컴퓨팅사고력 향상을 위한

문제해결과 알고리즘

1교시 : 자료탐색 알고리즘 2

양 숙 희

자료 탐색 알고리즘

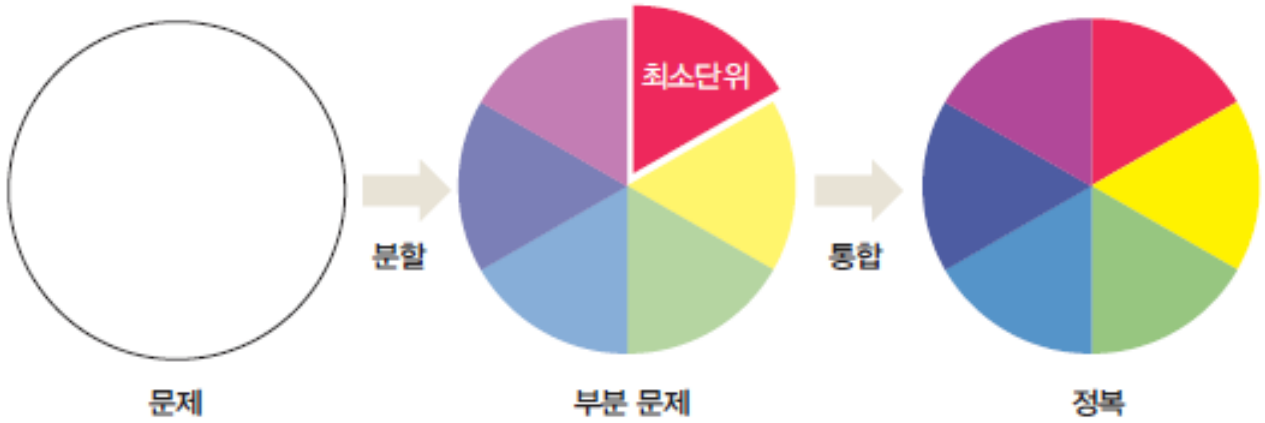
학습목표

1. 분할 정복 알고리즘의 개념을 이해 할 수 있다.

2. 이진탐색 알고리즘의 개념을 이해하고 활용할 수 있다.

분할 정복 알고리즘이란

- Divide and Conquer
- 주어진 문제를 최소 단위로 분할하여 문제를 해결하는 방식
- 즉, 작은 단위의 문제로 분할(Divide)한 후, 작은 단위의 문제를 풀면서 결국에는 전체 문제를 해결(Conquer)하는 문제해결 방법



실생활에서 분할 정복 알고리즘 활용

문제 상황 : 400명의 유치원생을 위하여 포장한 동일한 크리스마스 선물 가운데 선물이 들어가 있지 않은 선물 상자 찾아내야 함

실생활에서 분할 정복 알고리즘 활용

- 해결 방법 1 : 단순하게 문제 풀기
  - 한 개씩 비교
  - 최악의 경우 399번의 비교

- 해결 방법 2 : 분할 정복 알고리즘
  - ½ 로 나누어 비교

이진 탐색(BINARY SEARCH)

- 대표적인 분할 정복 알고리즘
- 이미 정렬된 자료를 대상으로 특정 값을 찾고자 할 때 중간 값과 찾고자 하는 값을 우선 비교한다.
- 중간 값보다 찾고자 하는 값이 특정 값이 작은 경우는 중간 값보다 뒤쪽에 위치한 모든 자료를 검토 대상에서 제외하고 중간 값보다 앞쪽에 위한 자료를 대상으로 탐색한다. 반복하여 중간 값과 비교하며 탐색 범위를 줄여간다.
- 이진 탐색을 실시할 때는 가장 작은 값, 중간 값, 가장 큰 값을 갖는 인덱스(index) 번호를 사용한다.
- 이진 탐색 알고리즘의 탐색 효율은 순차 탐색보다 뛰어나다.

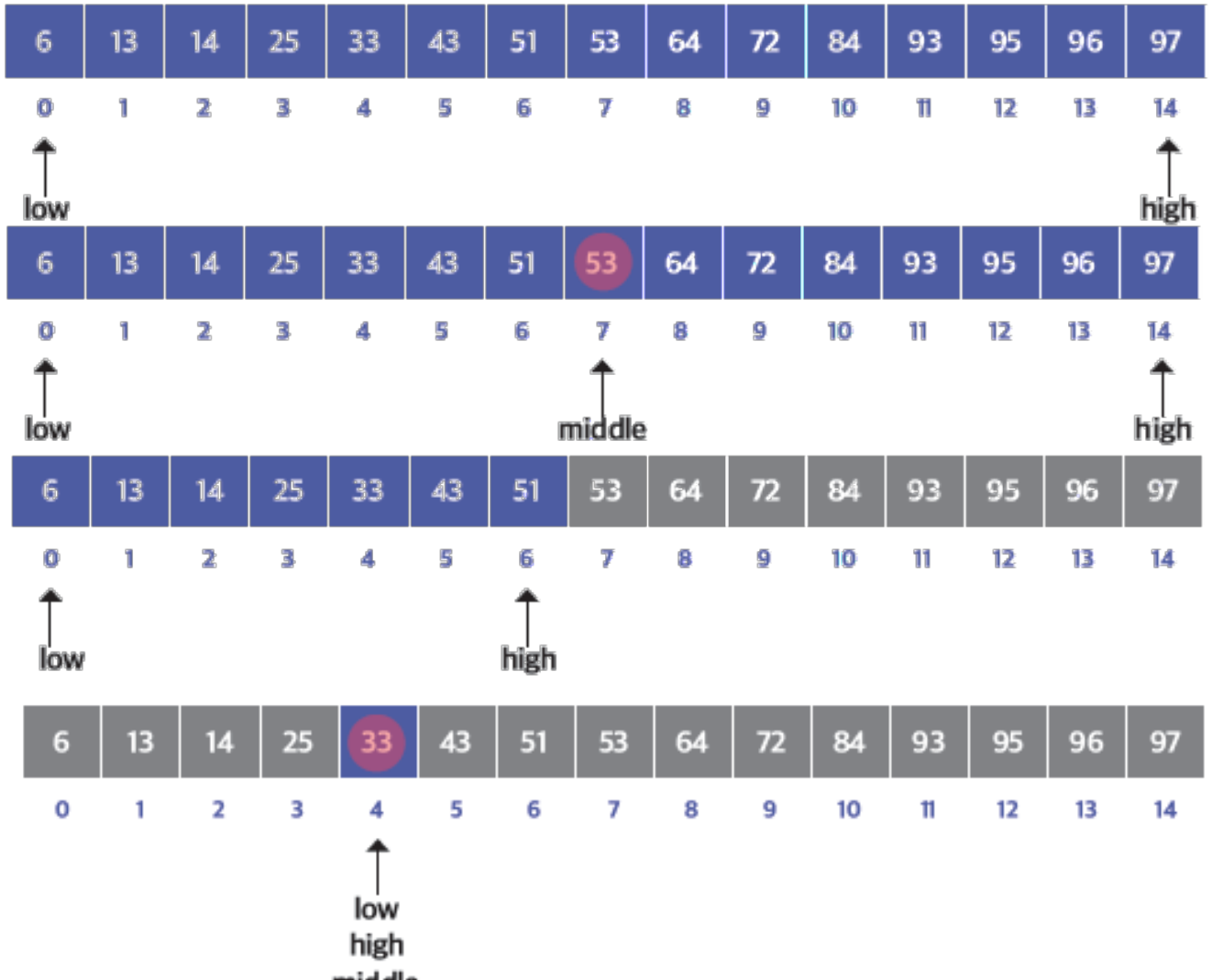
이진 탐색

Binary Search

1. 이진 탐색을 위하여 정렬된 자료를 리스트에 저장한다.
2. 탐색하고자 하는 특정 값을 변수에 저장한다.
3. 리스트 내의 중간 값을 찾는다.
4. 중간 값과 변수에 저장된 특정 값을 비교한다.
5. 만일 중간 값이 변수에 저장된 특정 값보다 작은 경우, 중간 값에서부터 마지막 값의 범위 안에서 특정 값을 비교한다. 만일 중간 값이 변수에 저장된 특정 값보다 큰 경우, 첫 번째 값에서부터 중간 값의 범위 안에서 특정 값을 비교한다.
6. 위의 단계를 반복하여 탐색하고자 하는 특정 값의 인덱스(index) 번호를 반환하여 문제를 해결한다.

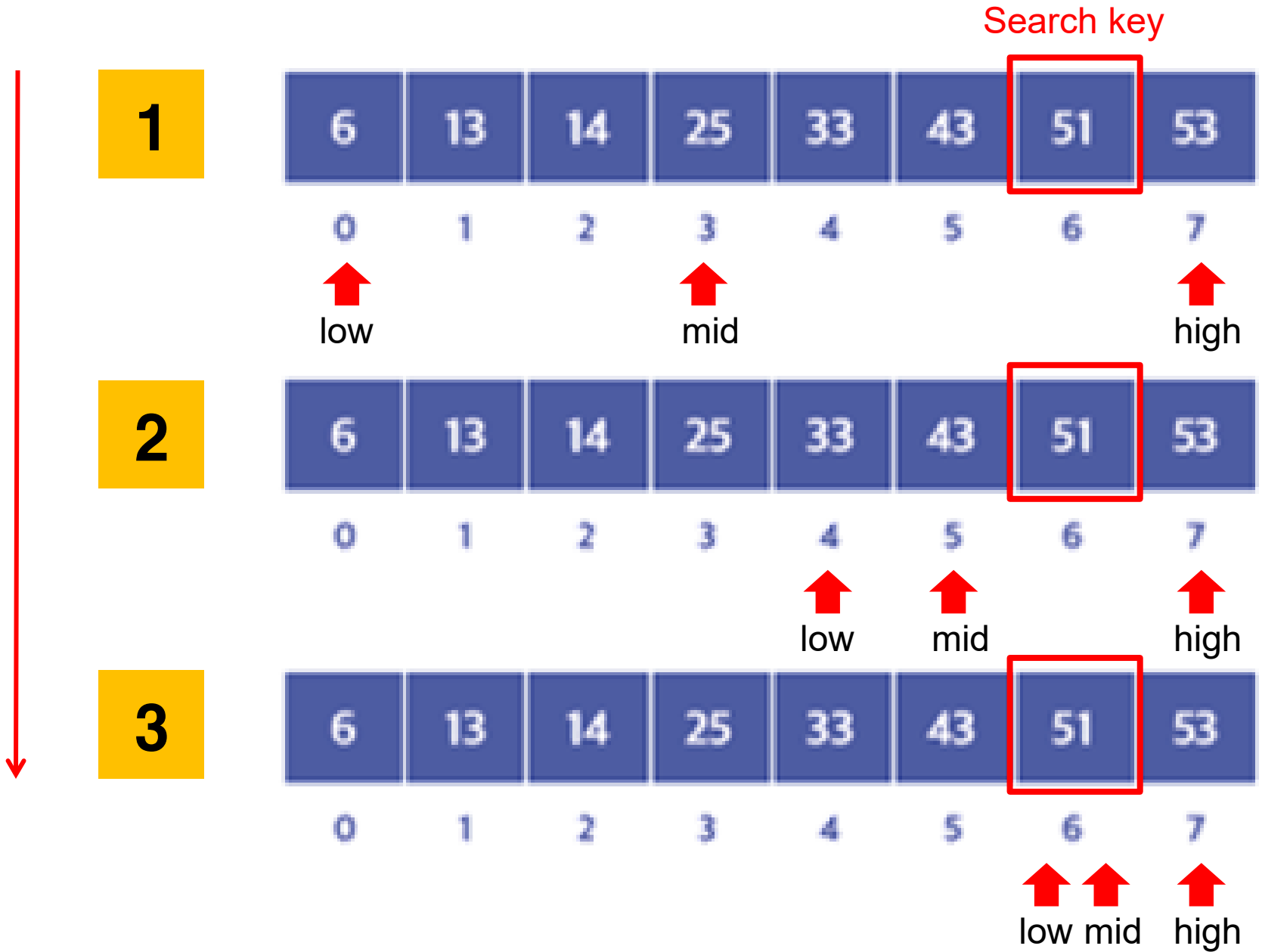
이진 탐색의 예

a = [6, 13, 14, 25, 33, 43, 51, 53, 64, 72, 84, 93, 95, 96, 97]
에서, 특정 값 33을 이진 탐색 방법으로 찾기

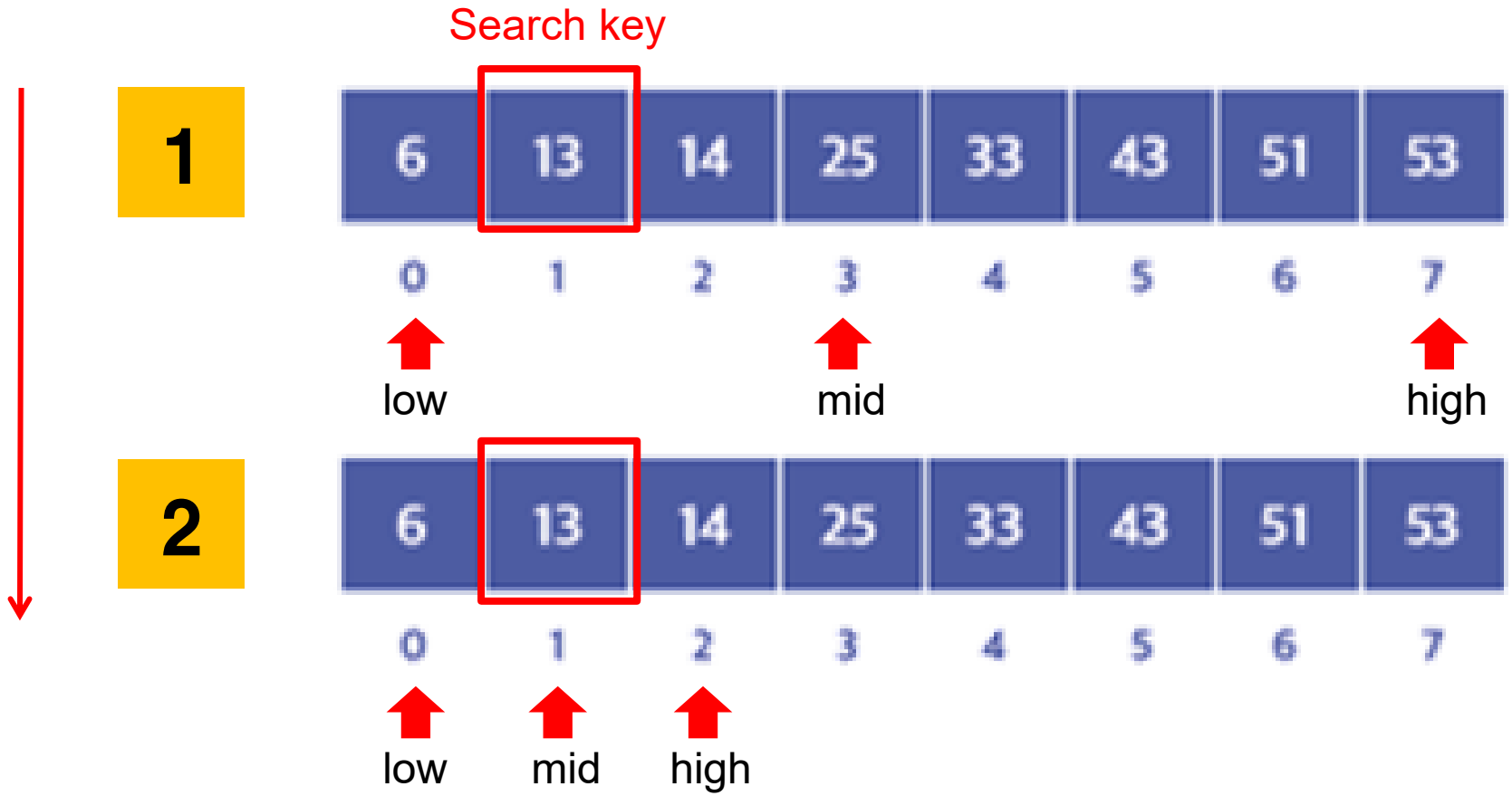


< … 생략

이진탐색과정


Search key
1
6 13 14 25 33 43 51 53
0 1 2 3 4 5 6 7
low
mid
high
2
6 13 14 25 33 43 51 53
0 1 2 3 4 5 6 7
low
mid
high
3
6 13 14 25 33 43 51 53
0 1 2 3 4 5 6 7
low mid
high

이진탐색과정


Search key
1
6
13
14
25
33
43
51
53
0
1
2
3
4
5
6
7
low
mid
high
2
6
13
14
25
33
43
51
53
0
1
2
3
4
5
6
7
low
mid
high

이진 탐색 파이썬 코드

data [6, 13, 14, 25, 33, 43, 51, 53]에서, 특정 값 51을 이진 탐색 방법으로 찾기

```
cnt=0 # 교환횟수
data=[13, 33, 53, 25, 6, 43, 51, 14]

print(data)

key=int(input('seach key : '))
result=

if result==-1:
    print('탐색실패')
else:
    print('%d번에 탐색성공' % cnt)
```

```
def binarySearch(key, data):

    low = 0
    high
    while low <= high :
        mid =                                    #
        cnt += 1
        if key < data[mid]:
            high = mid - 1
        elif key > data[mid]:
            low = mid + 1
        else:
            return mid

    return -1
```

지역변수와 전역변수

- 지역변수 : 함수 안에서 정의된 변수
- 전역변수 : 함수 밖에서 정의된 변수
- 전역변수는 프로그램 어디서든 읽을 수 있다.
- 단, 함수 안에서 전역변수를 수정할 수는 없다. 꼭 필요하다면 **global**문을 이용

```
# 전역변수와 지역변수

def stamp():
    global cnt
    cnt=cnt+1
    print(cnt)

# 메인코드

cnt=0 # 전역변수
stamp()
stamp()
```

자료 탐색 알고리즘

분할 정복 알고리즘이란

이진 탐색이해

- 알고리즘
- 파이썬 코드

수고하셨습니다.