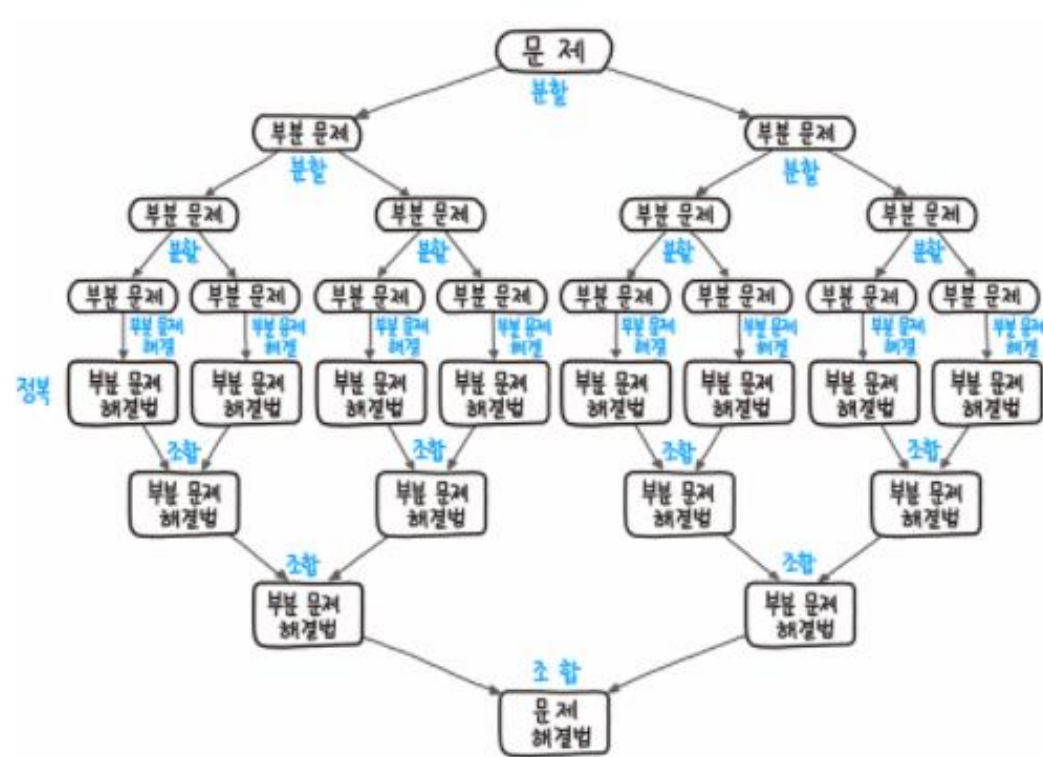


# 청년 AI 아카데미 28기 알고리즘 실습

## 재귀(Recursion)

# 재귀 (Recursion)



Top down!!

1. 문제를 굉장히 간단한 수준까지 하위 문제로 **분할**한다.
2. 하위 문제를 **해결**한다.
3. 하위 문제에 대한 결과를 원래 문제에 대한 결과로 **정복(합병)**한다.

c.f. Divide and Conquer  
기법이라고도 부른다.

# 실습 목표

- 재귀 문제를 해결하기 위한 방법을 배우고 다양한 재귀 문제들을 실습해보자.
  1. 피보나치 수열의  $n$ 번째 항 구하기
  2. 이진 탐색

# 재귀 함수 구현 Tip 3가지

1. 재귀 함수가 무엇을 하는 함수인지 **명확하게 정의합니다.**
  - **함수 이름**이 의미가 있을수록 좋습니다.
  - 함수의 **매개변수들**, 함수가 **무엇을 반환**하는지 등이 정의 안에 전부 녹아 있어야 합니다.Ex) 양의 정수 n이 주어졌을 때 1부터 n까지 합한 수를 계산하여라.

```
def Sum (a, b): # 자연수 a부터 b까지의 합을 반환하는 함수
```

Ex) Sum(4,9) = 39

# 재귀 함수 구현 Tip 3가지

2. 재귀 함수가 함수 내에서 호출이 된다면,

그것은 **올바른 답을 준다고 가정**하고, 하위 문제 정답을 통해 원래 문제의 정답을 구합니다.

- $\text{Sum}(a, b-1)$ 은  $a$ 부터  $b-1$ 까지의 합을 반환한다고 가정합니다.
- 따라서,  $\text{Sum}(a, b) = b + \text{Sum}(a, b-1)$ 입니다.

```
def Sum (a, b): # 자연수 a부터 b까지의 합을 반환하는 함수
```

```
    return Sum(a, b-1)+b
```



“ $\text{Sum}(a, b-1)$  은 자연수  $a$ 부터  $b-1$ 까지의 합을 반환해 줄 것이다” 라는 믿음

# 재귀 함수 구현 Tip 3가지

3. 재귀 함수엔 **명확한 종료** 조건이 반드시 있다.

- 재귀 함수는 자기 자신을 부르는 함수이므로, **종료 조건이 없다면 무한히 수행**됩니다.
- 따라서 함수 내에 **종료 조건**이 존재하고, 이는 **가장 단순한 경우**입니다.

Ex) Sum(a,b) 에서 a가 b보다 크다면? a와 b가 같다면?

```
def Sum (a, b): # 자연수 a부터 b까지의 합을 반환하는 함수
    if a==b:
        return a
    return Sum(a,b-1)+b
```

자연수 a부터 a까지의 합은 a이다.  
이미 알고 있는 사실

# 01. 피보나치 수열의 n번째 항 구하기

피보나치 수열의 n번째 항을 계산합니다.

$$F(1) = 1$$

$$F(2) = 1$$

$$F(3) = F(1) + F(2) = 1 + 1 = 2$$

...

$$F(N) = F(N-1) + F(N-2)$$

재귀 함수(Recursive Function)를 이용하여 구현하여 봅시다.

## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.

Start=0 (리스트 인덱스)

End=8

5	9	10	12	16	18	21	27	29
---	---	----	----	----	----	----	----	----

리스트: [ 21 11 18 ]

출력: [ ]

주어진 리스트에 21이 들어 있는지 확인하는 방법?

1. 리스트의 앞부터 차례대로 탐색  $O(n)$
2. 이진탐색  $O(\log n)$  [단, 리스트가 정렬되어 있을 때!]  
-> 굉장히 유용 ex: 검색시스템



## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.

Start=0 (리스트 인덱스)

End=8

5	9	10	12	16	18	21	27	29
---	---	----	----	----	----	----	----	----

리스트: [ 21 11 18 ]

출력: [ ]

## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.

Start=0 (리스트 인덱스)

End=8

5	9	10	12	16	18	21	27	29
---	---	----	----	----	----	----	----	----



Median(중앙값) =  $(0+8)/2 = 4$

16 < 21      오른쪽을 탐색!

리스트: [ 21 11 18 ]

출력: [ ]

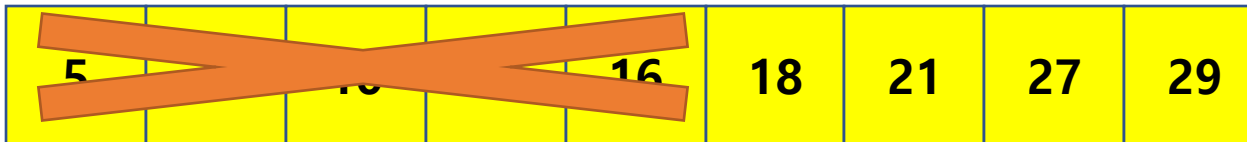
## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.

기존 Start=0

새 Start=5

End=8



The diagram shows a horizontal array of 10 yellow cells. The first four cells are crossed out with a large orange 'X'. The values in the cells are 5, 10, 16, 18, 21, 27, and 29. A blue arrow points from the text '(새 Start) = 5' to the cell containing the value 18.

5	10	16	18	21	27	29
---	----	----	----	----	----	----

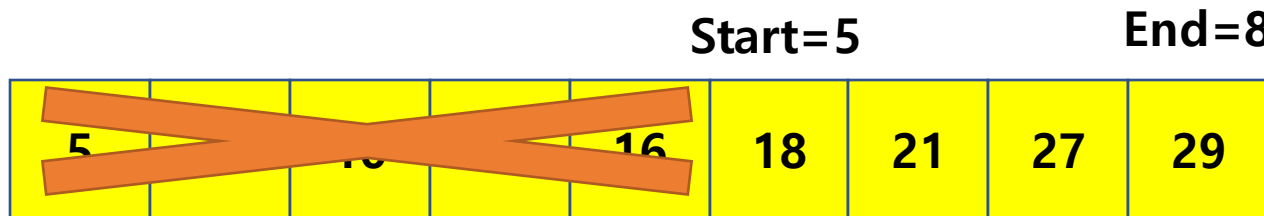
리스트: [ 21 11 18 ]

출력: [ ]

(새 Start) = 5 → index가 5보다 작은 원소들은 탐색하지 않아도 됨!

## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.



리스트: [ 21 11 18 ]

Median(중앙값) =  $(5+8)//2 = 6$

출력: [ 6 ]

왼쪽을 탐색!

List[6] = 21

## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.

Start=0

End=8

5	9	10	12	16	18	21	27	29
---	---	----	----	----	----	----	----	----



Median=4

왼쪽을 탐색!  $16 > 11$

리스트: [ 21 11 18 ]

출력: [ 6 ]

## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.

Start=0

End=3

5	9	10	12	16	21	29
---	---	----	----	----	----	----



Median=1

9 < 11    오른쪽을 탐색!

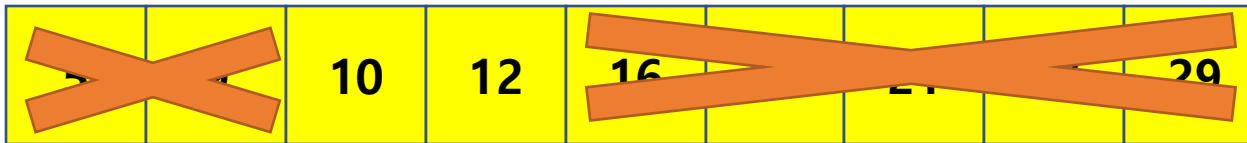
리스트: [ 21 11 18 ]

출력: [ 6 ]

## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.

Start=2 End=3



Median=2

오른쪽 탐색!  $10 < 11$

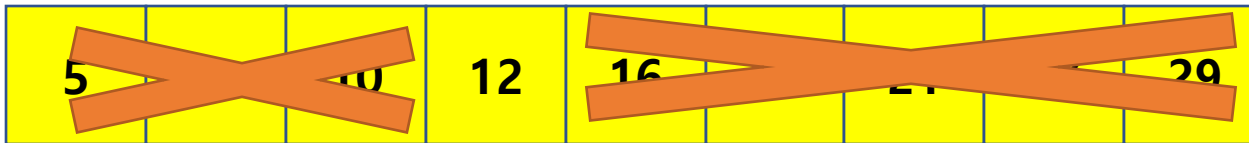
리스트: [ 21 11 18 ]

출력: [ 6 ]

## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.

Start=End=3



Median=3

왼쪽 탐색!  $12 > 11$

리스트: [ 21 11 18 ]

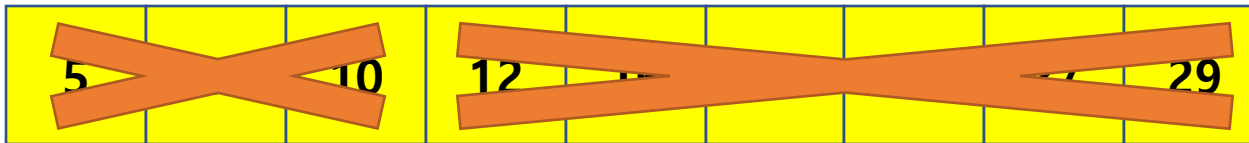
출력: [ 6 ]



## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.

End=2 Start=3



존재하지 않음. -1 리턴

리스트: [ 21 11 18 ]

출력: [ 6 -1 ]

## 02. 이진 탐색

(1) 정렬된 리스트와, (2) 찾고자 하는 원소들을 담은 리스트가 주어집니다.

5	9	10	12	16	18	21	27	29
---	---	----	----	----	----	----	----	----

리스트: [ 21 11 18 ]

최종 결과      출력: [ 6 -1 5 ]

## 02. 이진 탐색

**Note:** 각 테스트 케이스마다, **첫번째 줄**로 주어지는 리스트는 항상 오름차순으로 정렬돼 있음

5	9	10	12	16	18	21	27	29
---	---	----	----	----	----	----	----	----

(O)

5	10	9	12	16	18	21	27	29
---	----	---	----	----	----	----	----	----

(X) 이런 경우는 발생하지 않음

그렇기에, 중앙값을 기준으로 왼쪽 부분이나 오른쪽 부분 하나만 재귀적으로 조사하면 됨!

## 02. 이진 탐색

**Note:** 각 테스트 케이스마다, **첫번째 줄**로 주어지는 리스트는 항상 오름차순으로 정렬돼 있음

5	9	10	12	16	18	21	27	29
---	---	----	----	----	----	----	----	----

(O)

5	10	9	12	16	18	21	27	29
---	----	---	----	----	----	----	----	----

(X) 이런 경우는 발생하지 않음

하지만, 두번째 줄로 주어지는 리스트는 그렇지 않을 수도 있음!

리스트: [ 9 21 18 ] (O)

리스트: [ 9 18 21 ] (O) 문제 없음