


Intersection of Two Arrays

# Index	349
📅 CreatedAt	@September 28, 2022
👤 Person	 Ally Hyeseong Kim
🌟 Status	Done
🏷️ Tags	Binary Search Python
📅 UpdatedAt	@September 28, 2022

References


LeetCode - The World's Leading Online Programming Learning Platform
Level up your coding skills and quickly land a job. This is the best place to expand your knowledge and get prepared for your next interview.

 <https://leetcode.com/problems/search-in-rotated-sorted-array/>



파이썬 알고리즘 인터뷰

2021 세종도서 학술부문 선정작. 현업과 실무에 유용한 주요 알고리즘 이론을 깊숙이 이해하고, 파이썬의 핵심 기능과 문법까지 상세하게 이해할 수 있는 취업용 코딩 테스트를 위한 완벽 가이드다. 200여 개가 넘는...

 <https://www.aladin.co.kr/shop/wproduct.aspx?ItemId=245495826>



References

1. [heapq](#)
2. [Brute Force](#)
3. [Binary Search](#)
4. [Two pointer](#)

1. heapq

```
class Solution:
    def intersection(self, nums1: List[int], nums2: List[int]) -> List[int]:
        nums1 = list(set(nums1))
        heapq.heapify(nums1)
        nums2 = list(set(nums2))
        heapq.heapify(nums2)
        answer = set()
        target = heapq.heappop(nums2)
        search = heapq.heappop(nums1)
        while True:
            if search == target:
                answer.add(search)
                if not nums1 or not nums2:
                    return list(answer)
                target = heapq.heappop(nums2)
                search = heapq.heappop(nums1)
            elif search > target:
                if not nums2:
                    return list(answer)
                target = heapq.heappop(nums2)
            elif search < target:
```

```

        if not nums1:
            return list(answer)
        search = heapq.heappop(nums1)

```

2. Brute Force

```

class Solution:
    def intersection(self, nums1: List[int], nums2: List[int]) -> List[int]:
        result = set()
        for n1 in nums1:
            for n2 in nums2:
                if n1 == n2:
                    result.add(n1)

        return result

```

3. Binary Search

```

class Solution:
    def intersection(self, nums1: List[int], nums2: List[int]) -> List[int]:
        result = set()
        nums2.sort()
        for n1 in nums1:
            i2 = bisect.bisect_left(nums2, n1)
            if len(nums2) > 0 and len(nums2) > i2 and n1 == nums2[i2]:
                result.add(n1)

        return result

```

4. Two pointer

```

class Solution:
    def intersection(self, nums1: List[int], nums2: List[int]) -> List[int]:
        result = set()
        nums1.sort()
        nums2.sort()
        i = j = 0

        while i < len(nums1) and j < len(nums2):
            if nums1[i] > nums2[j]:
                j += 1
            elif nums1[i] < nums2[j]:
                i += 1
            else:
                result.add(nums1[i])
                i += 1
                j += 1

        return result

```