노트북: 첫 번째 노트북
만든 날짜: 2020-11-12 오후 5:32
URL: https://continuous-development.tistory.com/189

Algorithm

[Algorithm] 26강 : 이진 탐색 알고리즘 정의와 구현

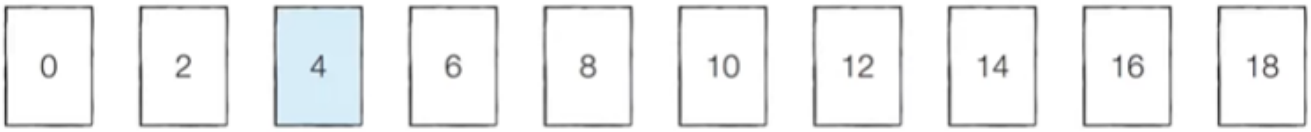
2020. 11. 12. 17:32 수정 삭제 공개

이진 탐색이란?

순차 탐색 : 리스트 안에 있는 특정한 데이터를 찾기 위해 앞에서부터 데이터를 하나씩 확인
이진 탐색 : 정렬되어 있는 리스트에서 탐색 범위를 절반씩 좁혀가며 데이터를 탐색
(이진 탐색은 시작점, 끝점, 중간점을 이용하여 탐색 범위를 설정)

이진 탐색 동작 예시

데이터 중에서 값이 4인 원소를 찾는 예시

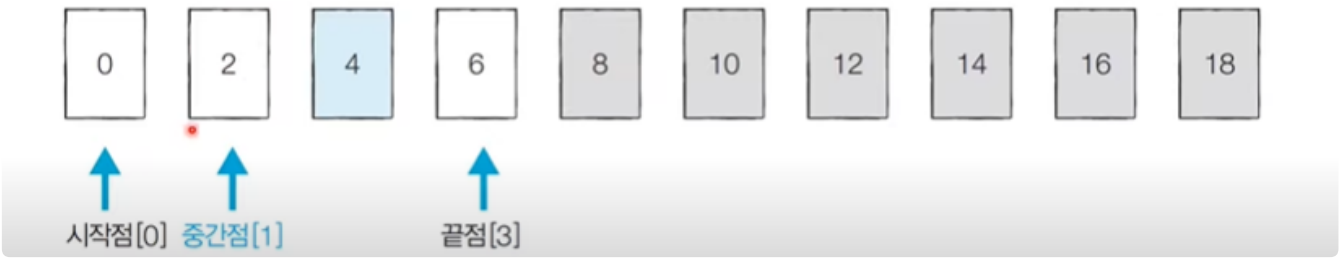


[Step 1]

- **[Step 1]** 시작점: 0, 끝점: 9, 중간점: 4 (소수점 이하 제거)

첫 번째 인덱스와 마지막 인덱스를 찾고 중간 점을 찾는다. 이 때 왼쪽인지 오른쪽인지를 찾는다.

[Step 2]



그중에 왼쪽이라고 하면 중간 값을 end로 잡고 다시 중간점을 찾는다. 그때 다시 찾는 값이 어디인지 찾는다.

[Step 3]

이 것을 반복하여 해당 값을 찾는다.

이진 탐색의 시간 복잡도

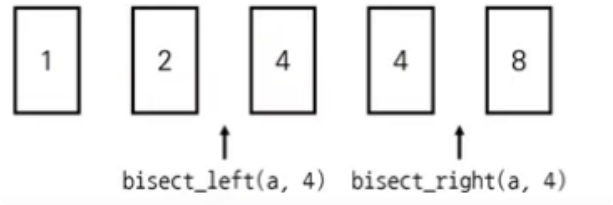
단계마다 탐색 범위를 2로 나누는 것과 동일하므로 연산 횟수는 log 2의 N 값에 비례
예를 들어 초기 데이터 개수가 32개 일 때, 이상적으로 1단계를 거치면 16개의 데이터가 남는다.
2단계를 거치면 8개
3단계를 거치면 4개의 데이터
이렇게 이진 탐색 범위를 절반씩 줄이며, 시간 복잡도는 O(logN)을 보장

이진 탐색 소스코드 : 재귀적 구현 (python)

```
def binary_search(array, target, start, end):
    if start > end:
        return None
    mid = (start + end) // 2

    # 값을 찾은 경우 인덱스 반환
    if array[mid] == target:
        return mid

    # 중간점의 값보다 찾고자 하는 값이 작은 경우 왼쪽 확인
    elif array[mid] > target:
        return binary_search(array, target, start, mid -1)

    # 중간점의 값보다 찾고자 하는 값이 큰 경우 오른쪽 확인
    else:
        return binary_search(array, target, mid+1, end)

# n(원소의 개수)과 target(찾고자 하는 값)을 입력받기
n , target = list(map(int, input().split()))

# 전체 원소 입력받기
array = list(map(int, input().split()))

# 이진 탐색 수행 결과 출력
result = binary_search(array,target, 0 , n-1)
if result == None:
    print('원소가 존재하지 않는다.')
else :
    print(result + 1)

=> 10 7 입력
=> 1,3,5,7,9,11,13,15,17,19
=> 4
```

파이썬 이진 탐색 라이브러리

bisect_left(a, x) : 정렬된 순서를 유지하면서 배열 a에 x를 삽입할 가장 왼쪽 인덱스를 반환
bisect_right(a, x): 정렬된 순서를 유지하면서 배열 a에 x를 삽입할 가장 오른쪽 인덱스를 반환




```
from bisect import bisect_left, bisect_right

a = [1,2,4,4,8]
x = 4

print(bisect_left(a,x))
=>2

print(bisect_right(a,x))
=>4
```

값이 특정 범위에 속하는 데이터 개수 구하기

```
from bisect import bisect_left, bisect_right

# 값이 [left_value, right_value]인 데이터의 개수를 반환하는 함수

def count_by_range(a, left_value, right_value):
    right_indext = bisect_right(a, right_value)
    left_index = bisect_left(a,left_value)
    retrun right_index - left_index

# 배열 선언
a = [1, 2, 3, 3, 3, 3, 4, 4, 8, 9]

#값이 4인 데이터 개수 출력
print(count_by_range(a,4,4))
```

```
=>2

#값이 [-1,3] 범위에 있는 데이터 개수 출력
print(count_by_range(a,-1,3))
=>6
```

www.youtube.com/watch?v=m-9pAwq1o3w&list=PLRx0vPvlEmdAghTr5mXQxGpHjWqSz0dgC

이 자료는 동빈 나 님의 이코 테 유튜브 영상을 보고 정리한 자료입니다.

'Algorithm' 카테고리의 다른 글□

[Algorithm] 26강 : 이진 탐색 알고리즘 정의와 구현□

[Algorithm] 25강 : 정렬 알고리즘 복잡도 비교 및 기본 문제□

[Algorithm] 24강 : 계수 정렬의 정의와 구현코드□

[Algorithm] 23강 : 퀵(quick) 정렬의 정의와 구현코드□

[Algorithm] 22강 : 삽입 정렬의 정의와 구현코드□

[Algorithm] 21강 : 선택 정렬의 정의와 구현코드□

python 이진 탐색 이진 탐색 이진탐색 알고리즘 파이썬 이진 탐색

나아무늘보

혼자 끄적끄적하는 블로그 입니다.