Algorithm

1주차: 정렬 알고리즘

1주차: 정렬 알고리즘

INDEX

정렬 알고리즘

Bubble Sort
Selection Sort
Recursive Bubble Sort
Insertion Sort
Recursive Insertion Sort
Merge Sort
Iterative Merge Sort
Quick Sort
Iterative Quick Sort
Heap Sort
Counting Sort
Radix Sort
Bucket Sort
ShellSort
TimSort
Comb Sort
Pigeonhole Sort
Cycle Sort
Cocktail Sort
Bitonic Sort
Pancake sorting
Strand Sort
Binary Insertion Sort
BogoSort or Permutation Sort
Gnome Sort
Sleep Sort – The King of Laziness / Sorting while Sleeping
Structure Sorting (By Multiple Rules) in C++
Stooge Sort
Tag Sort (To get both sorted and original)
Tree Sort
Cartesian Tree Sorting
Odd-Even Sort / Brick Sort
QuickSort on Singly Linked List
QuickSort on Doubly Linked List
3-Way QuickSort (Dutch National Flag)
Merge Sort for Linked Lists
Merge Sort for Doubly Linked List
3-way Merge Sort

왜 배워야 하는가?

1. 알고리즘 설계 / 디자인 하는 법을 배움
2. 실제 정렬을 해야 하는 데이터에 따라 유리한 정렬 알고리즘이 다름
3. 코딩테스트의 문제를 풀 때 가장 기초가 되는 것이 정렬임

정렬 알고리즘의 성능 측정

O(1)

```
printf("Hello");
```

O(n)

```
for (i=0; i<n; i++)
{
  printf("Hello");
}
```

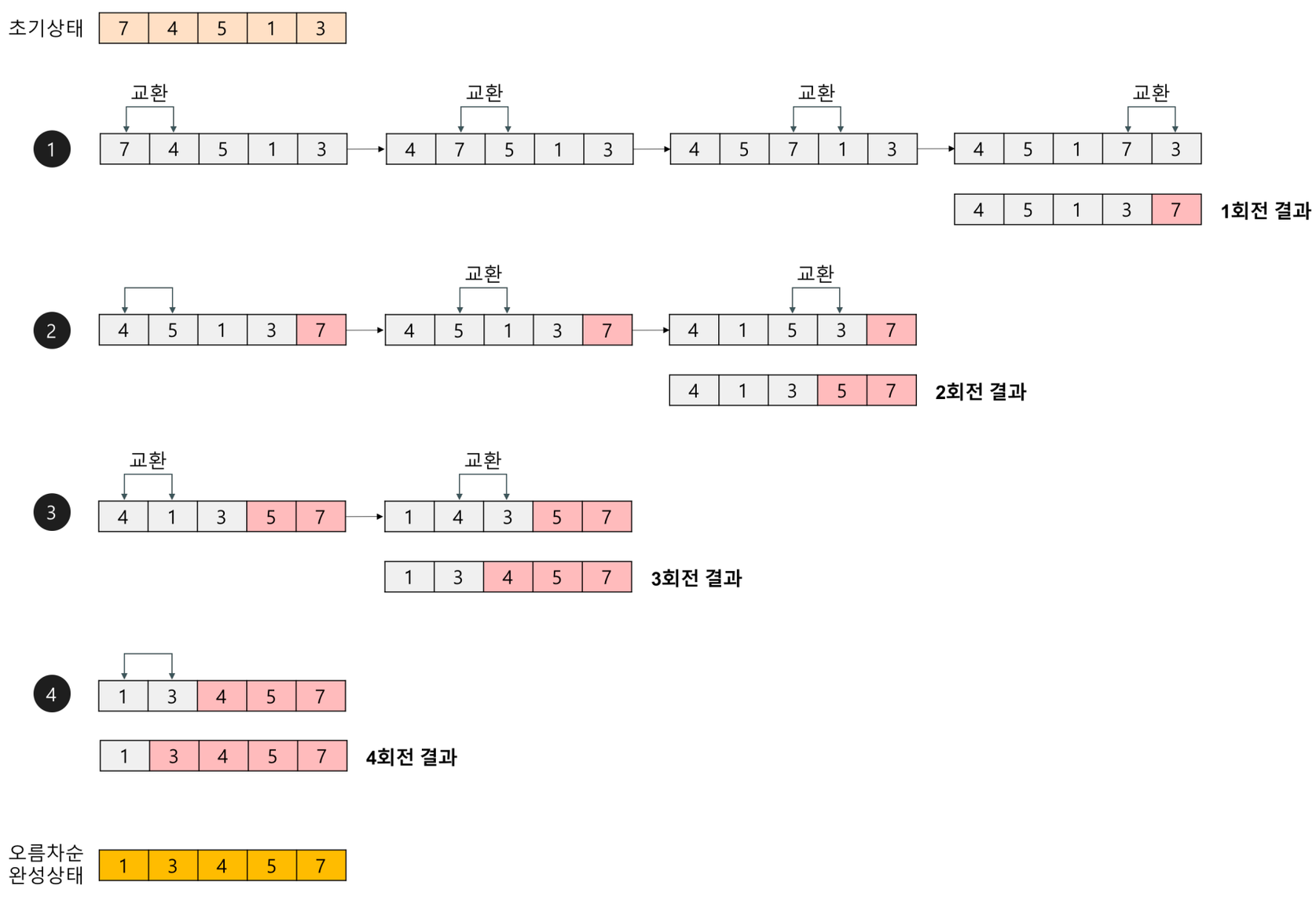
O(n²)

```
for (i=0; i<n; i++)
{
 for (j=0; j<n; j++)
 {
   printf("Hello");
 }
}
```

3. Bubble Sort

가장 간단한 정렬 방법

핵심 전략: Swap



최적화한 Bubble Sort

최적화 전

```c
for (i=0; i<n; i++)
{
  for (j=0; j<n-1; j++)
  {
    if (arr[j] > arr[j+1])
    {
      swap(&arr[j], &arr[j+1]);
    }
  }
}
```

최적화 후

```c
for (i=0; i<n; i++)
{
  for (j=0; j<n-i; j++)
  {
    if (arr[j] > arr[j+1])
    {
      swap(&arr[j], &arr[j+1]);
    }
  }
}
```

4. Selection Sort

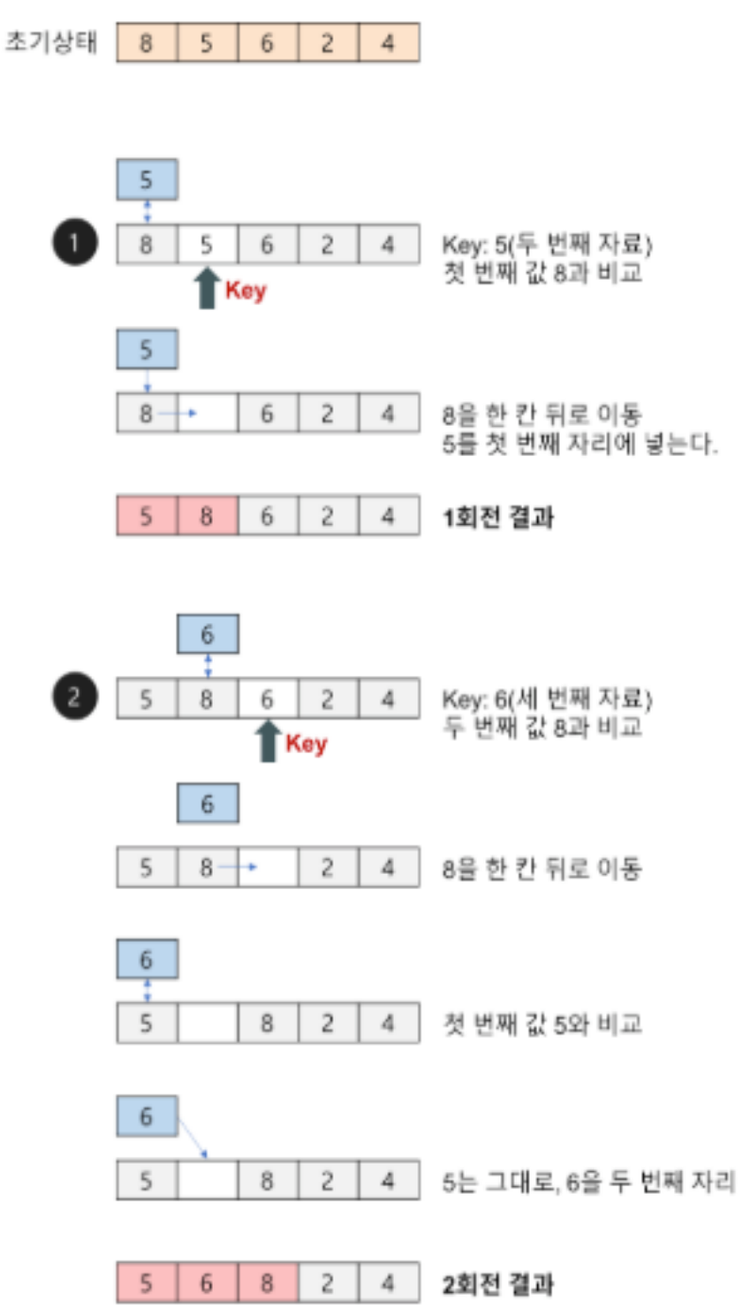
최대 / 최소값을 Select

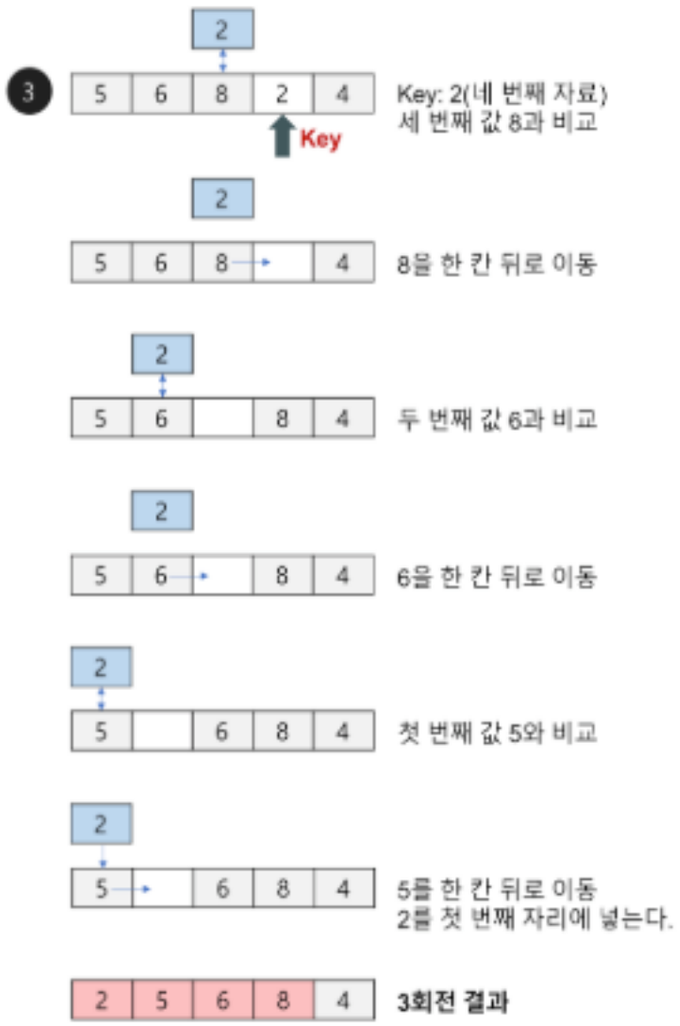
핵심 전략: Select

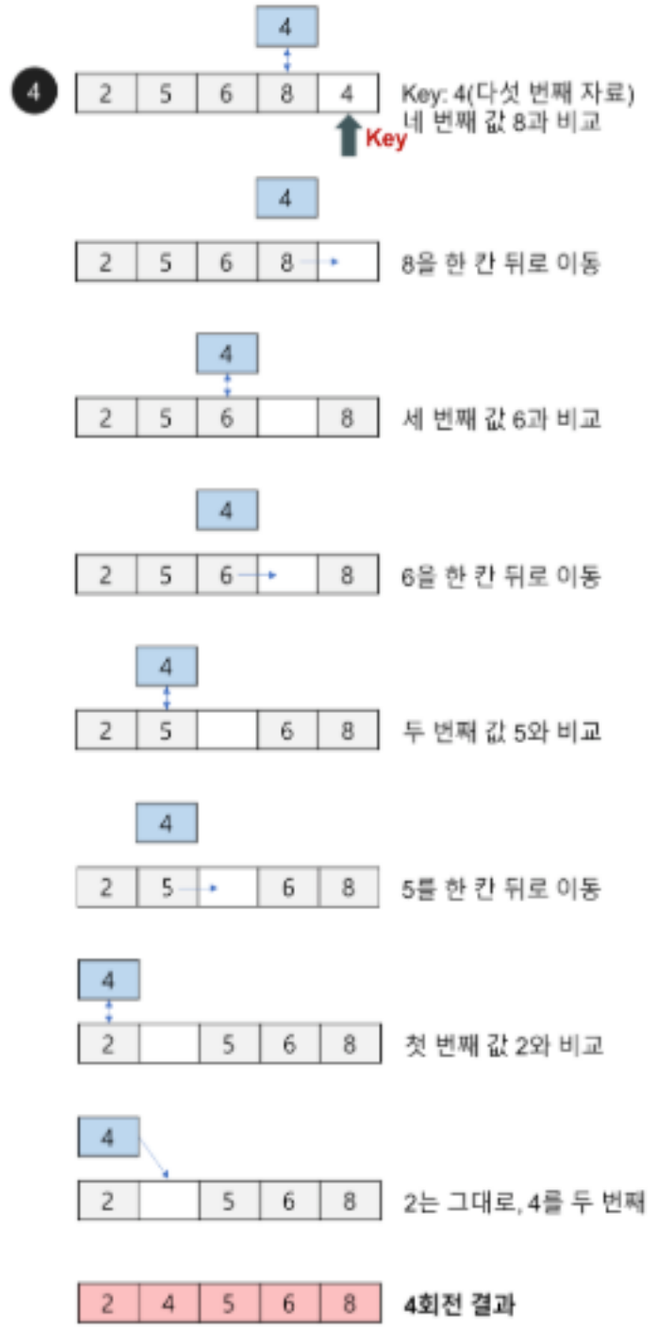
5. Insertion Sort

자기 위치를 찾아서 Insert

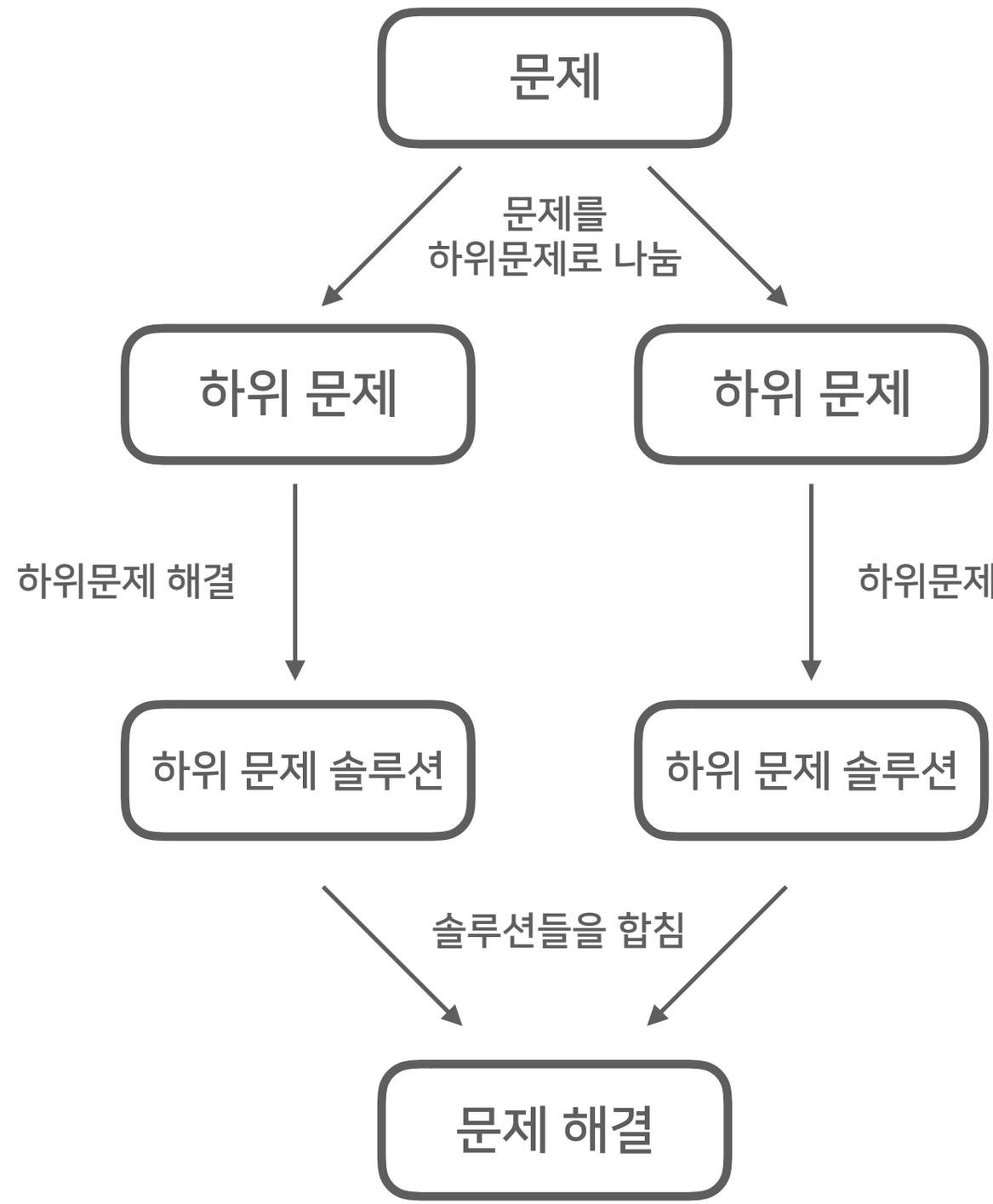
핵심 전략: Insert







재귀적으로 문제를 나눔



문제: A의 B승 계산

정수 A와 B를 입력받고 A의 B승을 계산하여 출력 하시오.

입력 예시1:
2 11

출력 예시1:
2048

입력 예시2:
5 10

출력 예시2:
9765625

해답: A의 B승 계산

```
for (i=0; i<B; i++)
{
  answer *= A;
}
```

O(n)

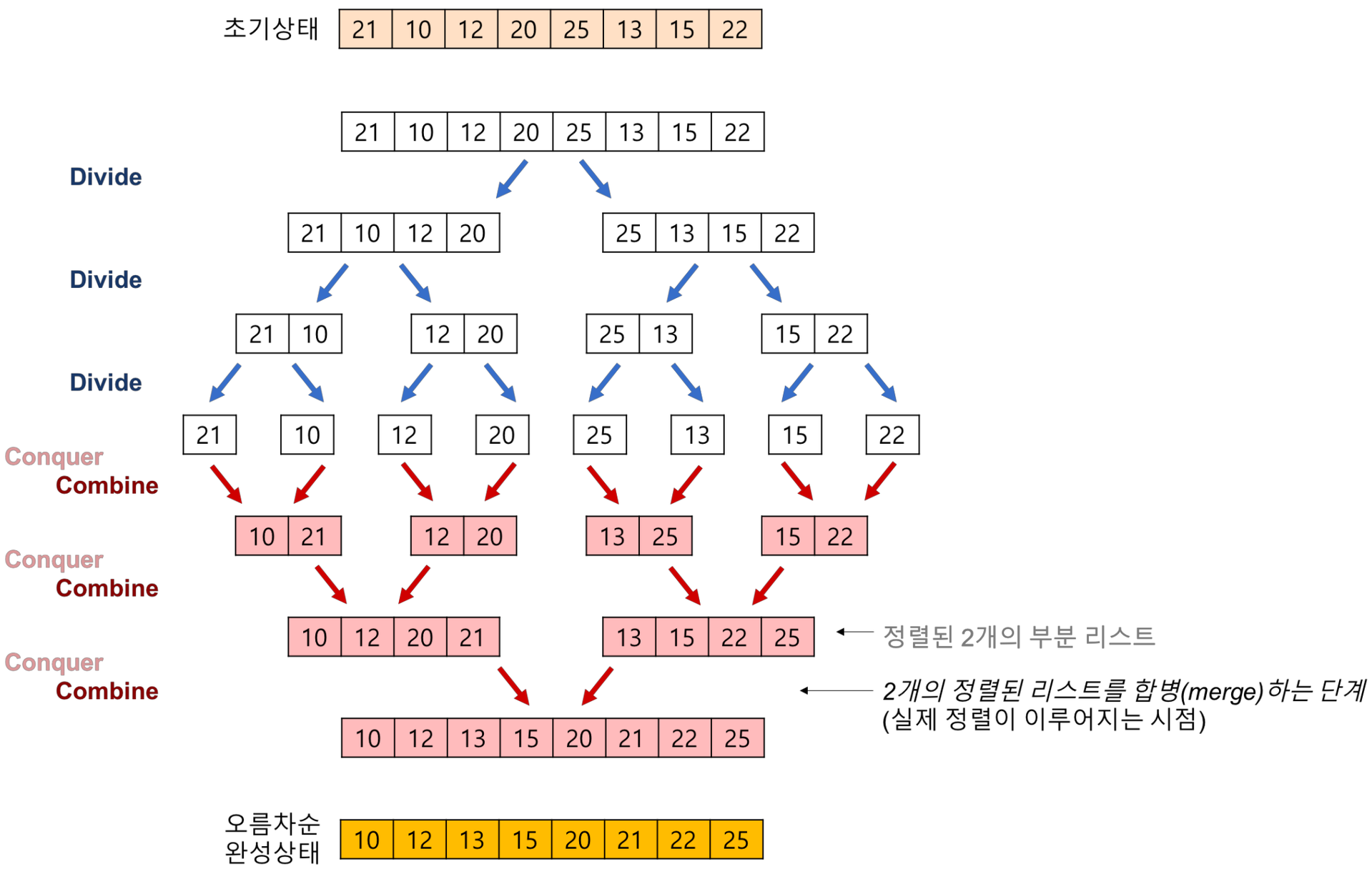
```
int power(int A, int B)
{
    int tmp;

    if (B == 0) return 1;

    tmp = power(A, B/2);

    if (B % 2 == 0) return tmp * tmp;
    else return A * tmp * tmp;
}
```

O(logn)

7. Merge Sort

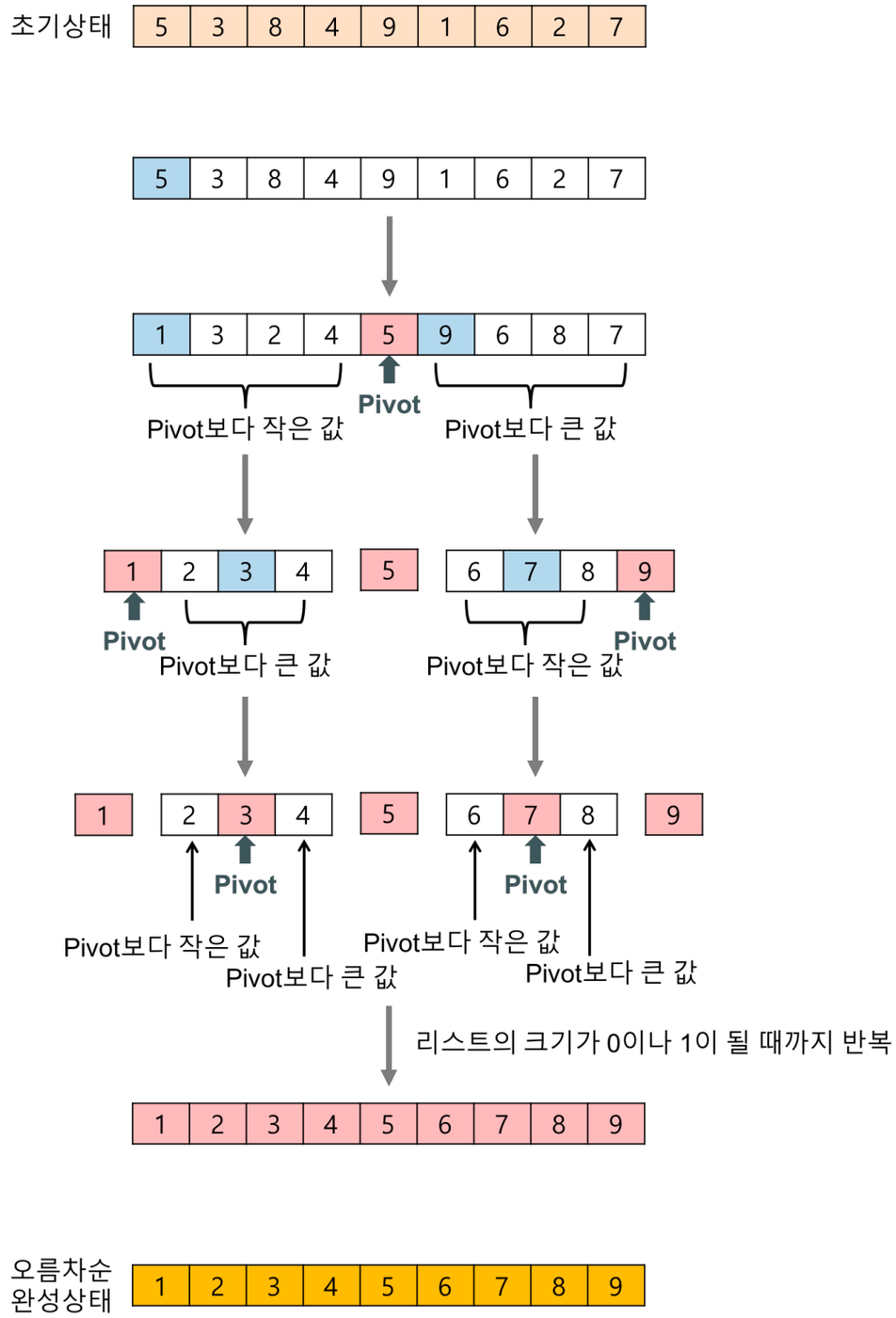
잘게 나눴다가 합치면서 정렬

핵심 전략: Divide and Conquer



Pivot 값을 기준으로 나눔

핵심 전략: Divide and Conquer



전체 요약

	In Place	Stable	Best	Worst	Average
Bubble Sort	O	O	O(n²)	O(n²)	O(n²)
Selection Sort	O	O	O(n²)	O(n²)	O(n²)
Insertion Sort	O	O	O(n)	O(n²)	O(n²)
Merge Sort	X	O	O(nlogn)	O(nlogn)	O(nlogn)
Quick Sort	O	X	O(nlogn)	O(n²)	O(nlogn)

9. 요약

가장 좋은 알고리즘은?

Quick Sort - Stable한 정렬이 필요하지 않고, 최악보단 평균이 중요할 때.

Merge Sort - Stable한 정렬이 필요할 때. 단, 공간복잡도가 O(n) 이다.

Insertion Sort - 크기가 작은 데이터를 정렬할 때.

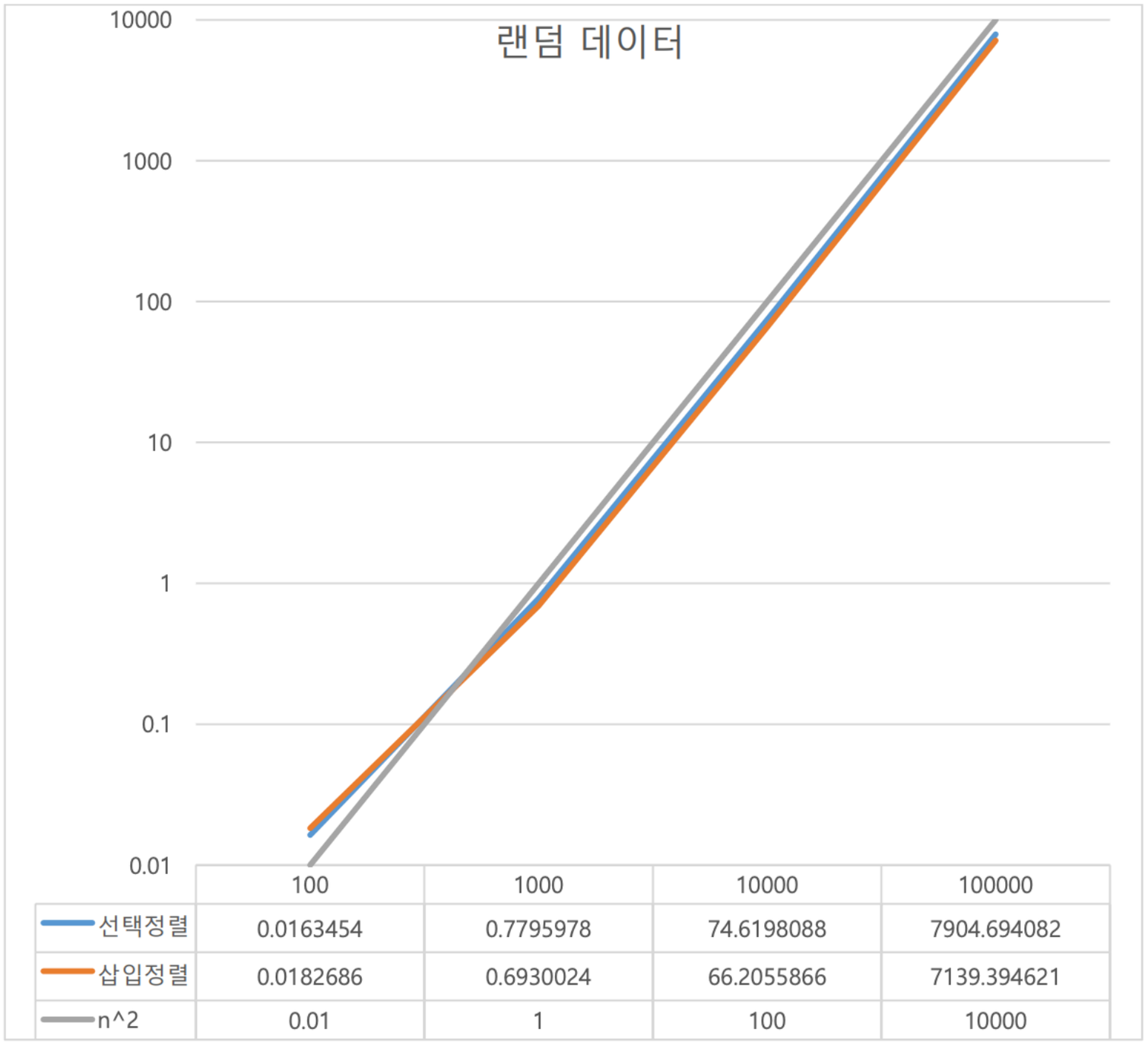
10. 과제

정렬 알고리즘의 수행시간 비교

- 지금까지 배운 정렬 알고리즘을 직접 구현하고, 수행시간을 측정해서 비교한다.
- 데이터의 갯수를 증가시키면서 시간복잡도와 비례하는지 확인한다.
- 데이터는 랜덤 데이터, 정렬 데이터, 역정렬 데이터를 사용하여 최선, 평균, 최악을 비교한다.
- 구현한 알고리즘과 수행시간을 측정하는 코드를 Github에 Pull Request를 보내서 제출한다.
- 수행시간을 기록하고, 해당 정렬방법의 시간복잡도와 비교 / 분석한 보고서를 작성한다.
- 보고서는 이메일 (jamesj0107@naver.com)로 제출한다.
- 기한은 2019년 7월 14일 23시 59분 까지.
- 기한내 제출 못할 시 벌금있음.

10. 과제

과제 예시



	100	1000	10000	100000
선택정렬	0.0163454	0.7795978	74.6198088	7904.694082
삽입정렬	0.0182686	0.6930024	66.2055866	7139.394621
n^2	0.01	1	100	10000

11. References

https://gmlwjd9405.github.io/2018/05/06/algorithm-bubble-sort.html

https://www.geeksforgeeks.org/analysis-of-different-sorting-techniques/

https://www.geeksforgeeks.org/divide-and-conquer/