n log n) 정렬의 가장 대표적인 예



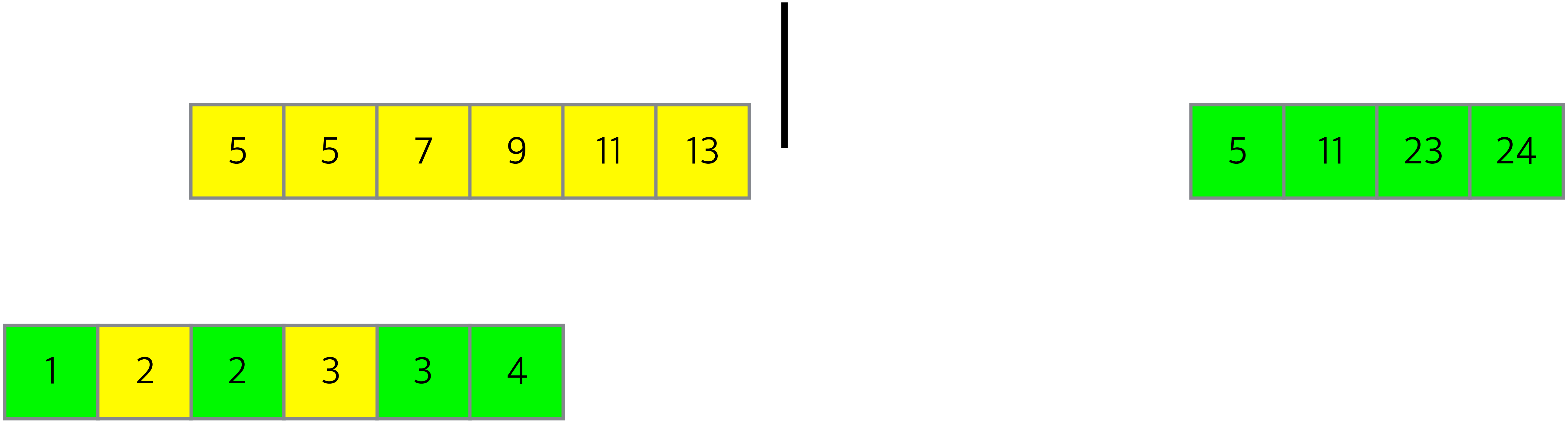
Merge sort

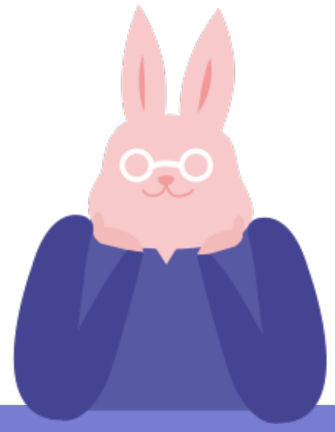
- O(n log n) 정렬의 가장 대표적인 예



Merge sort

- O(n log n) 정렬의 가장 대표적인 예





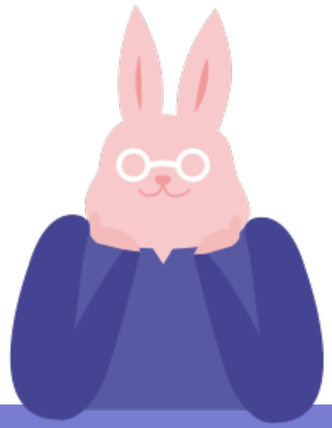
Merge sort

- O(n log n) 정렬의 가장 대표적인 예

5	7	9	11	13

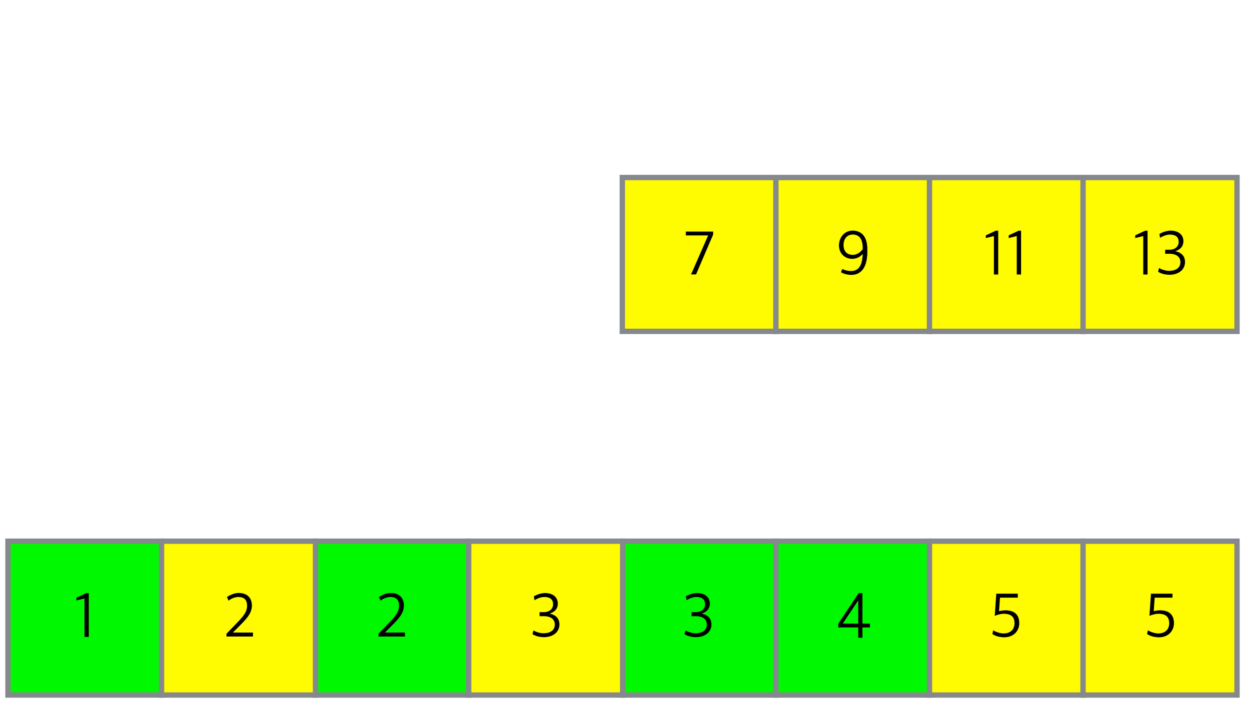
5	11	23	24

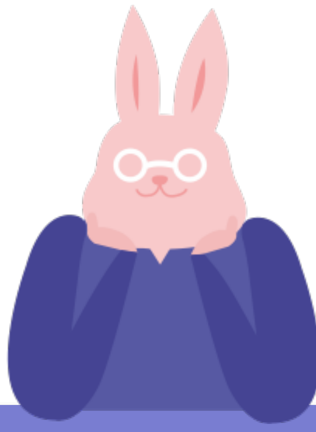
1	2	2	3	3	4	5



Merge sort

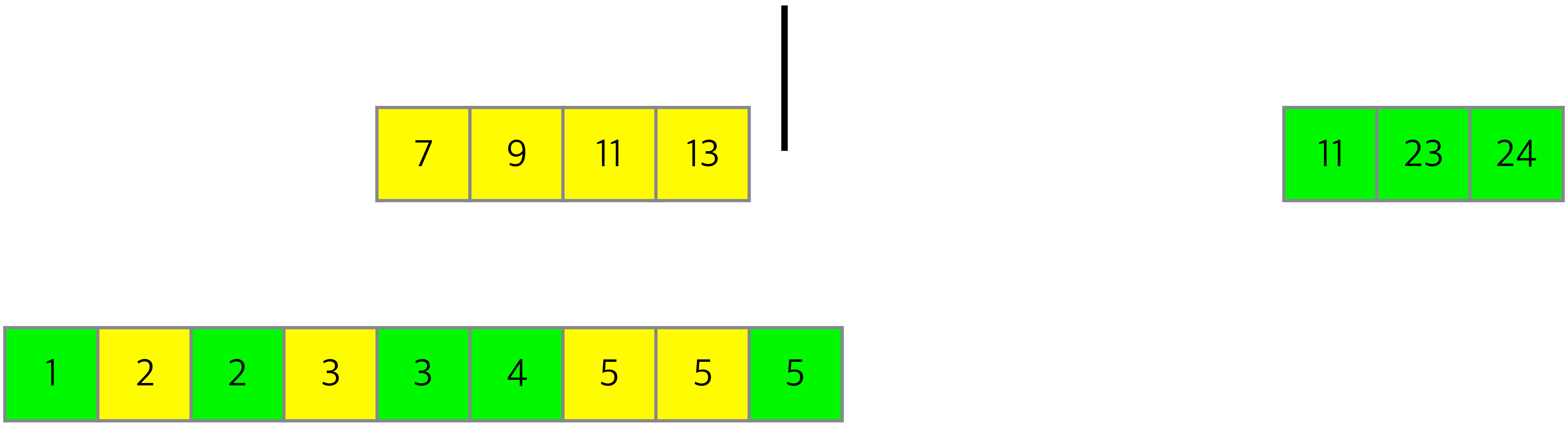
- O(n log n) 정렬의 가장 대표적인 예





Merge sort

- O(n log n) 정렬의 가장 대표적인 예



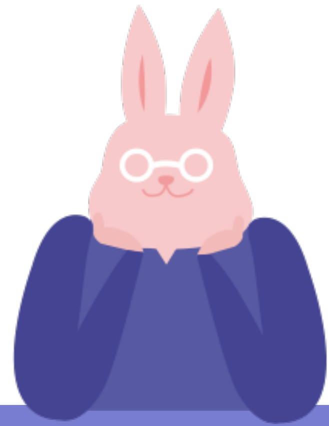
Merge sort

- O(n log n) 정렬의 가장 대표적인 예

9	11	13

11	23	24

1	2	2	3	3	4	5	5	5	7



Merge sort

- O(n log n) 정렬의 가장 대표적인 예

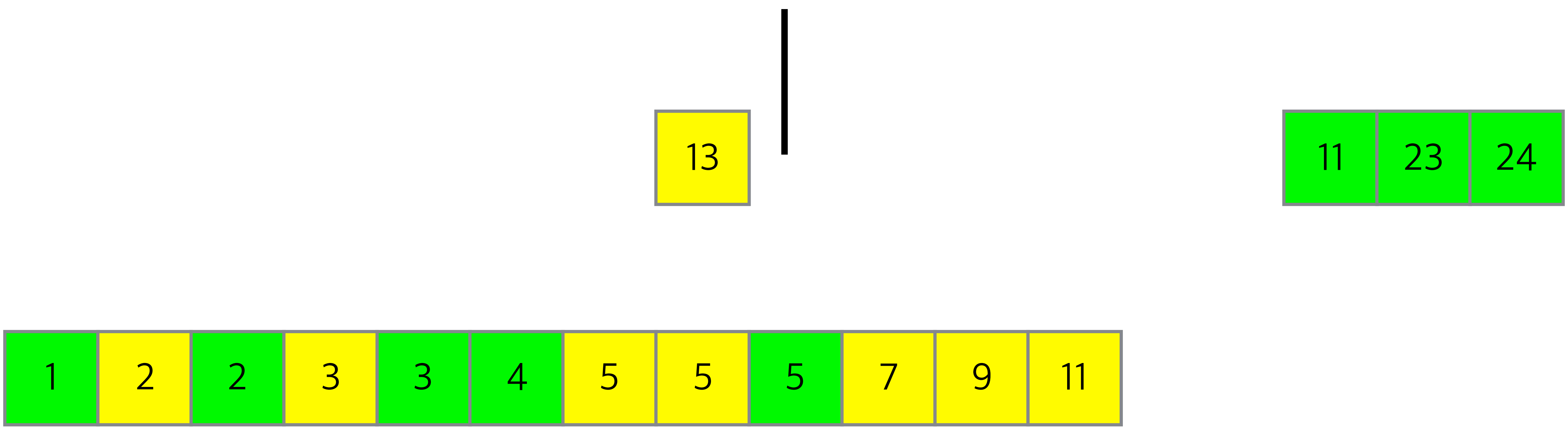
11	13

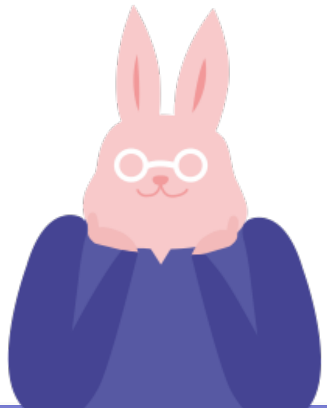
11	23	24

1	2	2	3	3	4	5	5	5	7	9

Merge sort

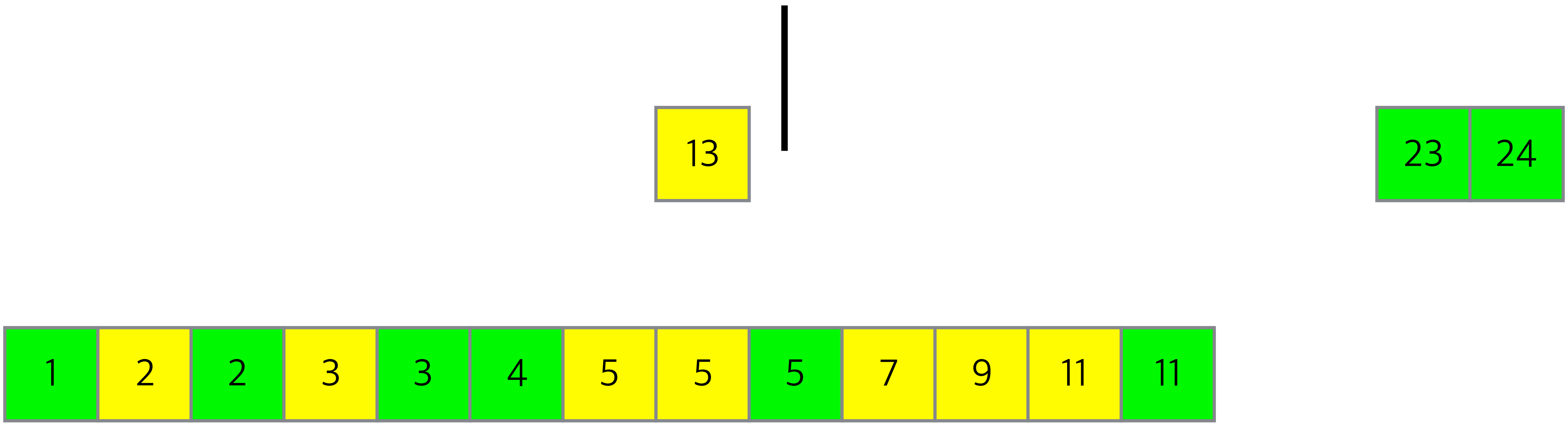
- O(n log n) 정렬의 가장 대표적인 예





Merge sort

- O(n log n) 정렬의 가장 대표적인 예

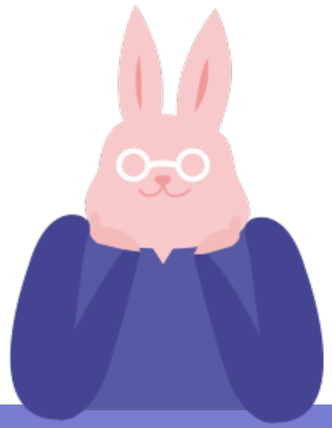


Merge sort

- O(n log n) 정렬의 가장 대표적인 예

23	24

1	2	2	3	3	4	5	5	5	7	9	11	11	13

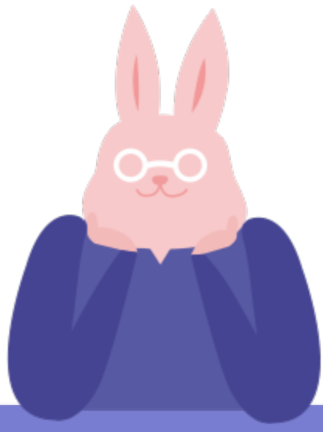


Merge sort

- O(n log n) 정렬의 가장 대표적인 예

24

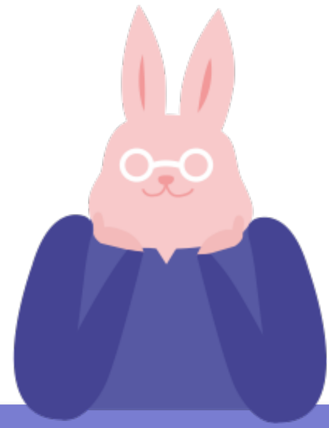
1	2	2	3	3	4	5	5	5	7	9	11	11	13	23



Merge sort

- O(n log n) 정렬의 가장 대표적인 예

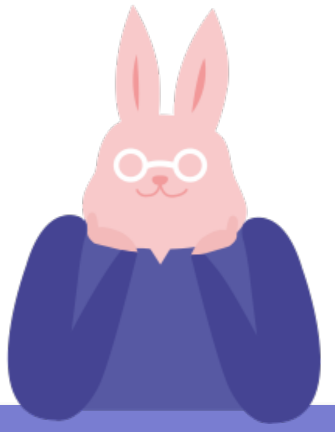
1	2	2	3	3	4	5	5	5	7	9	11	11	13	23	24



Merge sort

- O(n log n) 정렬의 가장 대표적인 예

1	2	2	3	3	4	5	5	5	7	9	11	11	13	23	24



Merge sort : 시간복잡도

3	5	7	2	5	9	13	11	24	11	23	1	4	5	3	2

T(N) = 2 * T(N/2) + O(N)

3	5	7	2	5	9	13	11

T(N/2) = 2 * T(N/4) + **O(N/2)**

3	5	7	2	5	9	13	11

T(N/2) = 2 * T(N/4) + **O(N/2)**

3	5	7	2

T(N/4) = ⋯ + **O(N/4)**

5	9	13	11

T(N/4) = ⋯ + **O(N/4)**

3	5	7	2

T(N/4) = ⋯ + **O(N/4)**

5	9	13	11

T(N/4) = ⋯ + **O(N/4**

Quick Sort

- 또 다른 O(n log n) 정렬의 대표적인 예

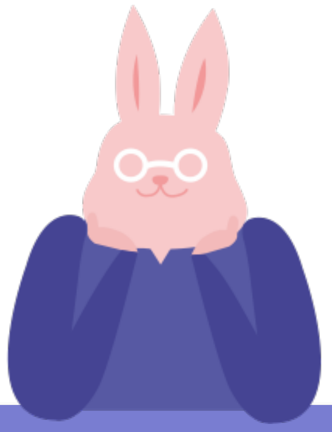
3	5	7	2	5	9	13	11	24	11	23	1	4	5	3	2

Quick Sort

- 또 다른 O(n log n) 정렬의 대표적인 예

3	5	7	2	5	9	13	11	24	11	23	1	4	5	3	2

Pivot!



Quick Sort

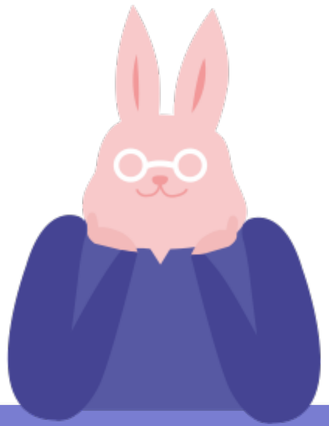
- 또 다른 O(n log n) 정렬의 대표적인 예

3	5	7

5	9	13	11	24	11	23

4	5

1	3	2	2



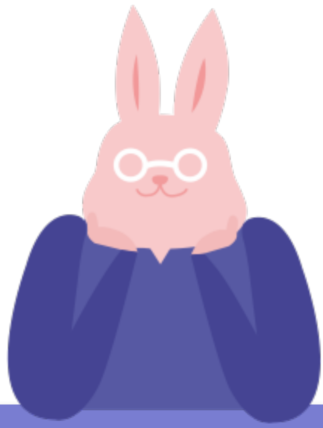
Quick Sort

- 또 다른 O(n log n) 정렬의 대표적인 예

3

1	3	2	2

5	7	5	9	13	11	24	11	23	4	5



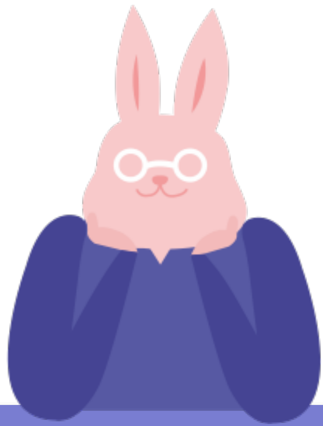
Quick Sort

- 또 다른 O(n log n) 정렬의 대표적인 예

3

1	2	2	3

4	5	5	5	7	9	11	11	13	23	24



Quick Sort

- 또 다른 O(n log n) 정렬의 대표적인 예

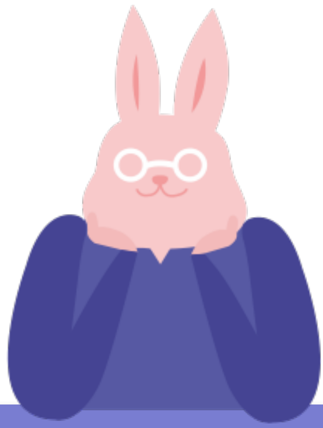
1	2	2	4	3	4	5	5	5	7	9	11	11	13	23	24

Quick Sort : 시간복잡도

- N개의 숫자를 정렬하는 데에 걸리는 시간을 T(N)이라 하자

- T(N) = 2 * T(N/2) + O(N)
  (대충 절반으로 나누어진다고 가정하면)

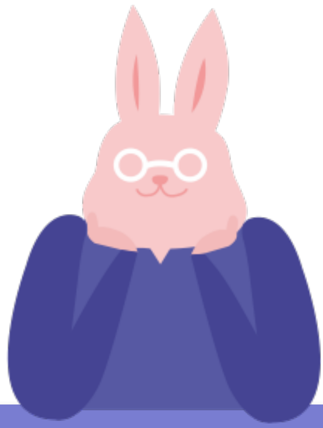
- O(N log N)



연속부분 최대합

- 연속부분 최대합을 구하라

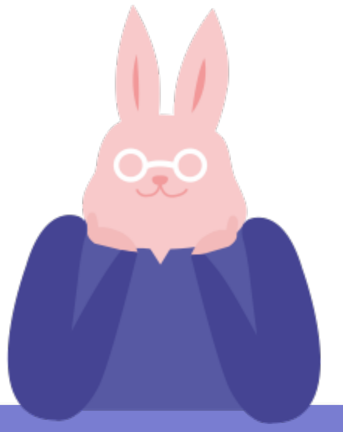
2	1	-2	5	-10	3	2	5	-3	7	9	-10



연속부분 최대합

- 우선 절반으로 나누어 각각을 구해보자

2	1	-2	5	-10	3	2	5	-3	7	9	-10



연속부분 최대합

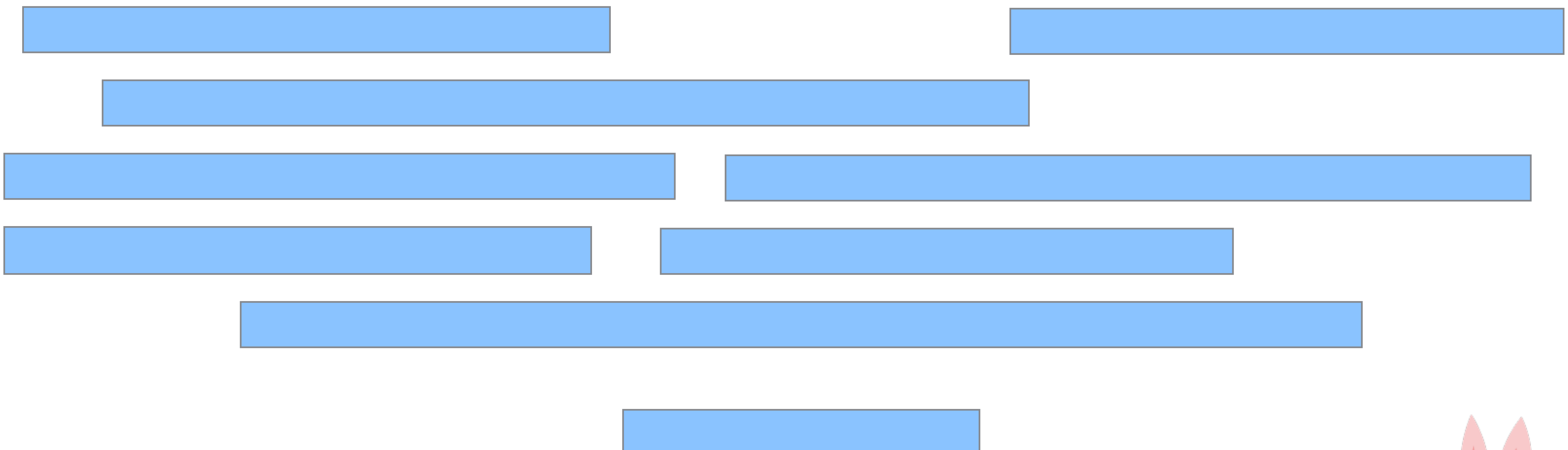
- 우선 절반으로 나누어 각각을 구해보자
  - 결과!

2	1	-2	5	-10	3	2	5	-3	7	9	-10

					3	3	2	2					
				-10	3	-7	7	2	5				
			5	-10	3	-2	4	2	5	-3			
		-2	5	-10	3	-4	11	2	5	-3	7		
	1	-2	5	-10	3	-3	**20**	2	5	-3	7	9	
2	1	-2	5	-10	3	-1	10	2	5	-3	7	9	-10

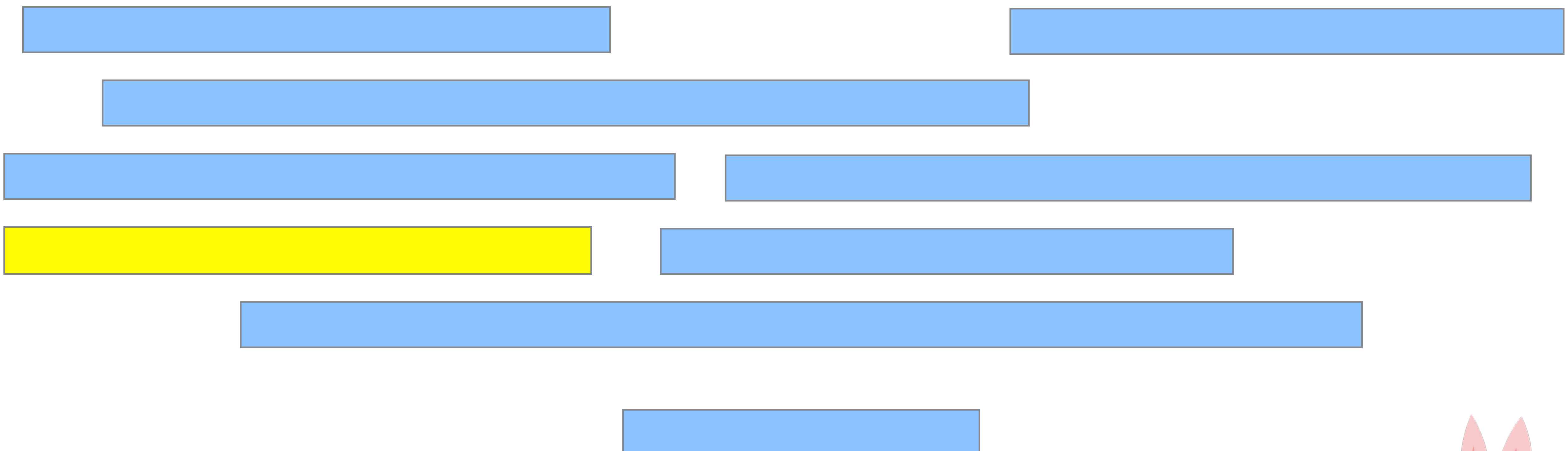
Interval Scheduling

- N개의 구간 중 겹치지 않는 구간을 최대한 많이 선택하라



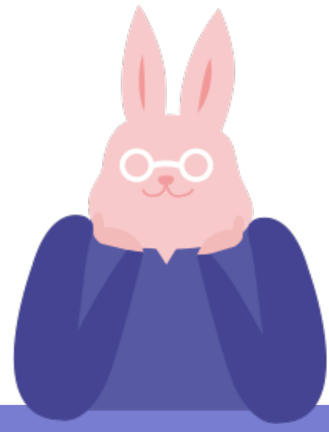
Interval Scheduling

- N개의 구간 중 겹치지 않는 구간을 최대한 많이 선택하라
  - 왼쪽부터 차례대로 선택한다고 하면, **빨리 끝나는 구간이 무조건 좋음 !**



Interval Scheduling

- N개의 구간 중 겹치지 않는 구간을 최대한 많이 선택하라
  - 왼쪽부터 차례대로 선택한다고 하면, **빨리 끝나는 구간이 무조건 좋음 !**

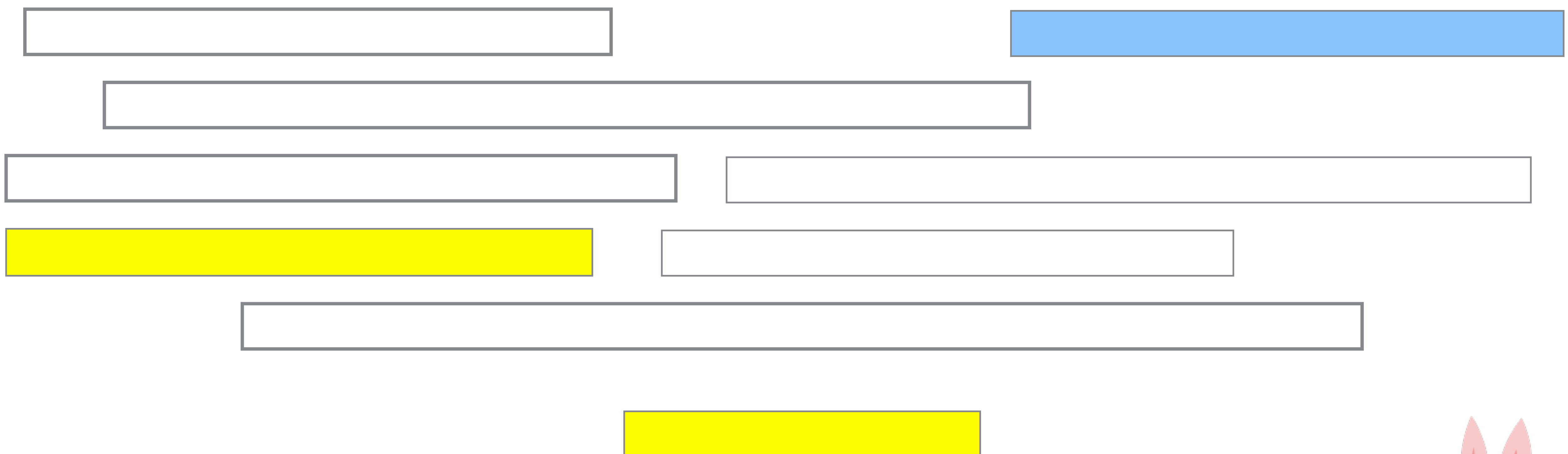


Interval Scheduling

- N개의 구간 중 겹치지 않는 구간을 최대한 많이 선택하라
  - 왼쪽부터 차례대로 선택한다고 하면, **빨리 끝나는 구간이 무조건 좋음 !**

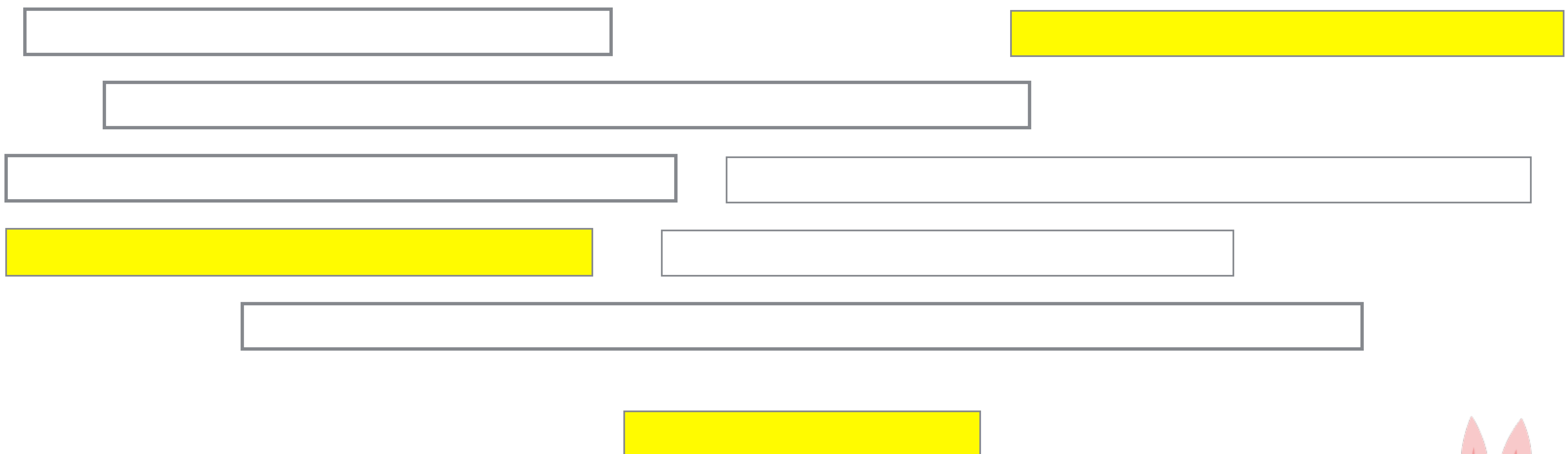
Interval Scheduling

- N개의 구간 중 겹치지 않는 구간을 최대한 많이 선택하라
  - 왼쪽부터 차례대로 선택한다고 하면, **빨리 끝나는 구간이 무조건 좋음 !**



Interval Scheduling

- N개의 구간 중 겹치지 않는 구간을 최대한 많이 선택하라
  - 왼쪽부터 차례대로 선택한다고 하면, **빨리 끝나는 구간이 무조건 좋음 !**

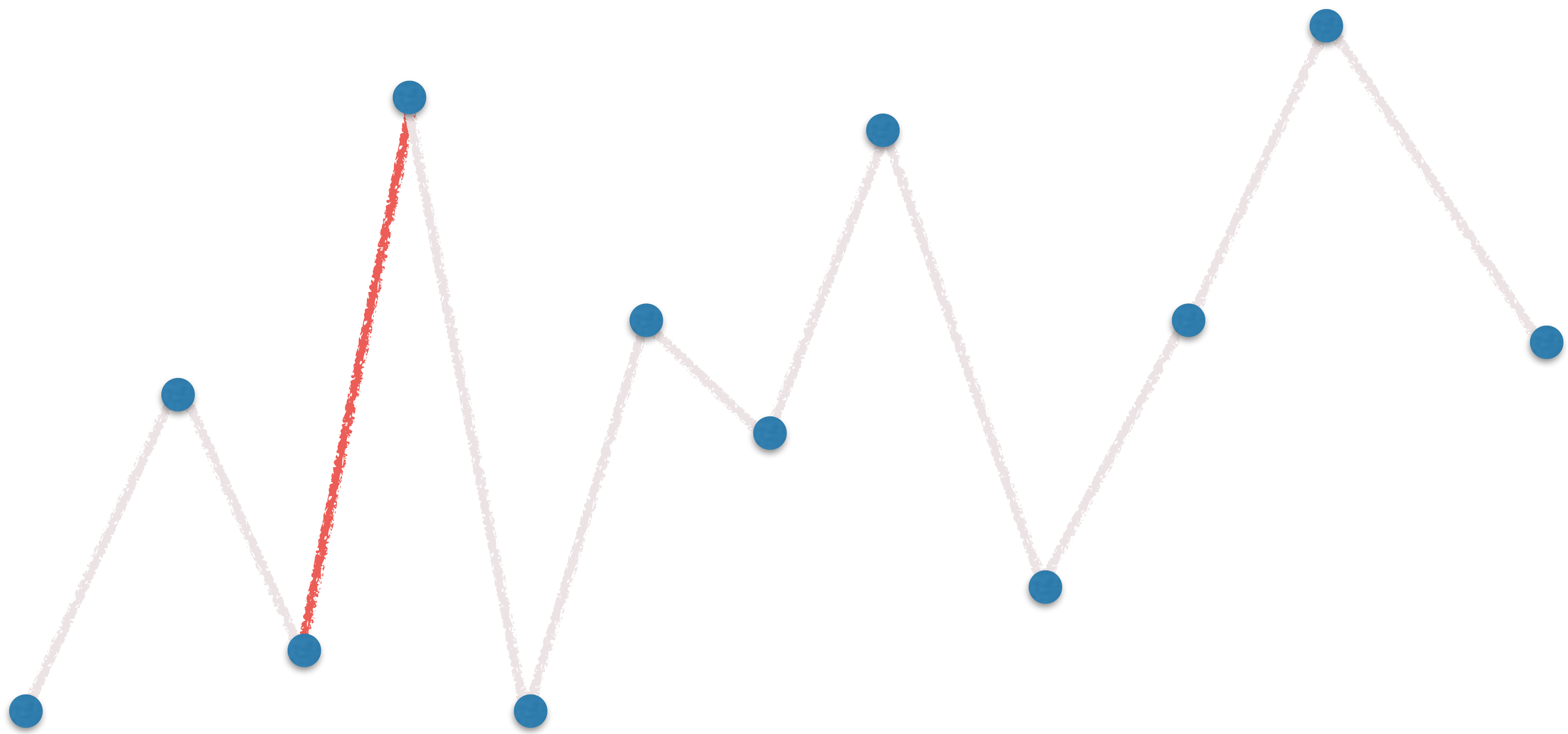


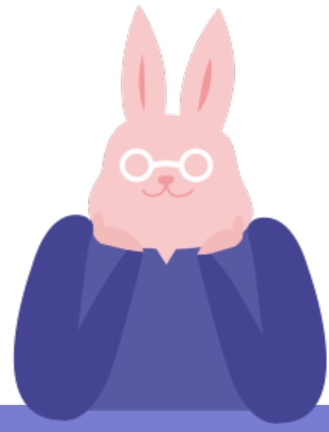
기울기의 최댓값 구하기

- N개의 점 중에서, 두 점을 이었을 때의 기울기의 최댓값은 ?

기울기의 최댓값 구하기

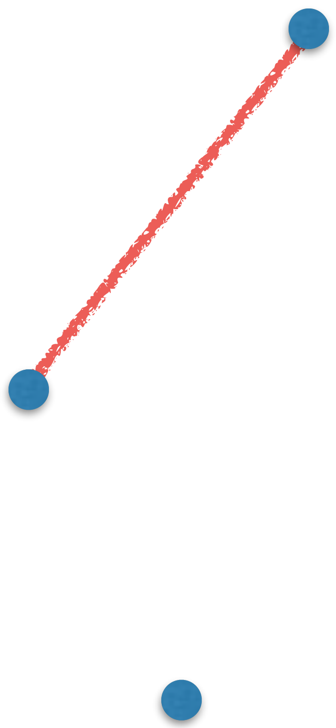
- x값으로 정렬 후, 인접한 두 점만 보면 충분하다





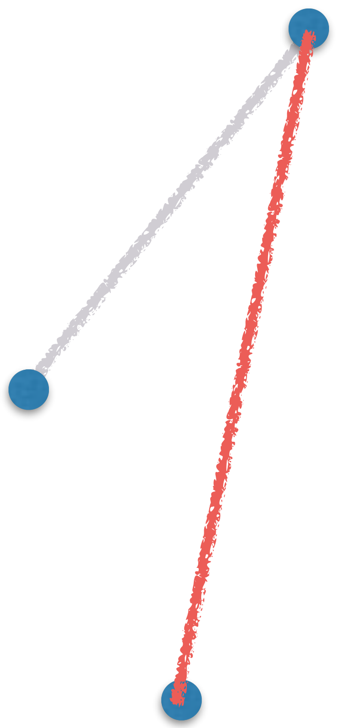
기울기의 최댓값 구하기

- 증명 : 귀류법
  - 인접하지 않은 두 점의 기울기가 최댓값이라 가정하자



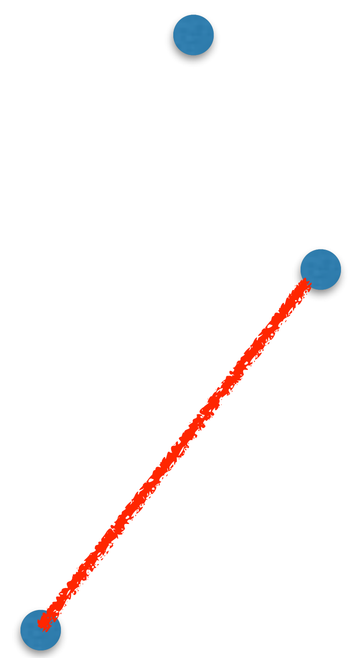
기울기의 최댓값 구하기

- 증명 : 귀류법
  - 인접하지 않은 두 점의 기울기가 최댓값이라 가정하자
  - 그럴 리 없음!



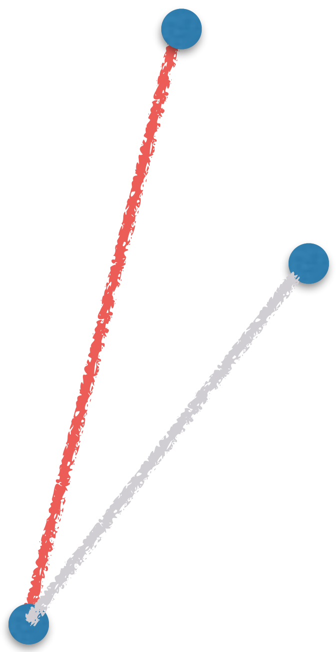
기울기의 최댓값 구하기

- 증명 : 귀류법
  - 인접하지 않은 두 점의 기울기가 최댓값이라 가정하자
  - 그럴 리 없음!



기울기의 최댓값 구하기

- 증명 : 귀류법
  - 인접하지 않은 두 점의 기울기가 최댓값이라 가정하자
  - 그럴 리 없음!



[활동문제 0] Merge sort

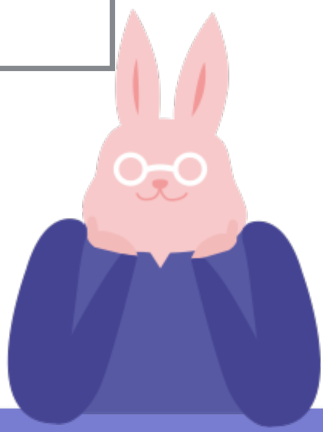
- 합병정렬 구현

입력의 예

```
10 9 8 7 6 5 4 3 2 1
```

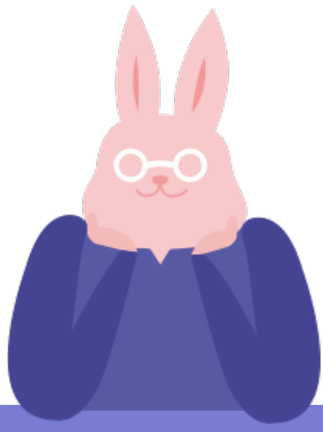
출력의 예

```
1 2 3 4 5 6 7 8 9 10
```



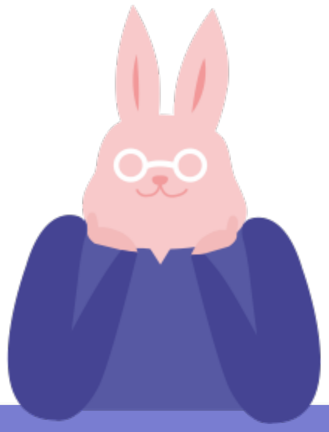
커리큘럼

1. 재귀호출, 추상화
2. 시간복잡도, 알고리즘 정확성 증명, 자료구조
3. 분할정복법, 탐욕적 기법
4. **<u>동적계획법 1</u>**
5. 동적계획법 2
6. 그래프 이론 1
7. 그래프 이론 2
8. 세계 여러 기업의 입사 인터뷰 문제 도전 (+ NP-Complete)



동적계획법 (Dynamic Programming)

- 부분문제를 푼 결과를 이용하여 전체문제를 푸는 방법
- 재귀호출 및 분할정복법과 느낌이 비슷합니다



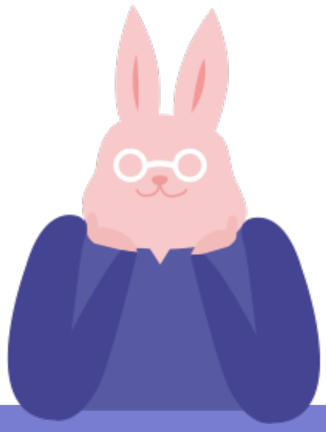
동적계획법 문제풀이 순서

1. Table을 정의한다

2. 점화식을 구한다

3. 어느 순서로 Table을 구해야 하는지를 생각한다

4. 답이 어디에 있는지를 찾는다



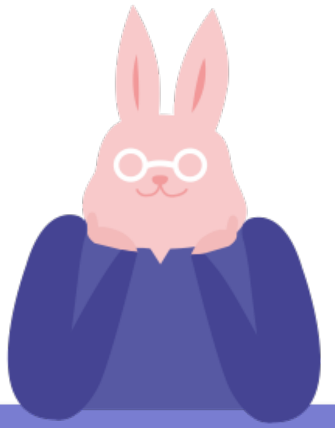
동적계획법 문제풀이 순서

1. <u>Table을 정의한다 : 모든 것을 결정한다</u>

2. 점화식을 구한다

3. 어느 순서로 Table을 구해야 하는지를 생각한다

4. 답이 어디에 있는지를 찾는다



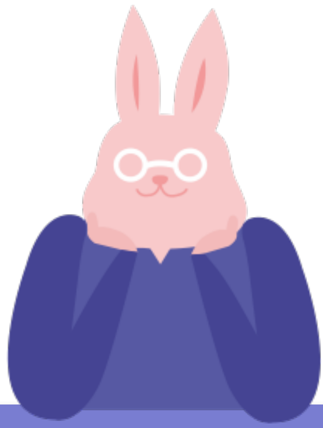
예제 : 팩토리얼 구하기

1. Table을 정의한다

2. 점화식을 구한다

3. 어느 순서로 Table을 구해야 하는지를 생각한다

4. 답이 어디에 있는지를 찾는다



예제 : 팩토리얼 구하기

1. Table을 정의한다
   - T(i) = i!

2. 점화식을 구한다

3. 어느 순서로 Table을 구해야 하는지를 생각한다

4. 답이 어디에 있는지를 찾는다

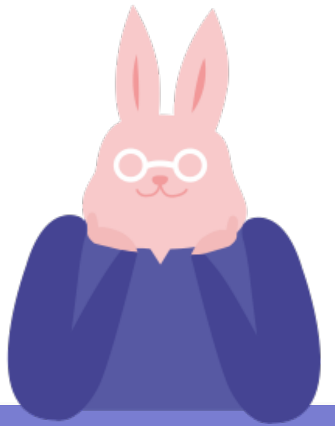
예제 : 팩토리얼 구하기

1. Table을 정의한다
   - $T(i) = i!$

2. 점화식을 구한다
   - $T(i) = T(i-1) * i$

3. 어느 순서로 Table을 구해야 하는지를 생각한다

4. 답이 어디에 있는지를 찾는다



예제 : 팩토리얼 구하기

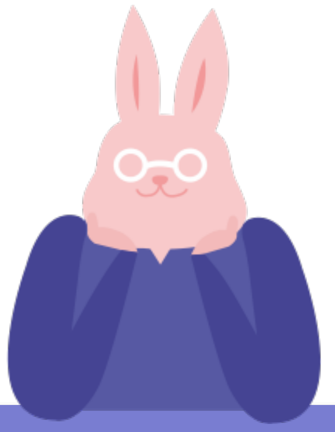
1. Table을 정의한다
   - T(i) = i!

0	1	2	3	4	5	6	7

2. 점화식을 구한다
   - T(i) = T(i-1) * i

3. 어느 순서로 Table을 구해야 하는지를 생각한다

4. 답이 어디에 있는지를 찾는다



예제 : 팩토리얼 구하기

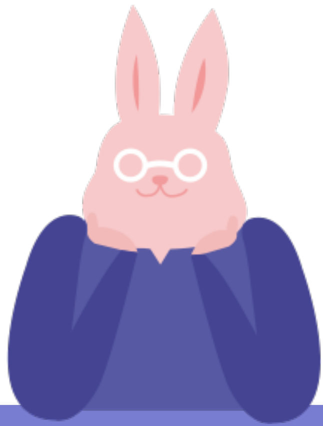
1. Table을 정의한다
   - T(i) = i!

0	1	2	3	4	5	6	7
1	1	2	6	24	120	720	5040

2. 점화식을 구한다
   - T(i) = T(i-1) * i

3. 어느 순서로 Table을 구해야 하는지를 생각한다

4. 답이 어디에 있는지를 찾는다



예제 : 팩토리얼 구하기

1. Table을 정의한다
   - $T(i) = i!$

2. 점화식을 구한다
   - $T(i) = T(i-1) * i$

3. 어느 순서로 Table을 구해야 하는지를 생각한다
   - i=0 —> i=n

4. 답이 어디에 있는지를 찾는다

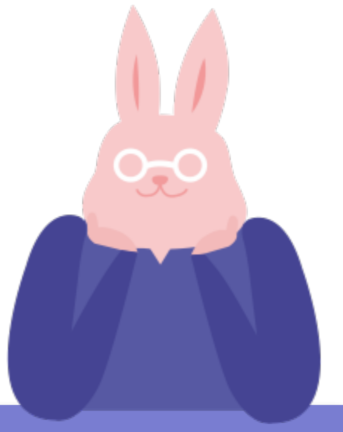
예제 : 팩토리얼 구하기

1. Table을 정의한다
   - T(i) = i!

2. 점화식을 구한다
   - T(i) = T(i-1) * i

3. 어느 순서로 Table을 구해야 하는지를 생각한다
   - i=0 —> i=n

4. 답이 어디에 있는지를 찾는다
   - T(n)



예제 : 피보나치 수 구하기

1. Table을 정의한다
2. 점화식을 구한다
3. 어느 순서로 Table을 구해야 하는지를 생각한다
4. 답이 어디에 있는지를 찾는다

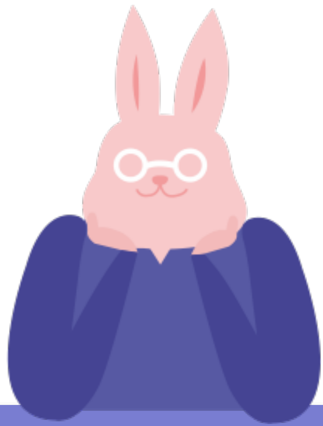
예제 : 피보나치 수 구하기

1. Table을 정의한다
   - T(i) = i번째 피보나치 수

2. 점화식을 구한다
   - T(i) = T(i-1) + T(i-2)

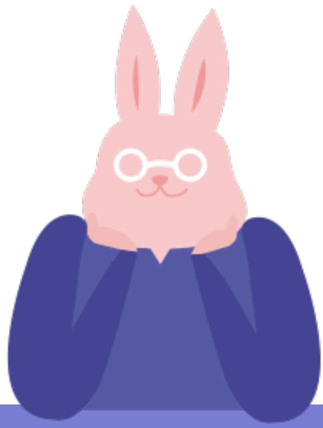
3. 어느 순서로 Table을 구해야 하는지를 생각한다
   - i=0 —> i=n

4. 답이 어디에 있는지를 찾는다
   - T(n)



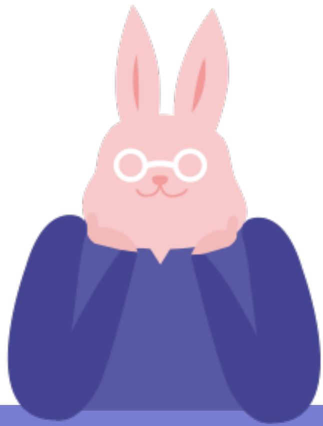
동적계획법을 향한 저의 마음

- 재귀호출만큼 컴퓨터 공학의 꽃이고
- 컴퓨터공학적 사고력을 높여주고
- 문제 풀이가 굉장히 간단하고 명료하며
- 코딩 또한 (이전에 우리가 했던 것에 비하여) 굉장히 간단하고
- 개인적으로 너무 좋아하며
- 삶의 질이 높아지고
- 익숙해 지고 나면 또 별게 아니며
- 아는척을 할 수 있고
- …



동적계획법 연습

- 무조건 정의를 많이 보고, 많이 풀어보는게 장땡
- 비슷한 패턴이 꽤 많습니다
- 오늘 **9개**의 예제를 볼겁니다



[연습문제] 최대구간의 합 구하기

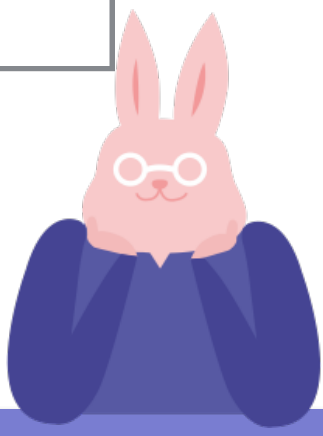
- n개의 숫자 중에서 연속 부분 최대합을 출력

입력의 예

```
1 2 3 4 -100 1
```

출력의 예

```
10
```



[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **i번째 수를 끝으로 하는** 최대 구간의 합

T(3) ?

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T												

[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **i번째 수를 끝으로 하는** 최대 구간의 합

T(3) ?

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T												

[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **<u>i번째 수를 끝으로 하는</u>** 최대 구간의 합

T(8) ?

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T												

[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **i번째 수를 끝으로 하는** 최대 구간의 합

T(8) ?

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T												

[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **<u>i번째 수를 끝으로 하는</u>** 최대 구간의 합

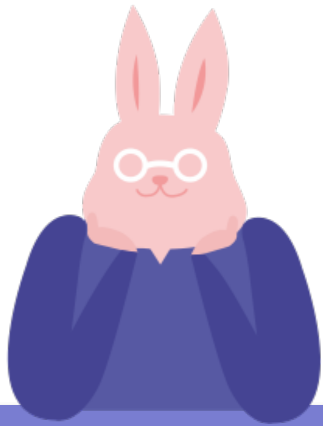
data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13

[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **<u>i번째 수를 끝으로 하는</u>** 최대 구간의 합

2. 점화식을 구한다

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13



[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **<u>i번째 수를 끝으로 하는</u>** 최대 구간의 합

2. 점화식을 구한다

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13

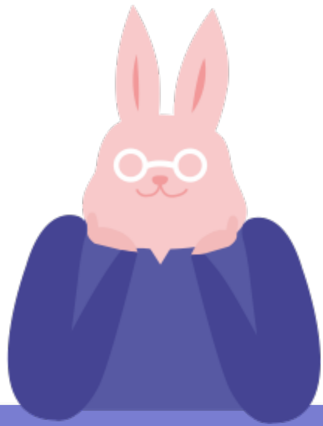
[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **i번째 수를** [illegible] 최대 구간의 합

2. 점화식을 구한다

						2
					3	2
				-10	3	2
			5	-10	3	2
		-2	5	-10	3	2
	1	-2	5	-10	3	2
2	1	-2	5	-10	3	2

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13



[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **i번째 수를** 최대 구간의 합

2. 점화식을 구하

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13

[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = <u>**i번째 수를 끝으로 하는**</u> 최대 구간의 합

2. 점화식을 구한다

	2
3	2

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13

[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = <u>**i번째 수를 끝으로 하는**</u> 최대 구간의 합

2. 점화식을 구한다

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13

[연습문제] 최대구간 합 구하기

1. Table을 정의한다
   - T(i) = **i번째 수를 끝으**
2. 점화식을 구한

										7		
									-3	7		
								5	-3	7		
							2	5	-3	7		
						3	2	5	-3	7		
					-10	3	2	5	-3	7		
				5	-10	3	2	5	-3	7		
			-2	5	-10	3	2	5	-3	7		
		1	-2	5	-10	3	2	5	-3	7		
	2	1	-2	5	-10	3	2	5	-3	7		
data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13

[연습문제] 최대구간 합 구하기

1. Table을 정의한다
   - T(i) = **i번째 수를 끝으**

2. 점화식을 구한다

									7
								-3	7
							5	-3	7
						2	5	-3	7
					3	2	5	-3	7
				-10	3	2	5	-3	7
			5	-10	3	2	5	-3	7
		-2	5	-10	3	2	5	-3	7
	1	-2	5	-10	3	2	5	-3	7
2	1	-2	5	-10	3	2	5	-3	7

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13

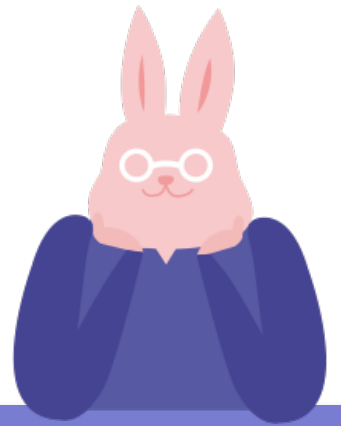
[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **<u>i번째 수를 끝으로 하는</u>** 최대 구간의 합

2. 점화식을 구한다

	7
7	7

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13

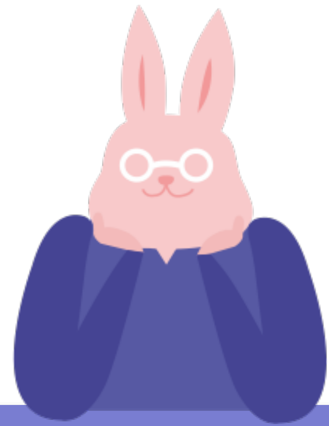


[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = <u>**i번째 수를 끝으로 하는**</u> 최대 구간의 합

2. 점화식을 구한다
   - T(i) = max(data[i], T(i-1) + data[i])

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13



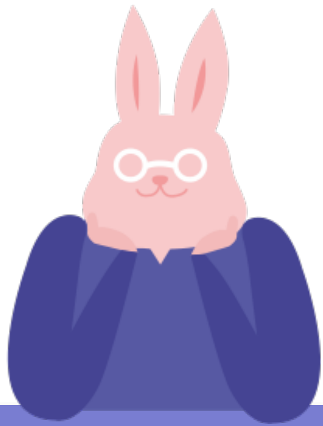
[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = <u>**i번째 수를 끝으로 하는**</u> 최대 구간의 합

2. 점화식을 구한다
   - T(i) = max(data[i], T(i-1) + data[i])

3. 어느 순서로 Table을 구해야 하는지를 생각한다

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13



[연습문제] 최대구간의 합 구하기

1. Table을 정의한다
   - T(i) = **<u>i번째 수를 끝으로 하는</u>** 최대 구간의 합

2. 점화식을 구한다
   - T(i) = max(data[i], T(i-1) + data[i])

3. 어느 순서로 Table을 구해야 하는지를 생각한다
   - i=0 —> i=n

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13

[연습문제] 최대구간의 합 구하기

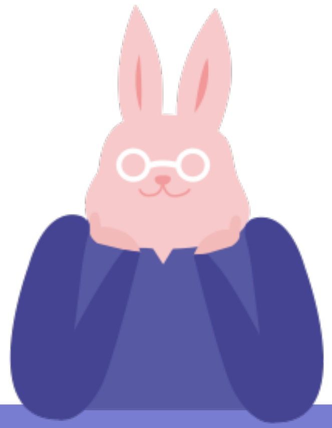
4. 답은 어디에 있는지를 찾는다

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13

[연습문제] 최대구간의 합 구하기

4. 답은 어디에 있는지를 찾는다
   - max(T(i))

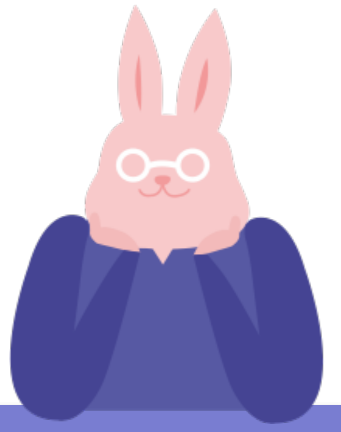
data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13



[연습문제] 최대구간의 합 구하기

- 시간복잡도는?

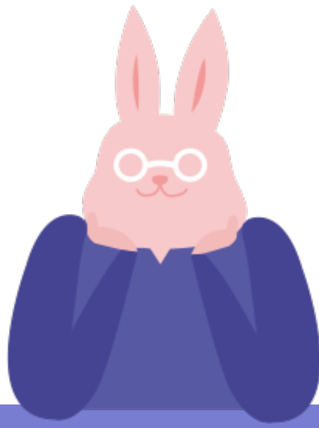
data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13



[연습문제] 최대구간의 합 구하기

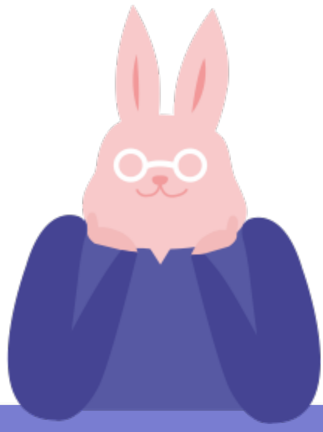
- 시간복잡도는?
  - O(n)

data	2	1	-2	5	-10	3	2	5	-3	7	9	-10
T	2	3	1	6	-4	3	5	10	7	14	23	13



[연습문제] 최대구간의 합 구하기

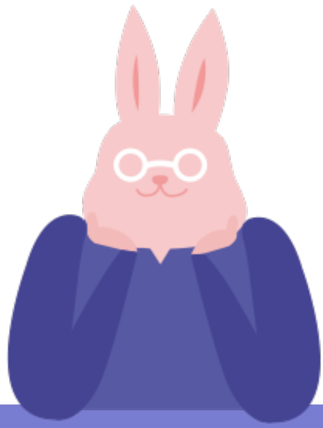
```
T = [ 0 for i in range(n) ]

T[0] = data[0]
result = data[0]

for i in range(n) :
    T[i] = max(T[i-1] + data[i], data[i])
    result = max(result, T[i])

return result
```



[연습문제] 계단 오르기

- n칸의 계단이 있고, 한 번에 최대 3칸까지 오를 수 있다.
n칸을 오르는 경우의 수는 ?

입력의 예	출력의 예
3	7
5	13

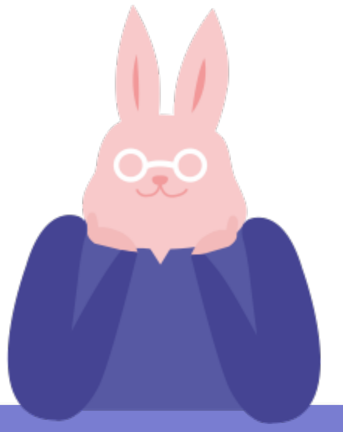


[연습문제] 계단 오르기

1. Table을 정의한다

[연습문제] 계단 오르기

1. Table을 정의한다
   - $T(i)$ = 계단 i칸을 오를 때의 경우의 수



[연습문제] 계단 오르기

1. Table을 정의한다
   - T(i) = 계단 i칸을 오를 때의 경우의 수

2. 점화식을 구한다

[연습문제] 계단 오르기

1. Table을 정의한다
   - T(i) = 계단 i칸을 오를 때의 경우의 수

2. 점화식을 구한다
   - 계단 i칸을 오르는 경우
     - 가장 마지막에 1칸을 오르는 경우
     - 가장 마지막에 2칸을 오르는 경우
     - 가장 마지막에 3칸을 오르는 경우

1 1 1 1 1
1 1 2 1
1 2 1 1
2 1 1 1
1 3 1
3 1 1
4 1
1 1 1 2
1 2 2
2 1 2
3 2
1 1 3
2 3

[연습문제] 계단 오르기

1. Table을 정의한다
   - T(i) = 계단 i칸을 오를 때의 경우의 수

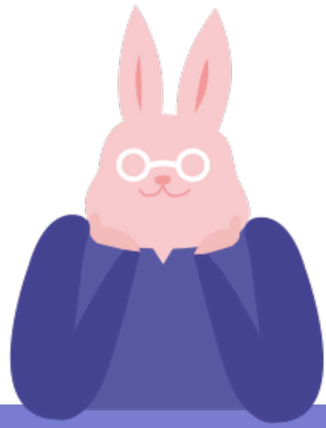
2. 점화식을 구한다
   - T(i) = T(i-1) + T(i-2) + T(i-3)

[연습문제] 계단 오르기

1. Table을 정의한다
   - T(i) = 계단 i칸을 오를 때의 경우의 수

2. 점화식을 구한다
   - T(i) = T(i-1) + T(i-2) + T(i-3)

3. 어느 순서로 Table을 구해야 하는지를 생각한다
   - i=0 —> i=n
4. 답이 어디에 있는지를 찾는다
   - T(n)



[연습문제] 포도주 마시기

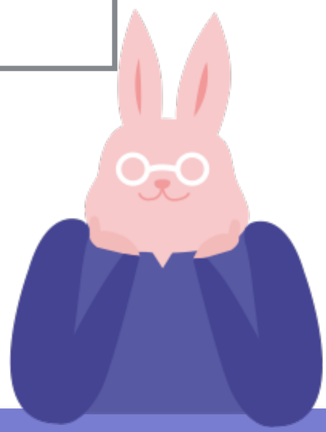
- n잔의 포도주가 있을 때, 마시는 포도주의 양을 최대화 하라
  단, 연속하여 3잔을 모두 마실 수는 없다

입력의 예

```
6
6 10 13 9 8 1
```

출력의 예

```
33
```



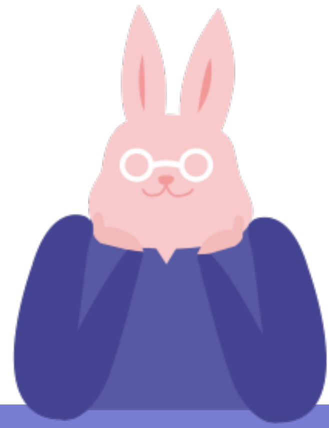
[연습문제] 포도주 마시기

1. Table을 정의한다

[연습문제] 포도주 마시기

1. Table을 정의한다
   - T(i) = 1 ~ i 까지의 포도주가 있고, i번째 포도주를 마실 때,
     마시는 양의 최댓값

data	6	10	13	9	8	1
T						



[연습문제] 포도주 마시기

1. Table을 정의한다
   - T(i) = 1 ~ i 까지의 포도주가 있고, i번째 포도주를 마실 때, 마시는 양의 최댓값

data	6	10	13	9	8	1
T	6	10	23	28	33	29

[연습문제] 포도주 마시기

1. Table을 정의한다
   - T(i) = 1 ~ i 까지의 포도주가 있고, i번째 포도주를 마실 때,
     마시는 양의 최댓값

2. 점화식을 구한다

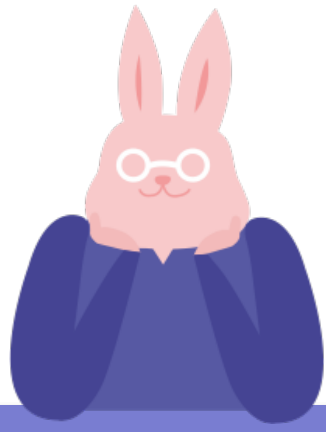
data	6	10	13	9	8	1
T	6	10	23	28	33	29

[연습문제] 포도주 마시기

1. Table을 정의한다
   - T(i) = 1 ~ i 까지의 포도주가 있고, i번째 포도주를 마실 때,
     마시는 양의 최댓값

2. 점화식을 구한다

data	6	10	13	9	8	1
T	6	10	23	28	33	29



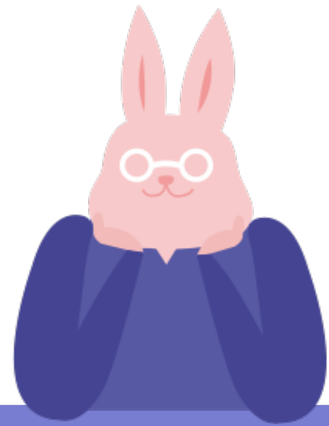
[연습문제] 포도주 마시기

1. Table을 정의한다
   - T(i) = 1 ~ i 까지의 포도주가 있고, i번째 포도주를 마실 때, 마시는 양의 최댓값

2. 점화식을 구한다

	X	13	9		
		X	9		

data	6	10	13	9	8	1
T	6	10	23	28	33	29



[연습문제] 포도주 마시기

1. Table을 정의한다
   - T(i) = 1 ~ i 까지의 포도주가 있고, i번째 포도주를 마실 때,
     마시는 양의 최댓값

2. 점화식을 구한다
   - T(i) = max(T(i-3) + data(i) + data(i-1), T(i-2) + data(i))

data	6	10	13	9	8	1
T	6	10	23	28	33	29

[연습문제] 포도주 마시기

3. 어느 순서로 Table을 구해야 하는지를 생각한다
   - i=1 —> i=n

4. 답은 어디에 있는지를 찾는다
   - max(T(i))

data	6	10	13	9	8	1
T	6	10	23	28	33	29

[연습문제] 특별한 이진수

- 길이가 n인 특별한 이진수의 개수를 구하여라
  특별한 이진수 : 1로 시작하며, 1이 두 번 연속으로 나타나지 않는 이진수

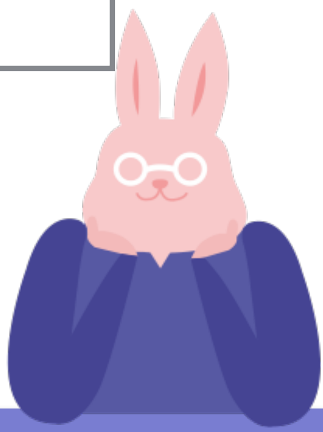
입력의 예

```
4
```

출력의 예

```
3
```

1000 1001 1010

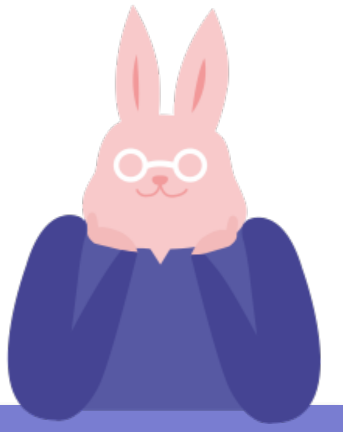


[연습문제] 특별한 이진수

1. Table을 정의한다

[연습문제] 특별한 이진수

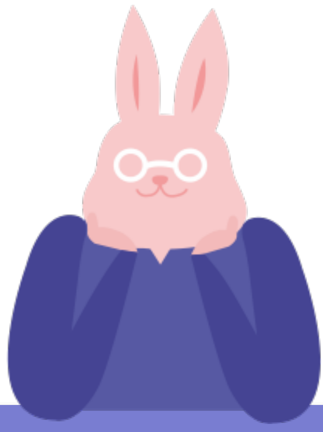
1. Table을 정의한다
   - T(i, 0) = 길이가 i이고, 맨 끝자리가 0인 특별한 이진수의 개수
     T(i, 1) = 길이가 i이고, 맨 끝자리가 1인 특별한 이진수의 개수



[연습문제] 특별한 이진수

1. Table을 정의한다
   - T(i, 0) = 길이가 i이고, 맨 끝자리가 0인 특별한 이진수의 개수
     T(i, 1) = 길이가 i이고, 맨 끝자리가 1인 특별한 이진수의 개수

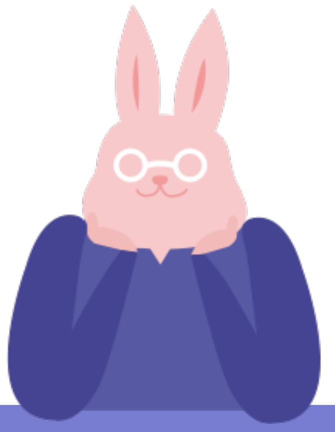
2. 점화식을 구한다



[연습문제] 특별한 이진수

1. Table을 정의한다
   - T(i, 0) = 길이가 i이고, 맨 끝자리가 0인 특별한 이진수의 개수
     T(i, 1) = 길이가 i이고, 맨 끝자리가 1인 특별한 이진수의 개수

2. 점화식을 구한다
   - T(i, 0) = T(i-1, 0) + T(i-1, 1)
     T(i, 1) = T(i-1, 0)

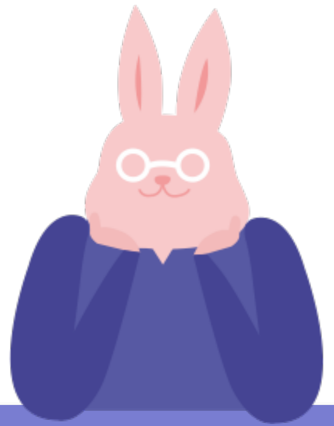


[연습문제] 특별한 이진수

1. Table을 정의한다
   - T(i, 0) = 길이가 i이고, 맨 끝자리가 0인 특별한 이진수의 개수
     T(i, 1) = 길이가 i이고, 맨 끝자리가 1인 특별한 이진수의 개수

2. 점화식을 구한다
   - T(i, 0) = T(i-1, 0) + T(i-1, 1)
     T(i, 1) = T(i-1, 0)

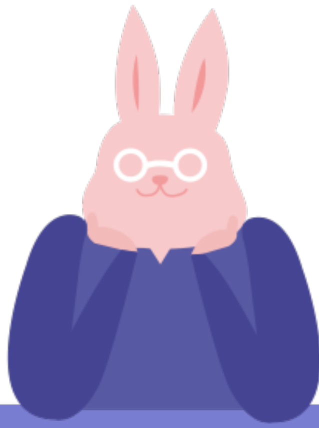
3. 어느 순서로 구해야 할지를 생각한다
   - i=0 —> i=n



[연습문제] 특별한 이진수

4. 답이 어디에 있는지를 찾는다
   - T(n, 0) + T(n, 1)

5. 시간복잡도
   - O(n)



[연습문제] R개를 고르는 경우의 수

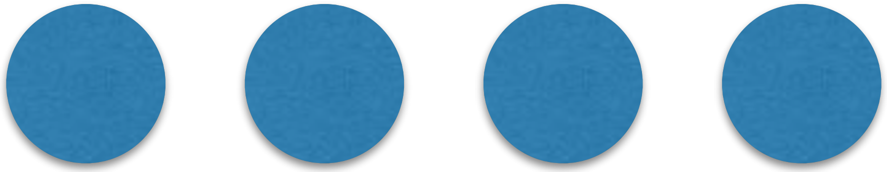
- N개 중에서 R개를 고르는 경우의 수를 구하여라

입력의 예

```
4 2
```

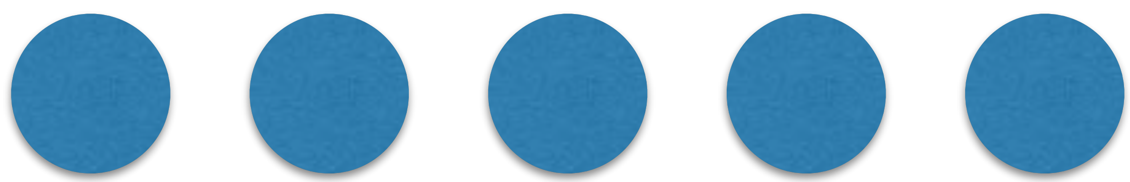
출력의 예

```
6
```



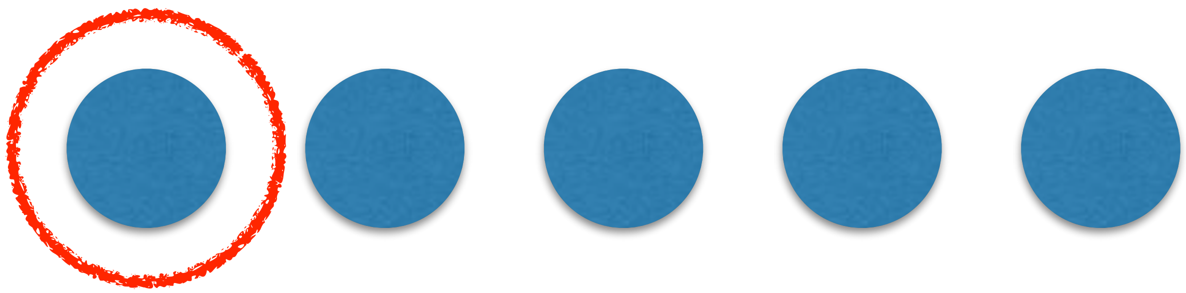
[연습문제] R개를 고르는 경우의 수

1. Table을 정의한다
   - T(i, j) = i개 중에서 j개를 고르는 경우의 수



[연습문제] R개를 고르는 경우의 수

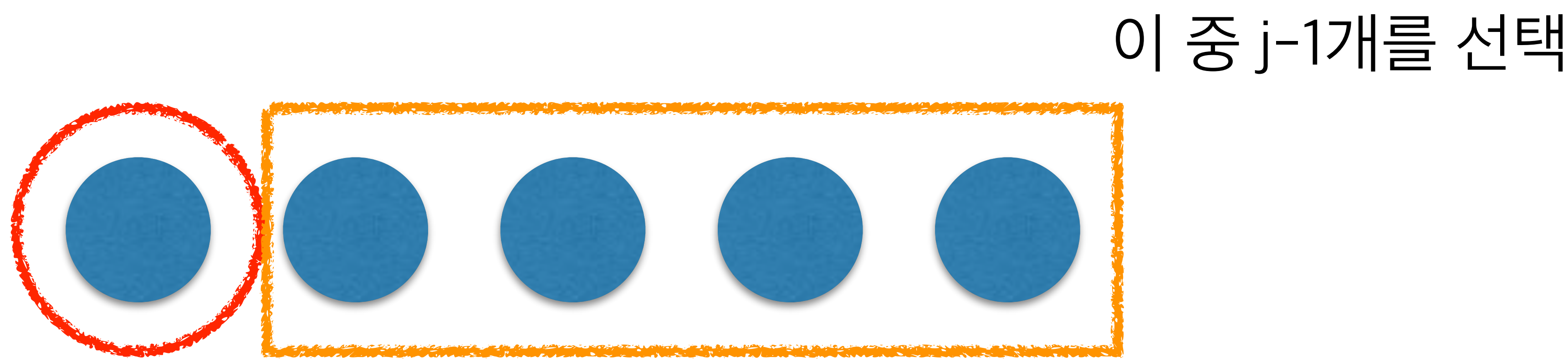
1. Table을 정의한다
   - T(i, j) = i개 중에서 j개를 고르는 경우의 수



첫 번째 공을 고르는 경우와 고르지 않는 경우가 있다

[연습문제] R개를 고르는 경우의 수

1. Table을 정의한다
   - T(i, j) = i개 중에서 j개를 고르는 경우의 수



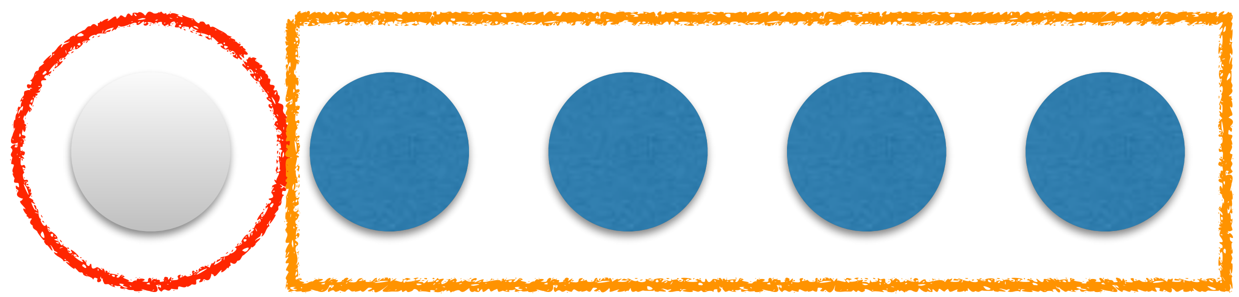
첫 번째 공을 고르는 경우와 고르지 않는 경우가 있다

고르는 경우 : T(i-1, j-1)

[연습문제] R개를 고르는 경우의 수

1. Table을 정의한다
   - T(i, j) = i개 중에서 j개를 고르는 경우의 수

이 중 j개를 선택



첫 번째 공을 고르는 경우와 고르지 않는 경우가 있다

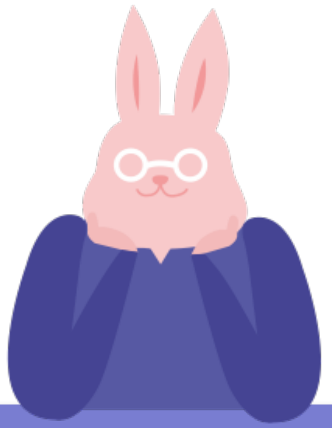
고르는 경우 : T(i-1, j-1)
고르지 않는경우 : T(i-1, j)

[연습문제] R개를 고르는 경우의 수

1. Table을 정의한다
   - T(i, j) = i개 중에서 j개를 고르는 경우의 수

2. 점화식을 구한다
   - T(i, j) = T(i-1, j) + T(i-1, j-1)

3. 어느 순서로 Table을 구해야 하는지를 생각한다

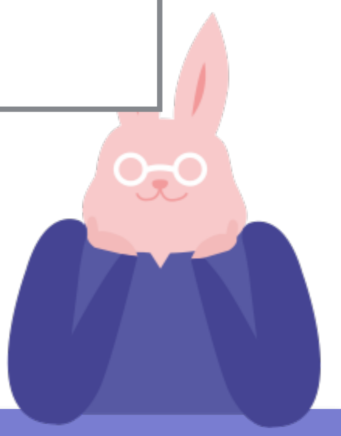


[연습문제] R개를 고르는 경우의 수

1. Table을 정의한다
   - T(i, j) = i개 중에서 j개를 고르는 경우

2. 점화식을 구한다
   - T(i, j) = T(i-1, j) + T(i-1, j-1)

3. 어느 순서로 Table을 구해야 하는지를

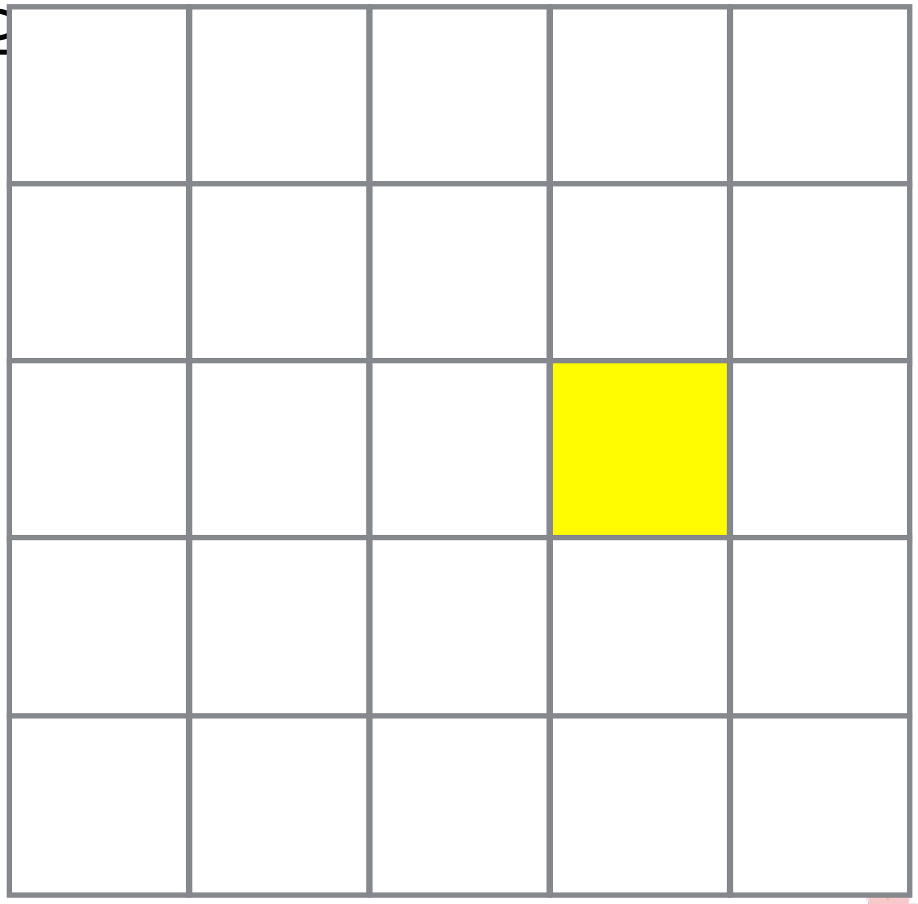


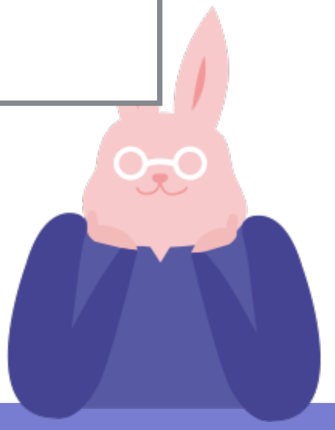
[연습문제] R개를 고르는 경우의 수

1. Table을 정의한다
   - T(i, j) = i개 중에서 j개를 고르는 경우

2. 점화식을 구한다
   - T(i, j) = T(i-1, j) + T(i-1, j-1)

3. 어느 순서로 Table을 구해야 하는지를



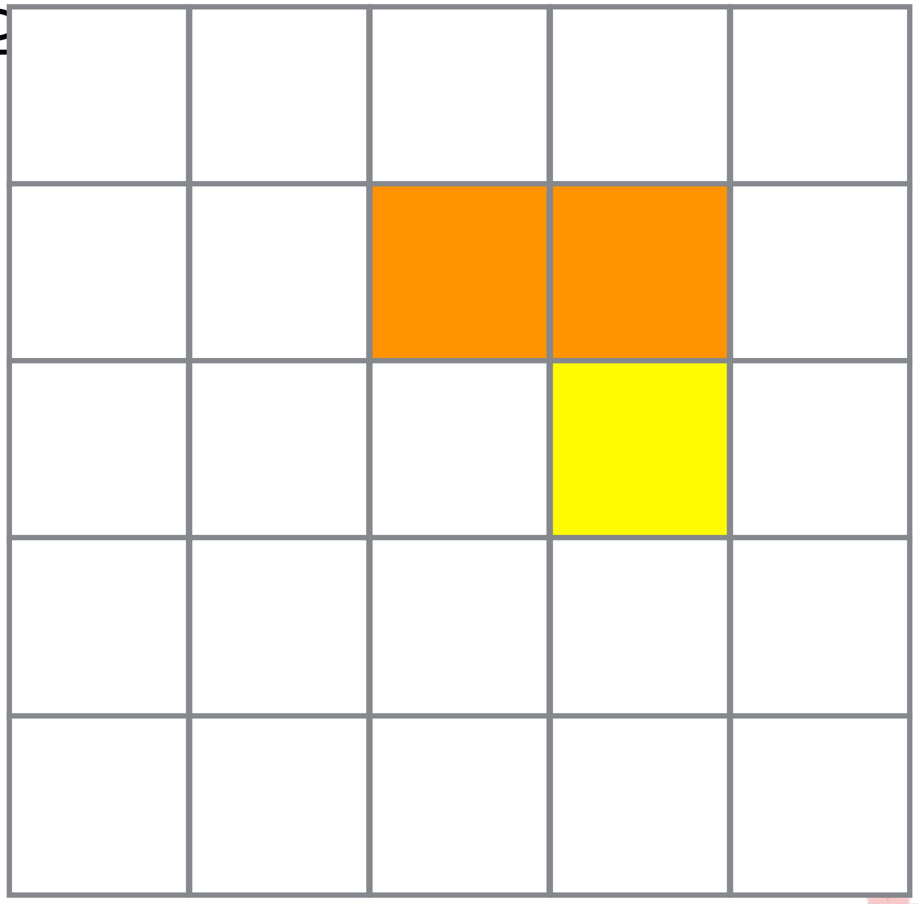


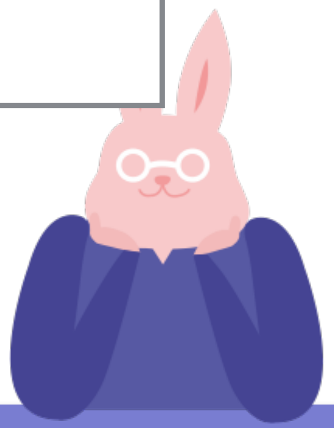
[연습문제] R개를 고르는 경우의 수

1. Table을 정의한다
   - T(i, j) = i개 중에서 j개를 고르는 경우

2. 점화식을 구한다
   - T(i, j) = T(i-1, j) + T(i-1, j-1)

3. 어느 순서로 Table을 구해야 하는지를



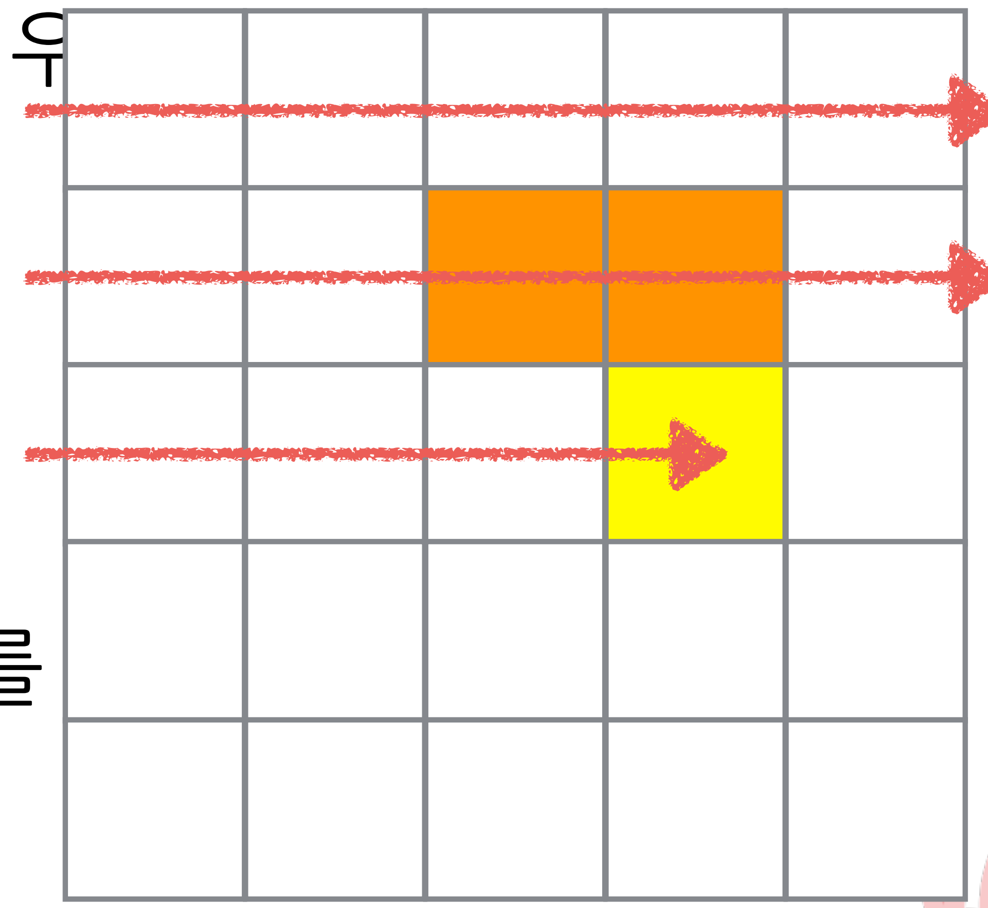


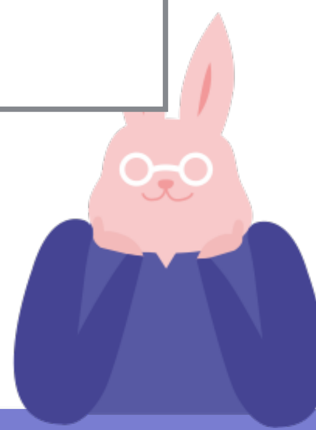
[연습문제] R개를 고르는 경우의 수

1. Table을 정의한다
   - T(i, j) = i개 중에서 j개를 고르는 경우

2. 점화식을 구한다
   - T(i, j) = T(i-1, j) + T(i-1, j-1)

3. 어느 순서로 Table을 구해야 하는지를

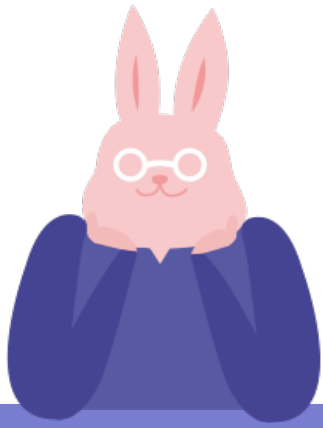




[연습문제] R개를 고르는 경우의 수

4. 답이 어디에 있는지를 찾는다
   - T(n, r)

5. 시간복잡도
   - O(n^2)



[연습문제] 짜장, 짬뽕, 볶음밥

- 매일 짜장, 짬뽕, 볶음밥의 선호도가 다르며, 전날 먹은건 오늘 먹지 않는다. 만족도를 최대화 하라.

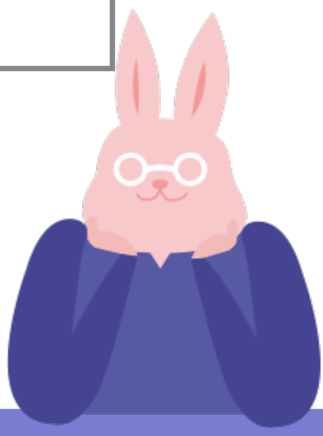
입력의 예

```
3
27 8 35
18 36 10
7 22 45
```

출력의 예

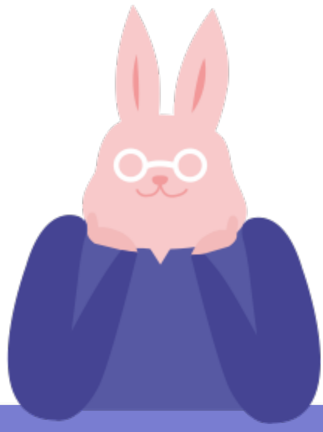
```
116
```

1000 1001 1010



[연습문제] 짜장, 짬뽕, 볶음밥

1. Table을 정의한다
   - T(i, 0) = i번째 날까지 밥을 먹으며, i번째 날에 짜장을 먹을 경우 최대 만족도
   - T(i, 1) = i번째 날까지 밥을 먹으며, i번째 날에 짬뽕을 먹을 경우 최대 만족도
   - T(i, 2) = i번째 날까지 밥을 먹으며, i번째 날에 볶음밥을 먹을 경우 최대 만족도



[연습문제] 짜장, 짬뽕, 볶음밥

1. Table을 정의한다
   - T(i, 0) = i번째 날까지 밥을 먹으며, i번째 날에 짜장을 먹을 경우 최대 만족도
   - T(i, 1) = i번째 날까지 밥을 먹으며, i번째 날에 짬뽕을 먹을 경우 최대 만족도
   - T(i, 2) = i번째 날까지 밥을 먹으며, i번째 날에 볶음밥을 먹을 경우 최대 만족도

2. 점화식을 구한다

[연습문제] 짜장, 짬뽕, 볶음밥

1. Table을 정의한다
   - T(i, 0) = i번째 날까지 밥을 먹으며, i번째 날에 짜장을 먹을 경우 최대 만족도
   - T(i, 1) = i번째 날까지 밥을 먹으며, i번째 날에 짬뽕을 먹을 경우 최대 만족도
   - T(i, 2) = i번째 날까지 밥을 먹으며, i번째 날에 볶음밥을 먹을 경우 최대 만족도

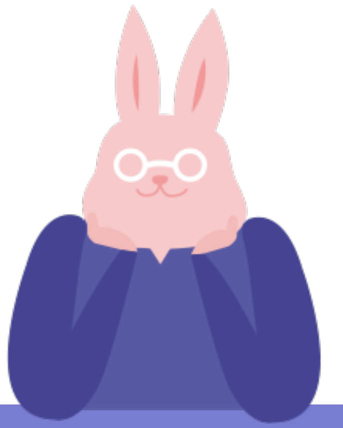
2. 점화식을 구한다
   - T(i, 0) = max(T(i-1, 1), T(i-1, 2)) + data(i, 0)
   - T(i, 1) = max(T(i-1, 0), T(i-1, 2)) + data(i, 1)
   - T(i, 2) = max(T(i-1, 0), T(i-1, 1)) + data(i, 2)

[연습문제] 짜장, 짬뽕, 볶음밥

3. 어느 순서로 Table을 구해야 하는지를 생각한다
   - i=1 —> i=n

4. 답은 어디에 있는지를 찾는다
   - max(T(i))

5. 시간복잡도
   - O(n)



[활동문제 1] 최장 증가 부분 수열

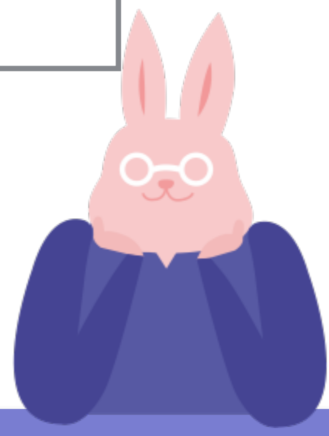
- N개의 숫자 중 최장 증가 부분 수열을 구하여라

입력의 예

```
5
1 4 2 3 5
```

출력의 예

```
4
```



[활동문제 1] 최장 증가 부분 수열

1. Table을 정의한다

[활동문제 1] 최장 증가 부분 수열

1. Table을 정의한다
   - T(i) = i번째 숫자를 끝으로 하는 최장 증가 부분 수열의 길이

[활동문제 1] 최장 증가 부분 수열

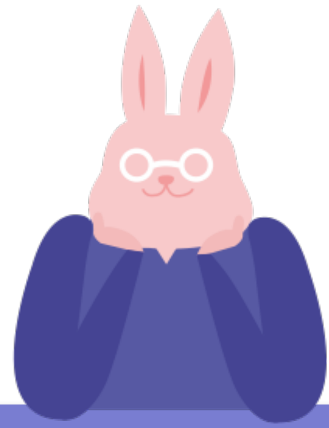
1. Table을 정의한다
   - T(i) = i번째 숫자를 끝으로 하는 최장 증가 부분 수열의 길이

data	5	2	8	6	3	6	9	7
T								

[활동문제 1] 최장 증가 부분 수열

1. Table을 정의한다
   - T(i) = i번째 숫자를 끝으로 하는 최장 증가 부분 수열의 길이

data	5	2	8	6	3	6	9	7
T	1	1	2	2	2	3	4	4



[활동문제 1] 최장 증가 부분 수열

1. Table을 정의한다
   - T(i) = i번째 숫자를 끝으로 하는 최장 증가 부분 수열의 길이

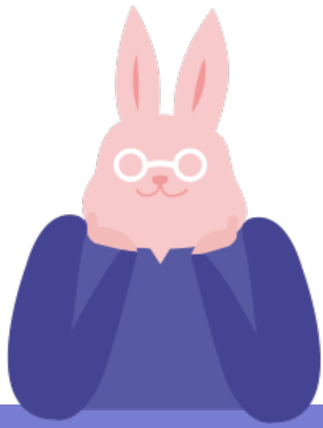
data	5	2	8	6	3	6	9	7
T	1	1	2	2	2	3	4	4

[활동문제 1] 최장 증가 부분 수열

1. Table을 정의한다
   - T(i) = i번째 숫자를 끝으로 하는 최장 증가 부분 수열의 길이

2. 점화식을 구한다

data	5	2	8	6	3	6	9	7
T	1	1	2	2	2	3	4	4



[활동문제 1] 최장 증가 부분 수열

1. Table을 정의한다
   - $T(i)$ = i번째 숫자를 끝으로 하는 최장 증가 부분 수열의 길이

2. 점화식을 구한다
   - $T(i) = \max(T(j) + 1) \quad \text{if } data[j] < data[i]$

data	5	2	8	6	3	6	9	7
T	1	1	2	2	2	3	4	4

[활동문제 1] 최장 증가 부분 수열

3. 어느 순서로 Table을 구해야 하는지를 생각한다
   - i=1 —> i=n

4. 답은 어디에 있는지를 찾는다
   - max(T(i))

5. 시간복잡도
   - O(n^2)

[실습문제 3-1] 최대 공통 부분 수열

- 두 문자열의 최대 공통 부분 수열의 길이를 구하여라

입력의 예

```
television
telephone
```

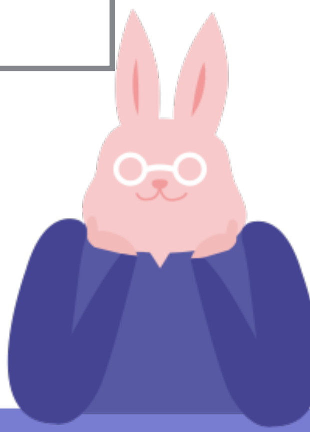
출력의 예

```
6
```

televisi**on**

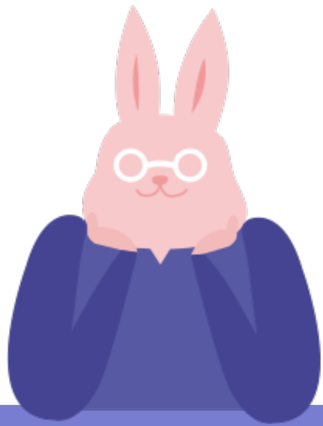
teleph**on**e

telex



[실습문제 3-1] 최대 공통 부분 수열

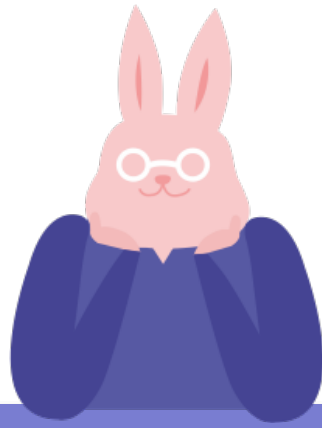
1. Table을 정의한다
   - T(i, j) = str1의 1 ~ i, str2의 1 ~ j의 최대공통부분수열 길이



[실습문제 3-1] 최대 공통 부분 수열

1. Table을 정의한다
   - T(i, j) = str1의 1 ~ i, str2의 1 ~ j의 최대공통부분수열 길이

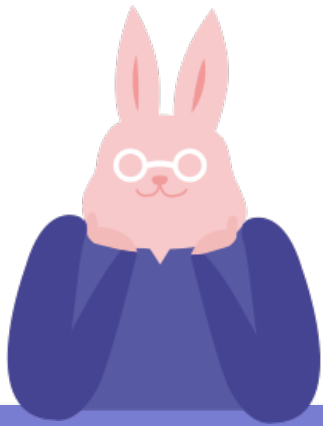
	S	N	O	W	Y
S					
U					
N					
N					
Y					



[실습문제 3-1] 최대 공통 부분 수열

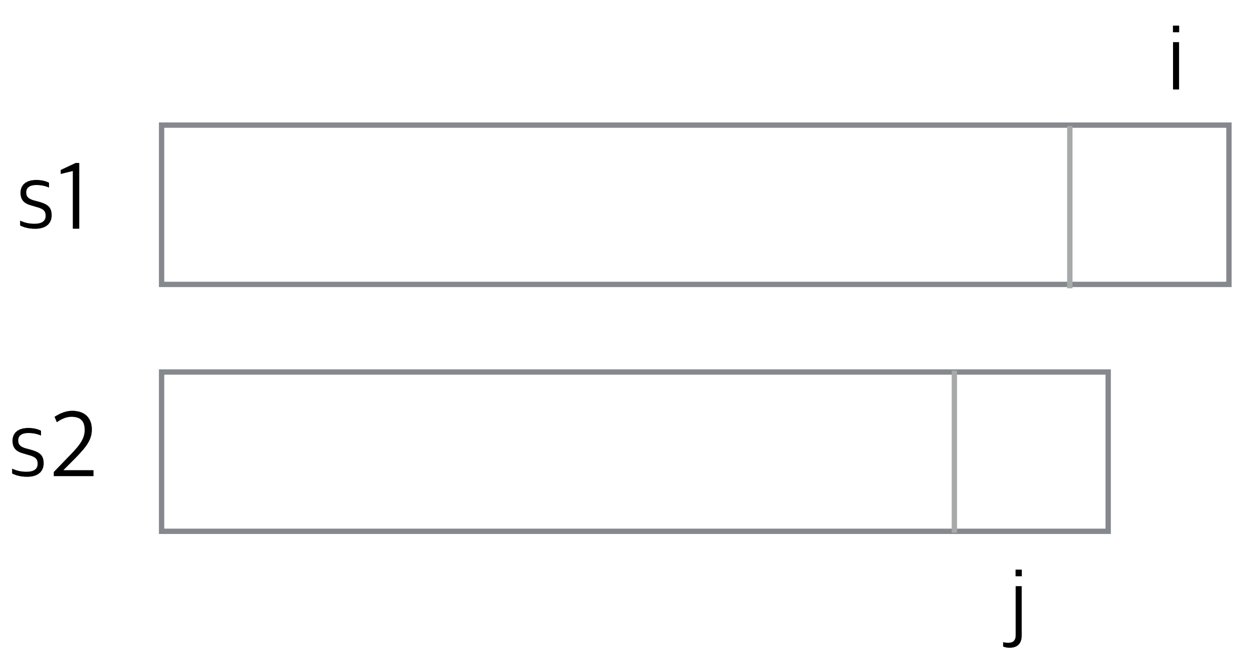
1. Table을 정의한다
   - T(i, j) = str1의 1 ~ i, str2의 1 ~ j의 최대공통부분수열 길이

	S	N	O	W	Y
S	1	1	1	1	1
U	1	1	1	1	1
N	1	2	2	2	2
N	1	2	2	2	2
Y	1	2	2	2	3



[실습문제 3-1] 최대 공통 부분 수열

1. Table을 정의한다
   - T(i, j) = str1의 1 ~ i, str2의 1 ~ j의 최대공통부분수열 길이

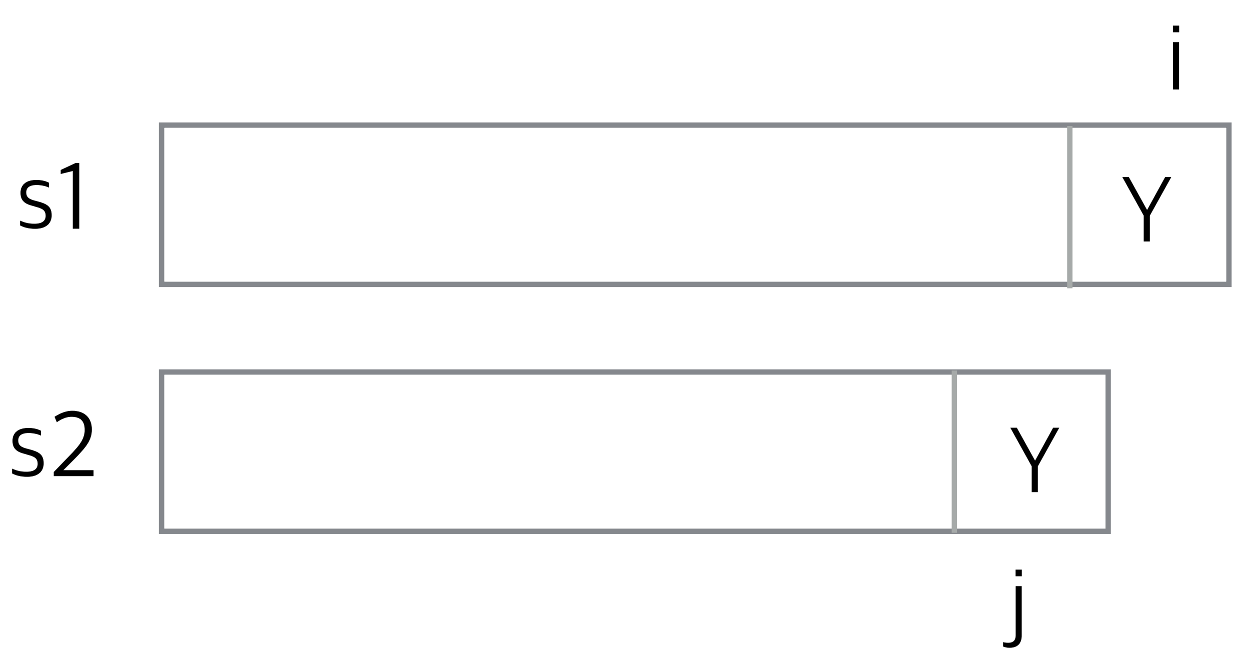


	S	N	O	W	Y
S	1	1	1	1	1
U	1	1	1	1	1
N	1	2	2	2	2
N	1	2	2	2	2
Y	1	2	2	2	3

[실습문제 3-1] 최대 공통 부분 수열

1. Table을 정의한다
   - T(i, j) = str1의 1 ~ i, str2의 1 ~ j의 최대공통부분수열 길이

if s1[i] == s2[j]



	S	N	O	W	Y
S	1	1	1	1	1
U	1	1	1	1	1
N	1	2	2	2	2
N	1	2	2	2	2
Y	1	2	2	2	3

[실습문제 3-1] 최대 공통 부분 수열

1. Table을 정의한다
   - T(i, j) = str1의 1 ~ i, str2의 1 ~ j의 최대공통부분수열 길이

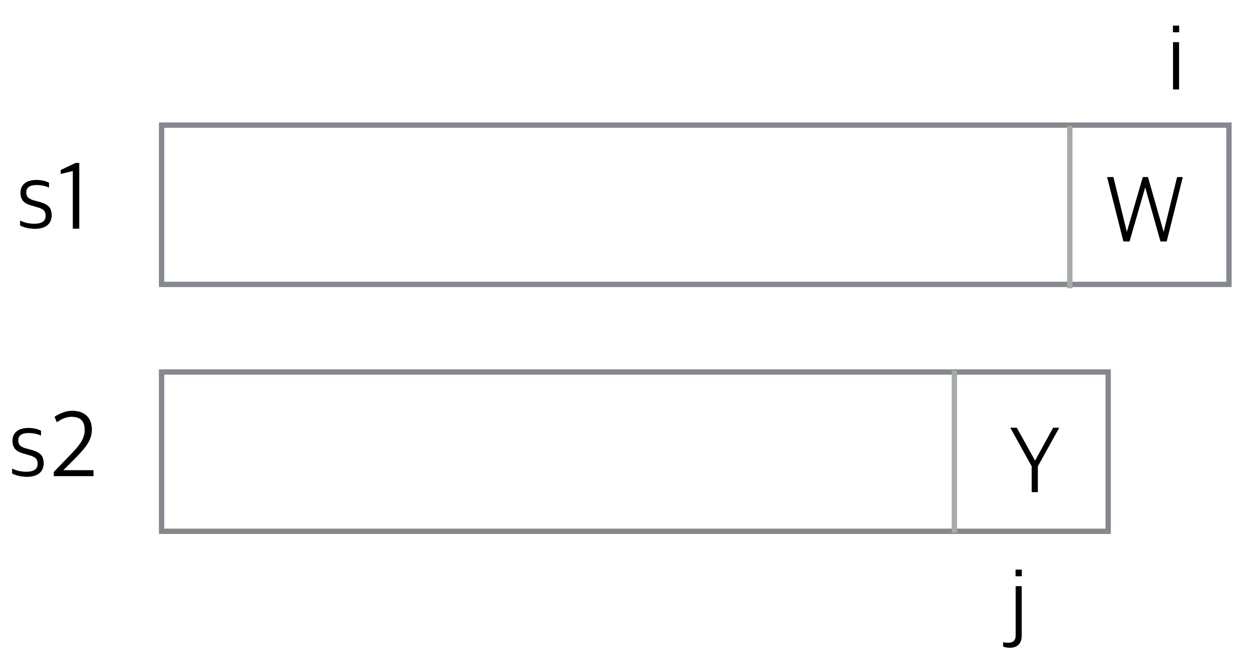
if s1[i] == s2[j]
  T(i, j) = T(i-1, j-1) + 1

	S	N	O	W	Y
S	1	1	1	1	1
U	1	1	1	1	1
N	1	2	2	2	2
N	1	2	2	2	2
Y	1	2	2	2	3

[실습문제 3-1] 최대 공통 부분 수열

1. Table을 정의한다
   - T(i, j) = str1의 1 ~ i, str2의 1 ~ j의 최대공통부분수열 길이

if s1[i] != s2[j]

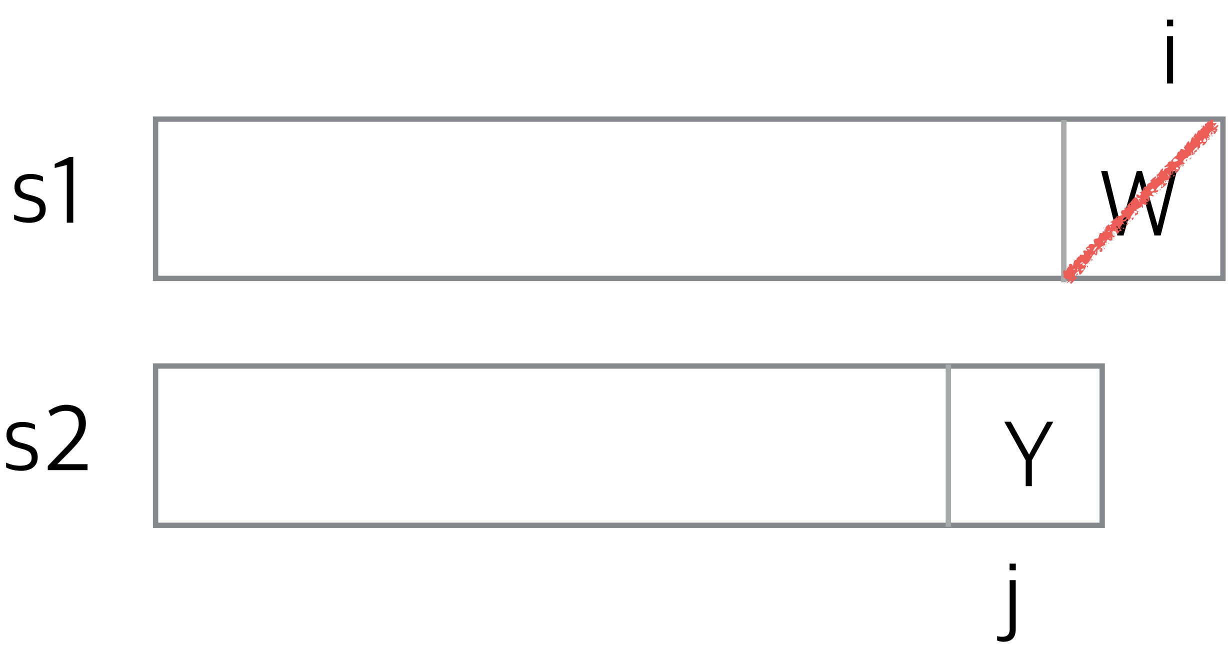


	S	N	O	W	Y
S	1	1	1	1	1
U	1	1	1	1	1
N	1	2	2	2	2
N	1	2	2	2	2
Y	1	2	2	2	3

[실습문제 3-1] 최대 공통 부분 수열

1. Table을 정의한다
   - T(i, j) = str1의 1 ~ i, str2의 1 ~ j의 최대공통부분수열 길이

if s1[i] != s2[j]
    T(i, j) = T(i-1, j)

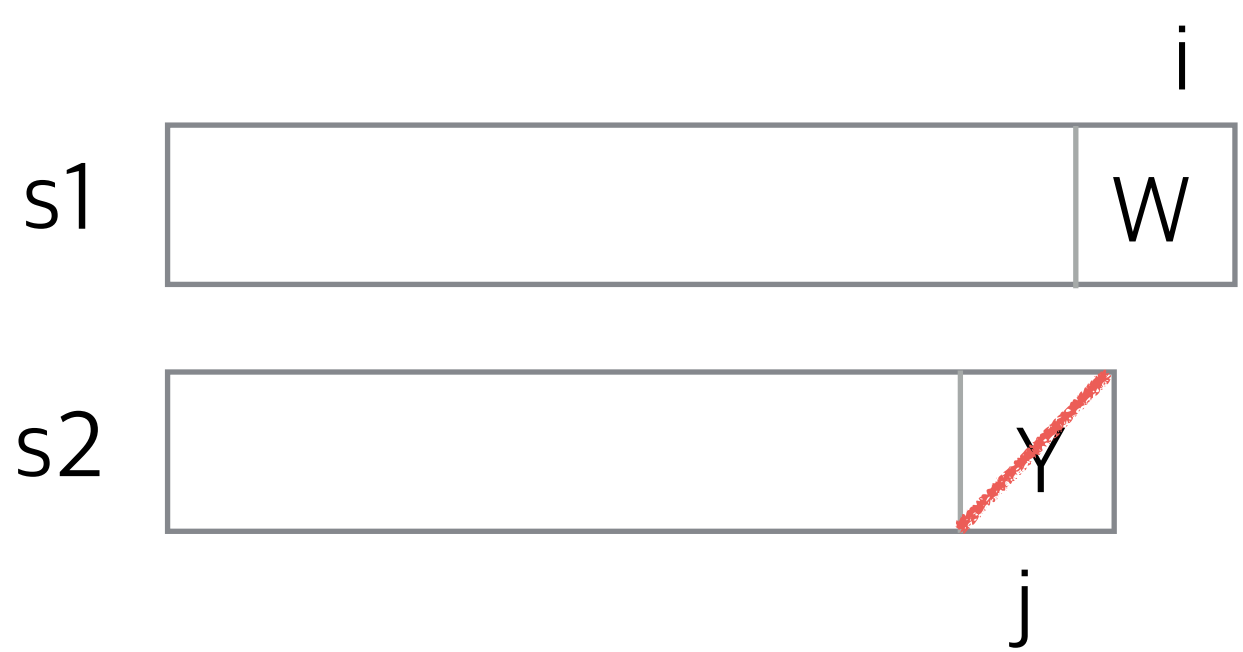


	S	N	O	W	Y
S	1	1	1	1	1
U	1	1	1	1	1
N	1	2	2	2	2
N	1	2	2	2	2
Y	1	2	2	2	3

[실습문제 3-1] 최대 공통 부분 수열

1. Table을 정의한다
   - T(i, j) = str1의 1 ~ i, str2의 1 ~ j의 최대공통부분수열 길이

if s1[i] != s2[j]
    T(i, j) = max(T(i-1, j), T(i, j-1))



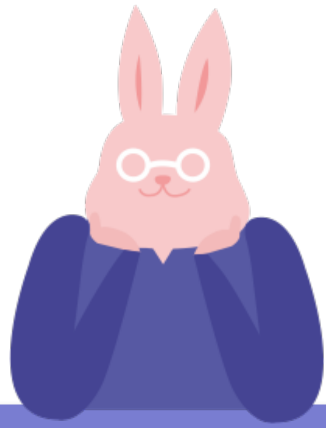
	S	N	O	W	Y
S	1	1	1	1	1
U	1	1	1	1	1
N	1	2	2	2	2
N	1	2	2	2	2
Y	1	2	2	2	3

[실습문제 3-1] 최대 공통 부분 수열

1. Table을 정의한다
   - T(i, j) = str1의 1 ~ i, str2의 1 ~ j의 최대공통부분수열 길이

2. 점화식을 구한다
   - T(i, j) = max(T(i-1, j), T(i, j-1)) if s1[i] != s2[j]

     T(i-1, j-1) + 1 otherwise



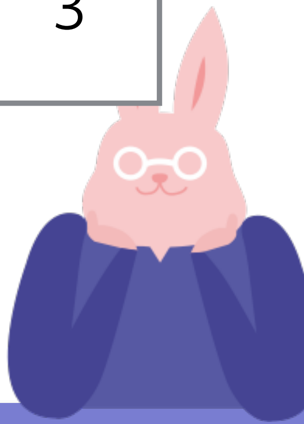
[실습문제 3-1] 최대 공통 부분 수열

3. 어느 순서로 Table을 구해야 하는지를 생각한다
   - i=1 —> i=n

4. 답은 어디에 있는지를 찾는다
   - T(n, m)
     n = len(str1), m = len(str2)

5. 시간복잡도
   - O(nm)

1	1	1	1	1
1	1	1	1	1
1	2	2	2	2
1	2	2	2	2
1	2	2	2	3



[예제] 두 문자열의 최단거리

- 문자열 A을 시작으로 문자열 B를 만들기 위한 최소 연산
  연산 : 한 개의 알파벳을 추가 / 제거

입력의 예

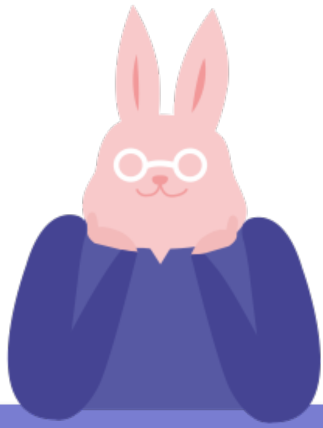
```
television
telephone
```

출력의 예

```
7
```

[예제] 두 문자열의 최단거리

- { s1 - LCS(s1, s2) } 는 제거하고, { s2 - LCS(s1, s2) } 를 추가
- (len(s1) - LCS(s1, s2)) + (len(s2) - LCS(s1, s2))
- 증명?



[예제] 두 문자열의 최단거리

- { s1 - LCS(s1, s2) } 는 제거하고, { s2 - LCS(s1, s2) } 를 추가
- (len(s1) - LCS(s1, s2)) + (len(s2) - LCS(s1, s2))
- 증명?
  - 연산 순서를 강제해도 괜찮다 : 제거를 먼저 한 후, 추가를 하자
  - s1 —> s' —> s2 where s' $\subseteq$ s1, s' $\subseteq$ s2
  - s' 는 길수록 좋다
- O(nm) where n = len(s1), m = len(s2)

감사합니다!

신현규

E-mail : hyungyu.sh@kaist.ac.kr
Kakao : yougatup