



13 مارس، 2025، 1:23 ص

تحدي 4

لحل مشكلة "البحث عن موقع الهدف في قائمة مرتبة"،
استخدام خوارزمية البحث الثنائي، التي تتمتع بتعقيد
زمني $O(\log n)$. تنفيذ ذلك بلغة Python:

```
python``  
:def search_insert(nums, target)  
    left, right = 0, len(nums)  
  
    :while left < right  
        mid = (left + right) // 2  
  
        :if nums[mid] < target  
            left = mid + 1  
        :else  
            right = mid  
  
    return left
```

أمثلة على الاستخدام

2 # الناتج: print(search_insert([1, 3, 5, 6], 5))

1 # الناتج: print(search_insert([1, 3, 5, 6], 2))

"""





الملاحظات >

```
print(search_insert([1, 3, 5, 6], 7))  
# الناتج: 4  
````
```

## ### الشرح:

1. **\*\*تحديد الحدود\*\***: نبدأ بتحديد الحدود ``left`` و ``right`` التي تمثل بداية ونهاية القائمة.
2. **\*\*البحث الثنائي\*\***: نقوم بحساب العنصر المتوسط (``mid``) من القائمة.
  - إذا كان العنصر في الموضع ``mid`` أقل من ``target`` ، فإننا نبحث في النصف الأيمن (نحدث ``left``).
  - إذا كان العنصر في الموضع ``mid`` أكبر أو يساوي ``target`` ، فإننا نبحث في النصف الأيسر (نحدث ``right``).
3. **\*\*إرجاع الموضع\*\***: عندما ينتهي البحث، فإن ``left`` سيكون هو الموضع الذي يحتاج إليه ``target`` للحفاظ على ترتيب القائمة.

## ### المميزات:

- هذه الخوارزمية فعالة جدًا وتعمل بشكل جيد مع القوائم الكبيرة.

