이진 탐색(Binary Search)

이진 탐색

오름차순으로 정렬된 배열에서 원하는 숫자를 찾는 알고리즘이다.

1. 배열 전체의 중간값을 target값과 비교
2. 중간값이 target 값보다 크면 왼쪽 부분만 선택
3. 왼쪽 부분의 중간값을 다시 target과 비교



Python 코드

정방향과 재귀를 이용한 2가지 풀이 방법이 있다.

정방향

target = 25

m\_list = [30, 94, 27, 92, 21, 37, 25, 47, 25, 53, 98, 19, 32, 32, 7]

length = len(m\_list)

m\_list.sort()

left = 0

right = length-1

while left<=right:

mid = (left+right)//2

if m\_list[mid] == target:

print(mid+1)

break

elif m\_list[mid]>target:

right = mid-1

else :

left = mid+1

재귀

def binarySearch(array, target, left, right):

middle\_idx = (left+right)//2

print(middle\_idx)

middle = array[middle\_idx]

if target == middle:

print('answer {}'.format(middle\_idx))

elif middle > target:

binarySearch(array, target,left,middle\_idx-1)

elif middle < target:

binarySearch(array, target,middle\_idx+1,right)

else:

return False

target = 25

m\_list = [30, 94, 27, 92, 21, 37, 25, 47, 25, 53, 98, 19, 32, 32, 7]

length = len(m\_list)

m\_list.sort()

left = 0

right = length-1

binarySearch(m\_list,target,0,right)

유용한 점

반을 버리면서 탐색을 하기 때문에 시간복잡도를 줄일 수 있다.

첫 시행 : 

두번째 시행 : 

세번째 시행 : 

K번째 시행 : 

계속해서 탐색을 반복하면 끝나는 시점에는 남은 자료가 1개 된다.



* 

K의 의미는 시행 횟수입니다 따라서

자료의 개수 N이면 시행 횟수는 이 나옵니다.

시간복잡도를 줄일 수 있는 유용한 방법입니다.

참고한 사이트

<https://velog.io/@madfinger/Binary-Search%EC%9D%B4%EC%A7%84-%ED%83%90%EC%83%89-%ED%8C%8C%EC%9D%B4%EC%8D%AC>

https://jwoop.tistory.com/9