Longest Consecutive Subsequence Problem - Notes

Problem:
Given an unsorted array of integers, find the length of the longest subsequence of consecutive integers.

Approach 1: Sorting
1. Sort the array.
2. Traverse through the sorted array.
3. Count the length of consecutive increasing elements.
4. Skip duplicates while counting.
5. Keep track of the maximum streak length.

Time Complexity: O(n log n) (due to sorting)
Space Complexity: O(1) (if sorting in-place)

```
class Solution:
    def longestConsecutive(self, arr):
        if not arr:
            return 0

        arr.sort()
        maxSequence = 1
        count = 1

        for i in range(1, len(arr)):
            if arr[i] == arr[i-1] + 1:   # consecutive
                count += 1
            elif arr[i] != arr[i-1]:     # skip duplicates
                maxSequence = max(maxSequence, count)
                count = 1

        return max(maxSequence, count)
```

Approach 2: Hashing (Using Set)
1. Store all numbers in a set for O(1) lookups.
2. Iterate through the set.
3. Only start counting from numbers that are the beginning of a sequence (num-1 not in set).
4. Count forward until sequence breaks.
5. Keep track of the maximum streak length.

Time Complexity: O(n)
Space Complexity: O(n) (for the set)

```
class Solution:
    def longestConsecutive(self, arr):
        if not arr:
            return 0

        numSet = set(arr)
        maxSequence = 0

        for num in numSet:
            if num - 1 not in numSet:  # start of sequence
                count = 1
                current = num
                while current + 1 in numSet:
                    current += 1
                    count += 1
                maxSequence = max(maxSequence, count)

        return maxSequence
```