알고리즘 스터디

분할정복 Divide and Conquer

알고리즘 스터디

재귀 풀이를 되짚어보자

- 재귀함수가 뭔가요?

```java
static void whatIsRecursion(int n, int depth) {
    StringBuilder str = new StringBuilder();
    for (int i = 0; i < depth; i++) {
        str.append("____");
    }
    if (n == 0) {
        System.out.printf(str.toString());
        System.out.println("\"재귀함수가 뭔가요?\"");
        System.out.printf(str.toString());
        System.out.println("\"재귀함수는 자기 자신을 호출하는 함수라네\"");
        System.out.printf(str.toString());
        System.out.println("라고 답변하였지.");
        return ;
    }

    System.out.printf(str.toString());
    System.out.println("\"재귀함수가 뭔가요?\"");
    System.out.printf(str.toString());
    System.out.println("\"잘 들어보게. 옛날옛날 한 산 꼭대기에 이세상 모든 지식을 통달한 선인이 있었어.");
    System.out.printf(str.toString());
    System.out.println("마을 사람들은 모두 그 선인에게 수많은 질문을 했고, 모두 지혜롭게 대답해 주었지.");
    System.out.printf(str.toString());
    System.out.println("그의 답은 대부분 옳았다고 하네. 그런데 어느 날, 그 선인에게 한 젊은이가 찾아와서 물었어.\"");
    whatIsRecursion(n - 1, depth + 1);
    System.out.printf(str.toString());
    System.out.println("라고 답변하였지.");
}
```

기저 조건
(재귀 종료)

호출 이전 (질문)

재귀 호출

호출 이후 (답변)

알고리즘 스터디

재귀 풀이를 되짚어보자

- 재귀함수가 뭔가요? N = 3

재귀 호출 1 : 재귀함수가 뭔가요?

if (기저조건?) -> 아님

재귀 호출 2 : 재귀함수가 뭔가요?

if (기저조건?) -> 아님

재귀 호출 3 : 재귀함수가 뭔가요?

if (기저조건?) -> O : 자기 자신을 호출하는 함수라네

답변 3

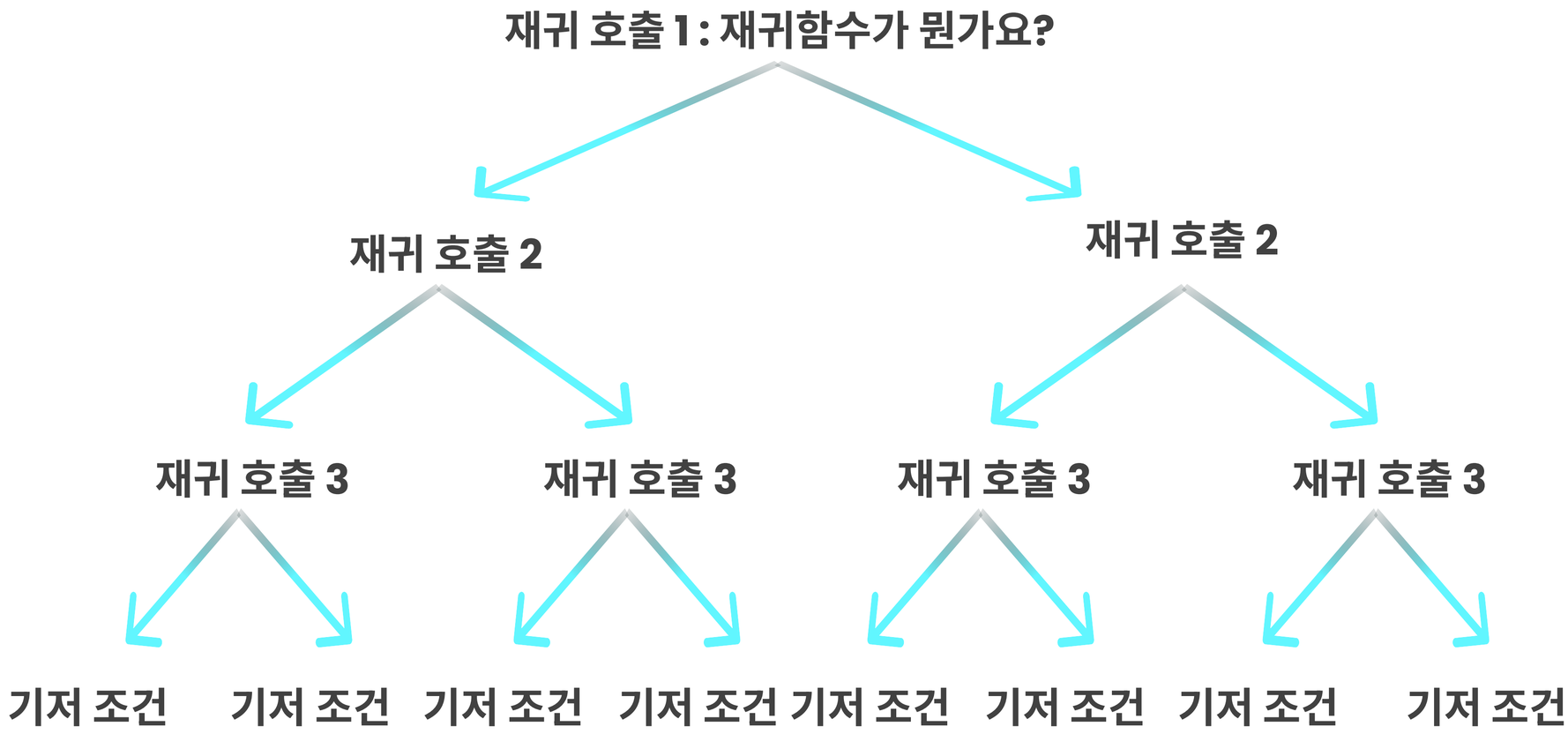
답변 2

답변 1

알고리즘 스터디

재귀 풀이를 되짚어보자

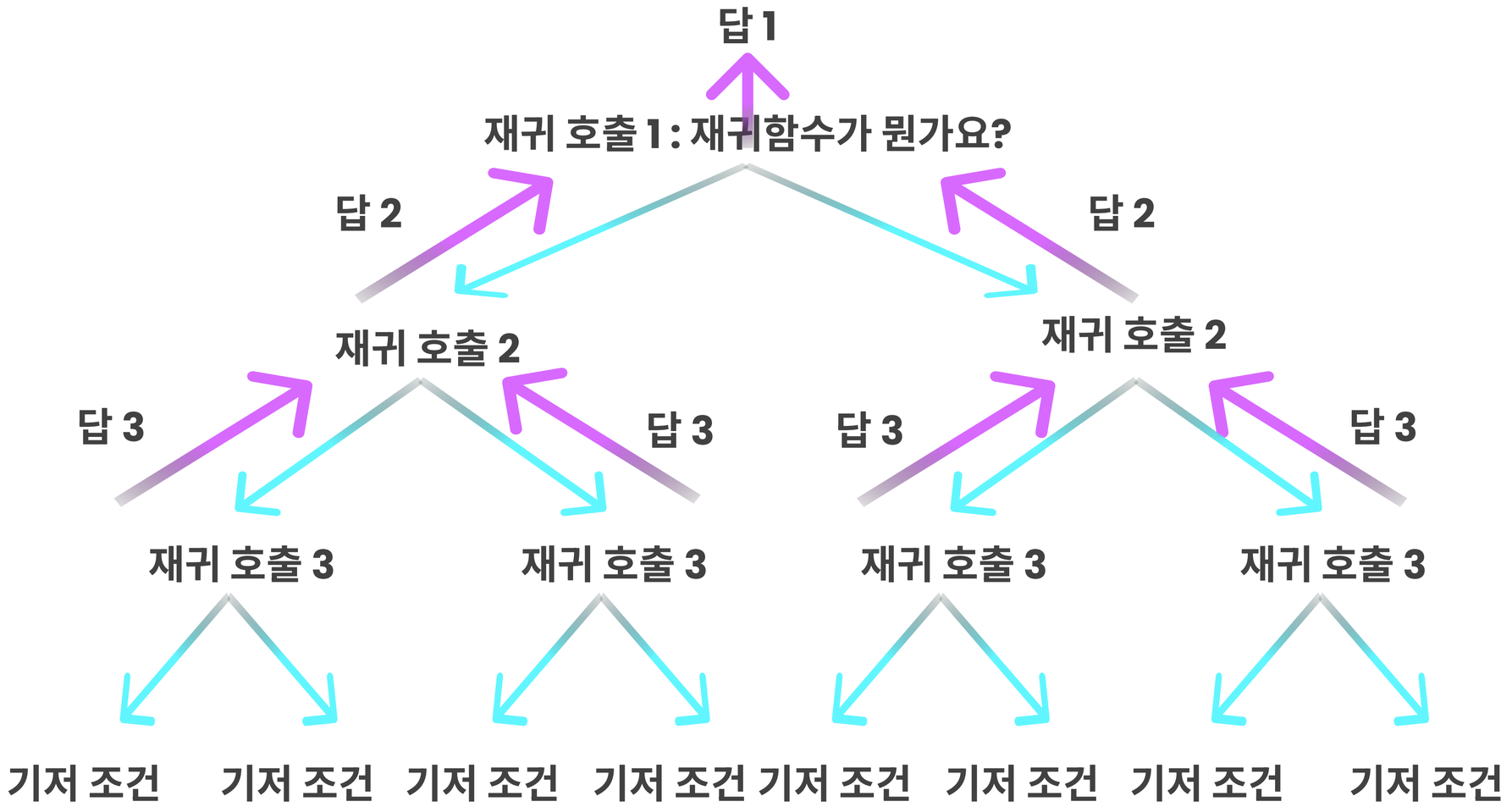
- 재귀 함수 호출을 하나가 아니라 두개를 해버리면....???



알고리즘 스터디

재귀 풀이를 되짚어보자

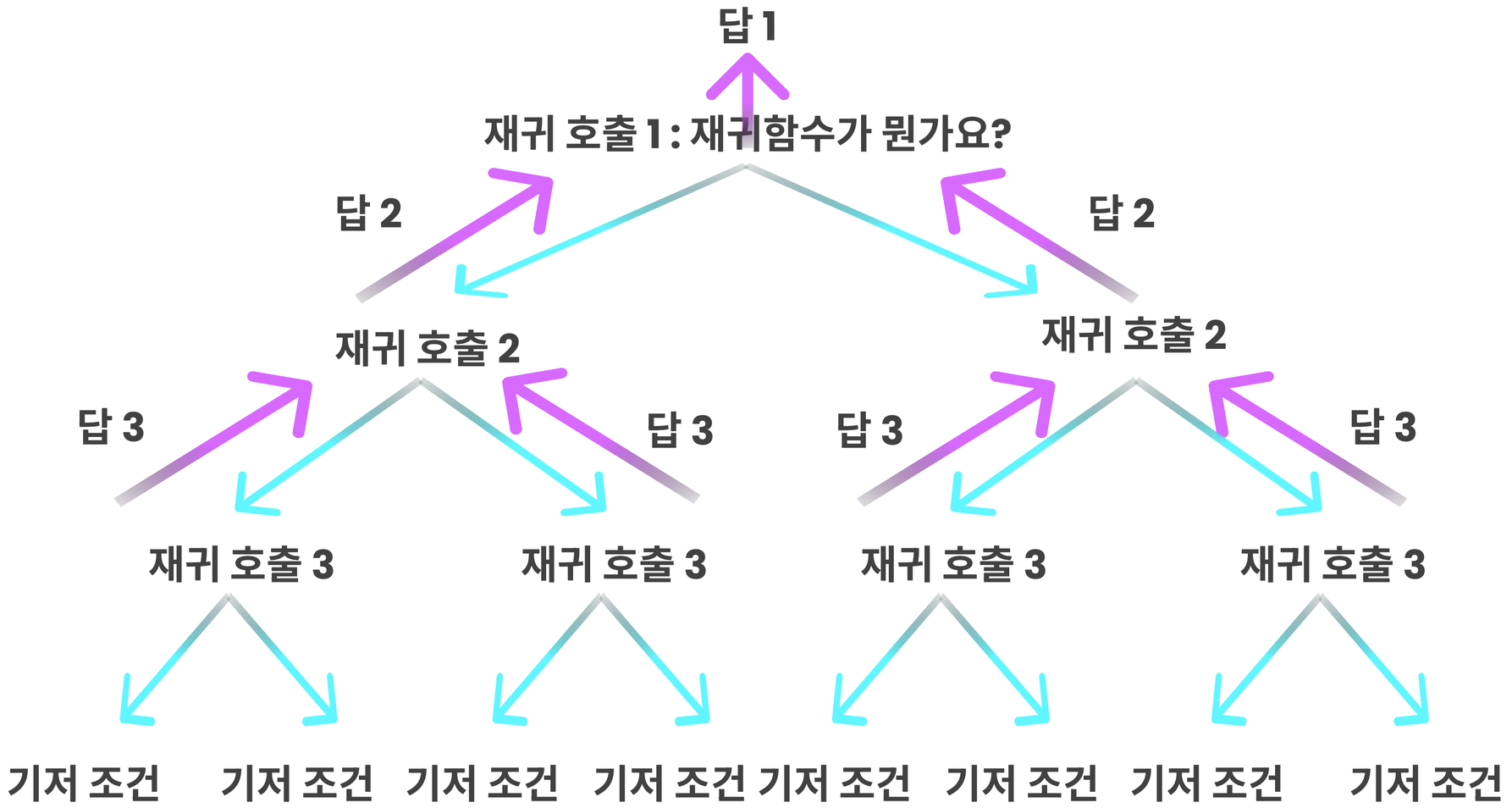
- 재귀 함수 호출을 하나가 아니라 두개를 해버리면....???



알고리즘 스터디

재귀 풀이를 되짚어보자

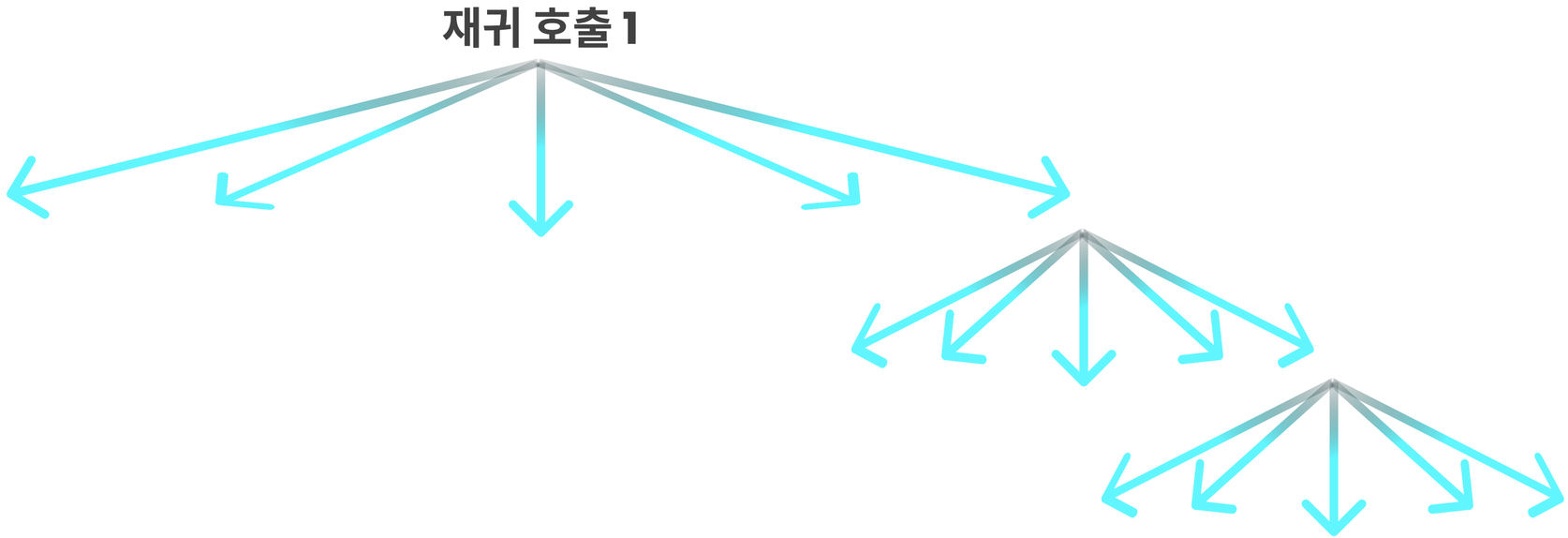
- 여기서 호출 이전(질문) 과 답 출력(기저조건) 이 문제를 위한 계산식이라면?



알고리즘 스터디

재귀 풀이를 되짚어보자 (번외)

- 2번 호출하는게 아니라, 반복문을 통한 n번 탐색이라면....?
  -> 백트래킹



알고리즘 스터디

분할정복 Divide and Conquer

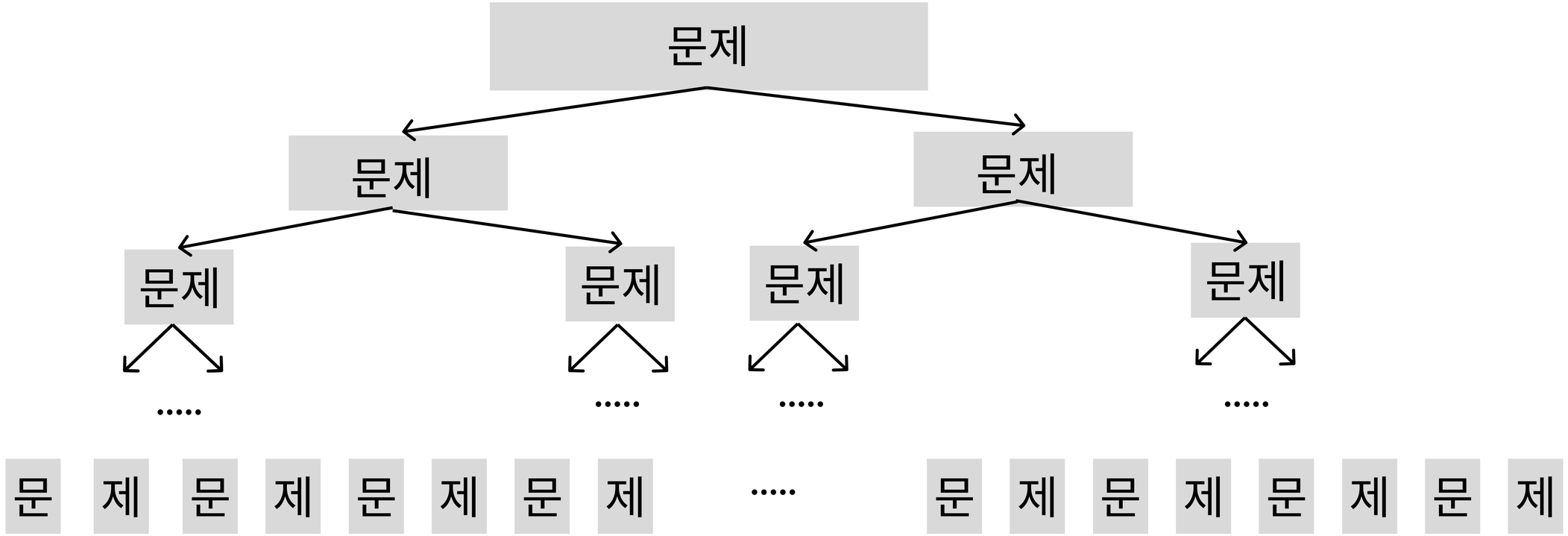
- 하나의 큰 문제를 작은 문제로 분할하여 문제를 해결

1. 하나의 문제를 둘 이상의 부분 문제로 나누는 것을
   더이상 분할할 수 없을때까지 진행한다.
2. 부분 문제에 대한 답을 구한다.
3. 2를 이용하여 나누기 이전 문제의 답을 구한다.
4. 3을 반복하다 보면 원래의 큰 문제에 대한 답에 도달한다.

분할정복 Divide and Conquer

- 하나의 문제를 둘 이상의 부분 문제로 나누는 것을 더이상 분할할 수 없을때까지 진행한다.

: Divde



알고리즘 스터디

분할정복 Divide and Conquer

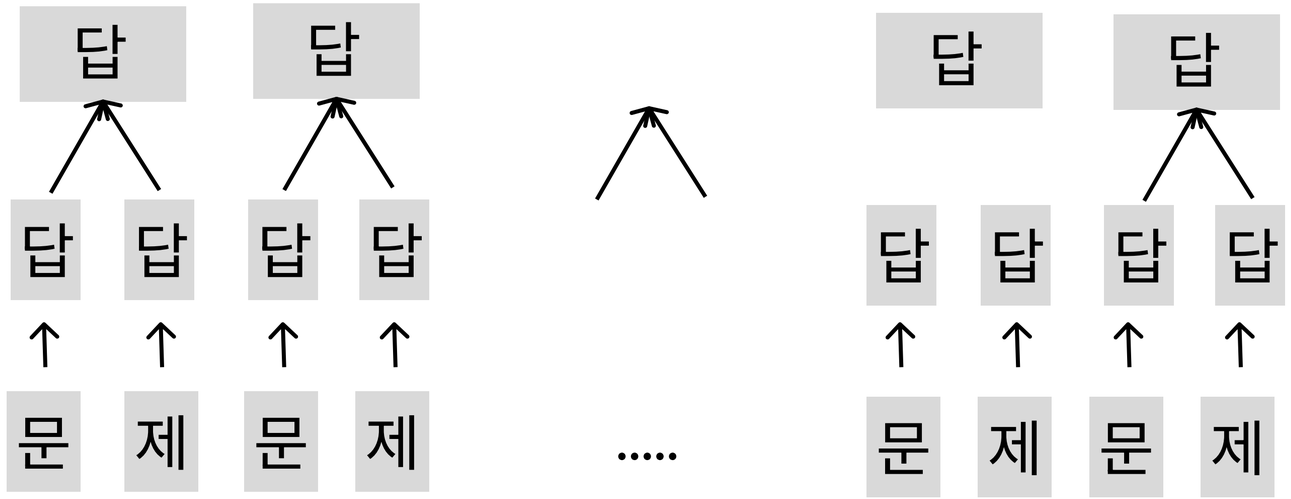
- 부분 문제에 대한 답을 구한다.

알고리즘 스터디

분할정복 Divide and Conquer

- 부분 문제의 답을 이용하여 나누기 이전 문제의 답을 구한다.

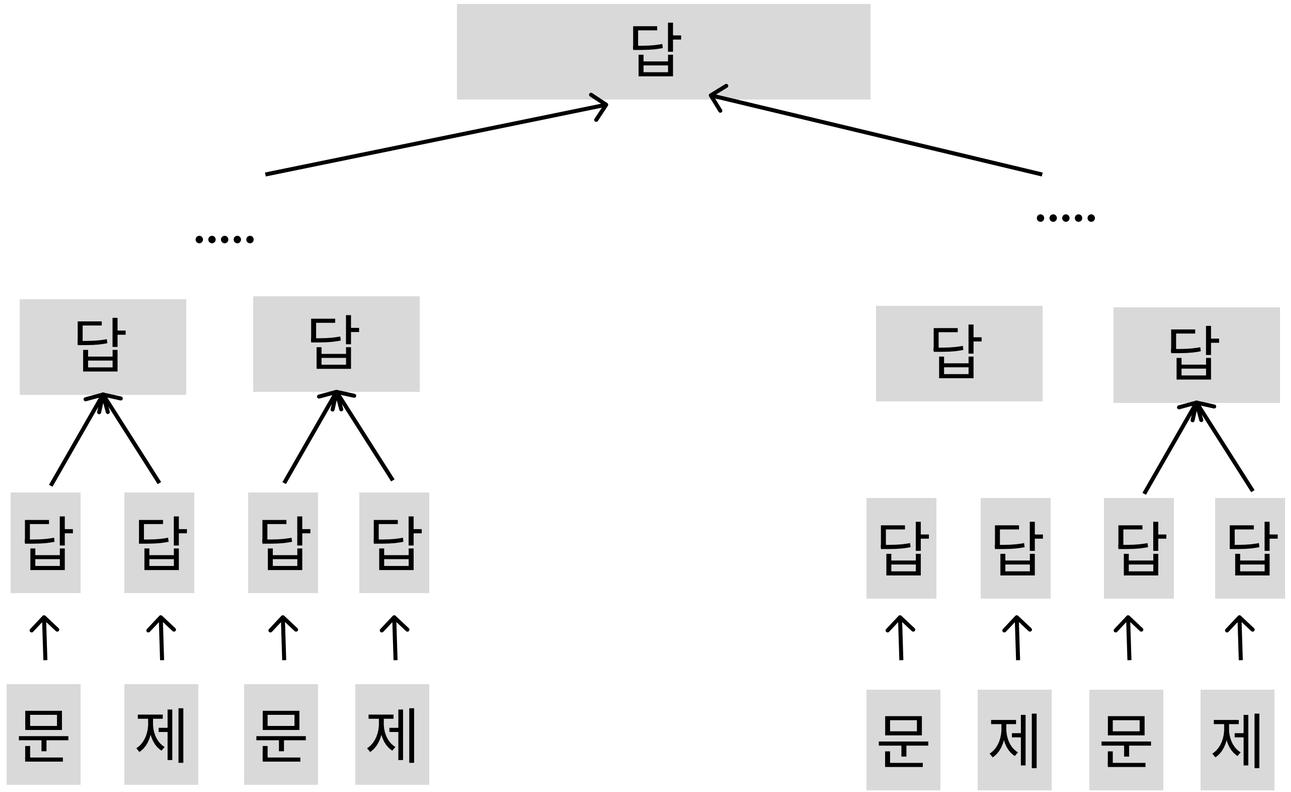
: Merge



분할정복 Divide and Conquer

- 이것을 반복하다 보면 원래 문제의 답에 도달한다.

: Merge



알고리즘 스터디

분할정복 사용처

- Merge Sort, Quick Sort 구현
- 거듭제곱의 계산 등...

알고리즘 스터디

분할정복 구조

```java
static void whatIsRecursion(int n, int depth) {
    StringBuilder str = new StringBuilder();

    if (n == ???) {
        // 기저 조건, 가장 작은 단위의 문제, Conquer
        return ???;
    }

    // 문제를 작은 단위로 분할
    whatIsRecursion(???);
    whatIsRecursion(???);


    // merge 과정, 답 도출을 위해 필요한 것이 무엇인가?
    return ???;
}
```

- Divde

** 문제를 어떻게 나눌 것인가?

- Conquer
- Merge

** 결과 도출을 위해 어떤 값을 가져갈 것인가?

분할정복 Merge Sort

- Merge Sort의 기본 방법 (merge의 과정)
  - 정렬된 두 수열을 비교하여 정렬된 상태로 합쳐진 수열을 만드는 방법

4 > 3 이므로 3이 먼저

3

알고리즘 스터디

분할정복 Merge Sort

- Merge Sort의 기본 방법 (merge의 과정)
  - 정렬된 두 수열을 비교하여 정렬된 상태로 합쳐진 수열을 만드는 방법

4 7 12 23

3 5 17 63

5 > 4 이므로 4가 먼저

3 4

알고리즘 스터디

분할정복 Merge Sort

- Merge Sort의 기본 방법 (merge의 과정)
  - 정렬된 두 수열을 비교하여 정렬된 상태로 합쳐진 수열을 만드는 방법

4 7 12 23

3 5 17 63

7 > 5 이므로 5가 먼저

3 4 5

분할정복 Merge Sort

- Merge Sort의 기본 방법 (merge의 과정)
  - 정렬된 두 수열을 비교하여 정렬된 상태로 합쳐진 수열을 만드는 방법

4 7 12 23

3 5 17 63

17 > 7 이므로 7가 먼저

3 4 5 7

알고리즘 스터디

분할정복 Merge Sort

- Merge Sort의 기본 방법 (merge의 과정)
  - 정렬된 두 수열을 비교하여 정렬된 상태로 합쳐진 수열을 만드는 방법

17 > 12 이므로 12가 먼저

3 4 5 7 12

분할정복 Merge Sort

- Merge Sort의 기본 방법 (merge의 과정)
  - 정렬된 두 수열을 비교하여 정렬된 상태로 합쳐진 수열을 만드는 방법

23 > 17 이므로 17이 먼저

3 4 5 7 12 17

알고리즘 스터디

분할정복 Merge Sort

- Merge Sort의 기본 방법 (merge의 과정)
  - 정렬된 두 수열을 비교하여 정렬된 상태로 합쳐진 수열을 만드는 방법

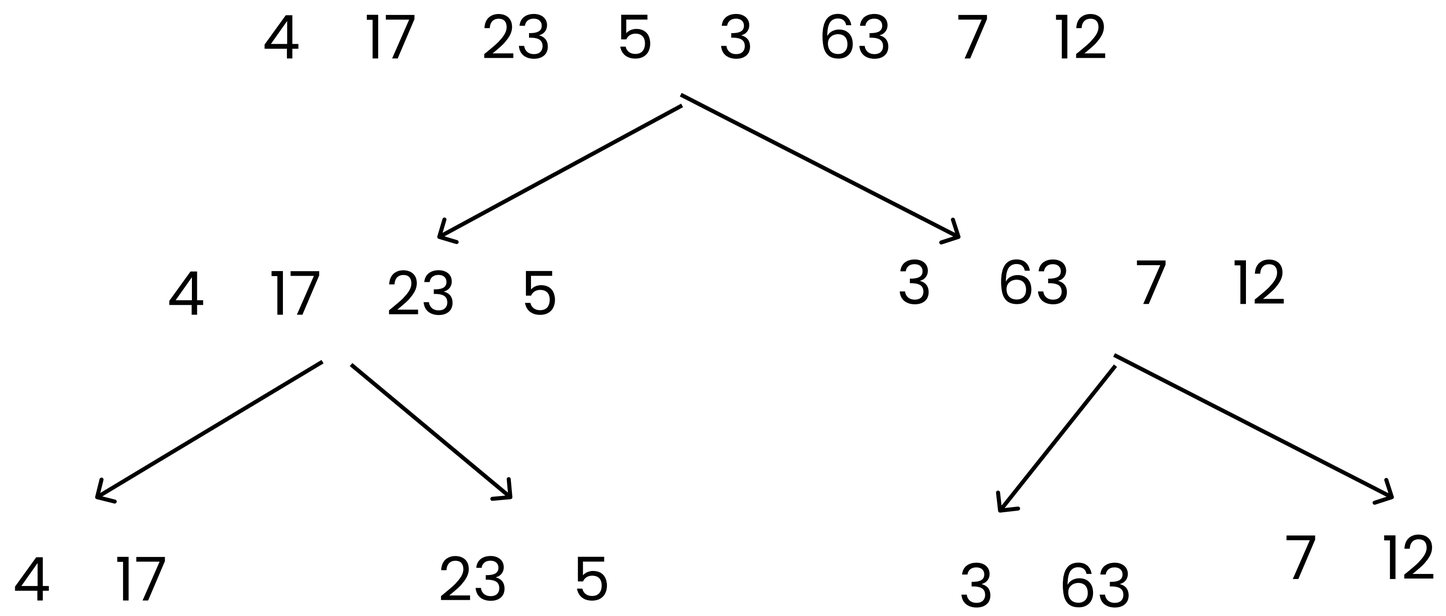
63 > 23 이므로 23이 먼저

3 4 5 7 12 17 23 63

알고리즘 스터디

분할정복 Merge Sort

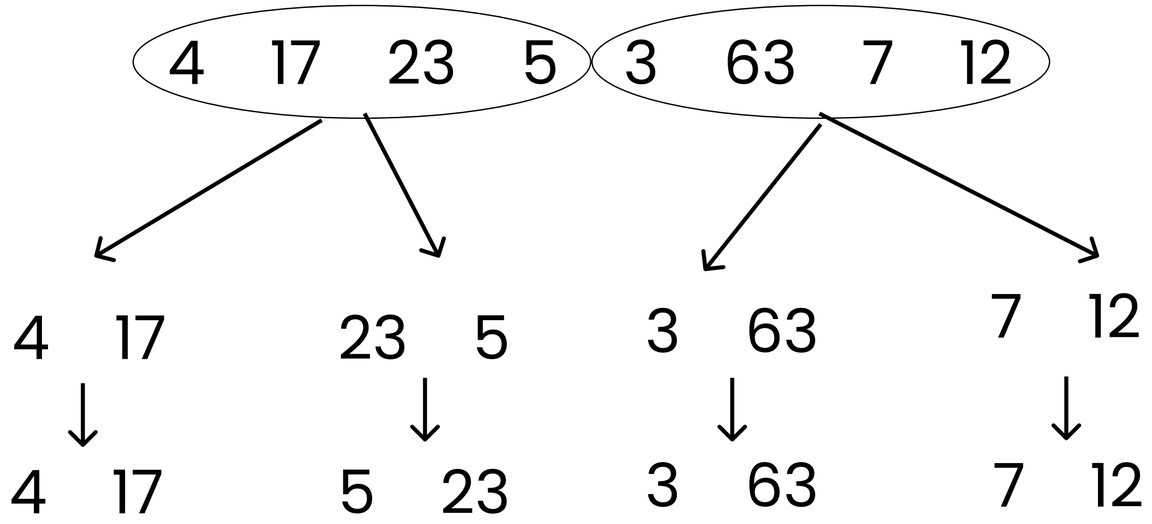
1. Divide : 문제 쪼개기



알고리즘 스터디

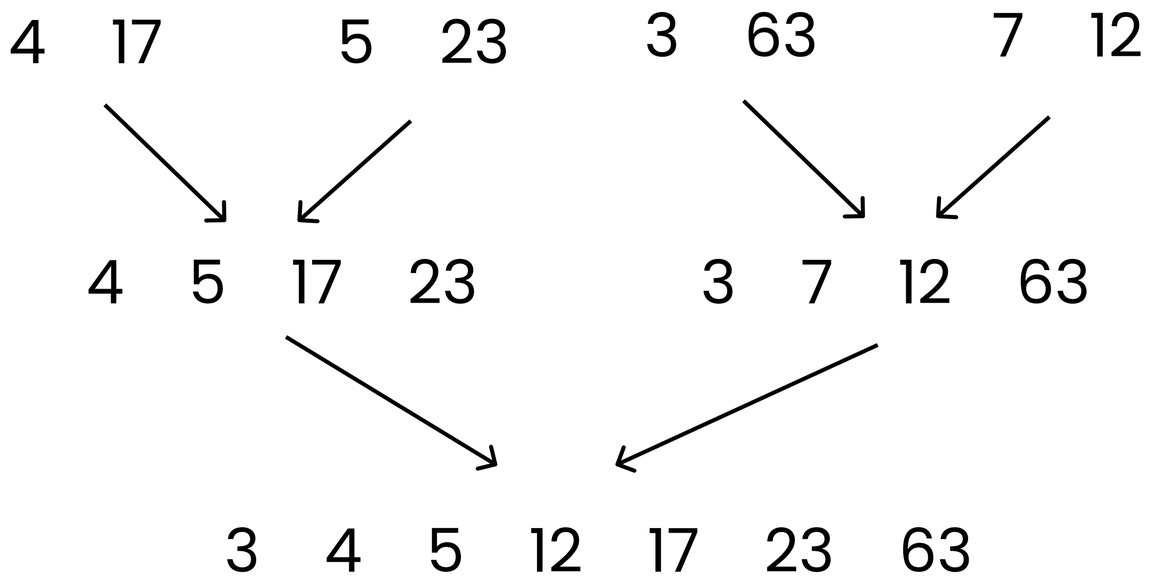
분할정복 Merge Sort

2. Conquer : 최소 단위 계산 (정렬)



분할정복 Merge Sort

3. Merge : 반환값을 이용해서 답 도출



알고리즘 스터디

분할정복 과제

1. 1992번 쿼드 트리 / 실버 1
2. 2447번 별 찍기 10 / 골드 10

- 응애 문제
  -