

아래는 이진 탐색 알고리즘을 사용한 예시 코드입니다.

초기에 `start_index` 는 0 으로, `end_index` 는 `len(some_list) - 1` 로 지정해줍니다. 그리고 리스트가 이미 정렬되어 있다는 사실을 활용하여 `start_index` 가 `end_index` 보다 작을 때까지, 아래의 내용을 반복합니다.

1. `element` 가 `some_list[midpoint]` 와 일치한다면, 그 인덱스를 리턴하고 종료합니다.
2. `element` 가 `some_list[midpoint]` 보다 작다면, 탐색 범위의 후반부는 제외시켜도 됩니다. 따라서 `end_index` 를 `midpoint - 1` 로 업데이트해줍니다.
3. `element` 가 `some_list[midpoint]` 보다 크다면, 탐색 범위의 전반부는 제외시켜도 됩니다. 따라서 `start_index` 를 `midpoint + 1` 로 업데이트해줍니다.

반복문이 종료될 때까지 원소를 찾지 못하면, `None` 의 값을 리턴해줍니다.

```
def binary_search(element, some_list):
    start_index = 0
    end_index = len(some_list) - 1

    while start_index <= end_index:
        midpoint = (start_index + end_index) // 2
        if some_list[midpoint] == element:
            return midpoint
        elif element < some_list[midpoint]:
            end_index = midpoint - 1
        else:
            start_index = midpoint + 1


    return None


print(binary_search(2, [2, 3, 5, 7, 11]))
print(binary_search(0, [2, 3, 5, 7, 11]))
print(binary_search(5, [2, 3, 5, 7, 11]))
print(binary_search(3, [2, 3, 5, 7, 11]))
print(binary_search(11, [2, 3, 5, 7, 11]))
```

```
0
None
2
1
4
```



수업을 완료하셨으면 체크해주세요.

 수강생 Q&A 보기

 질문하기

assignment_id=410&sort_by=popular)
(/questions/new?

assignment_id=410&op1=%ED%94%84%EB%A1%9C%EA%B7%

< 이전 강의 (/assignments/403) 다음 강의 (/assignments/404)
이진 탐색 - 반복문 + %EB%B0%98%EB%B0%B5%EB%A1%9C%EA%B7%BB+ %28%ED%95%B4%EC