



Solution Review: A List as a Subset of Another List

This review provides a detailed analysis of the solution to the A List as a Subset of Another List Challenge.

We'll cover the following



- Solution: Lookup in a Hash Table
- Time Complexity

Solution: Lookup in a Hash Table

```
1 def is_subset(list1, list2):
2     s = set(list1) # Create a set with list1 values
3     # Traverse list 2 elements
4     for elem in list2:
5         # Return false if an element not in list1
6         if elem not in s:
7             return False
8     # Return True if all elements in list1
9     return True
10
11
12 list1 = [9, 4, 7, 1, -2, 6, 5]
13 list2 = [7, 1, -2]
14 list3 = [10, 12]
15 print(is_subset(list1, list2))
16 print(is_subset(list1, list3))
17
```





The solution is very simple when working with the Pythonic hash table `Set`. We simply iterate over `list2` and `list3` to see whether their elements can be found in `list1`.

At the back end, the values are checked against their hashed indices in `list1`.

Time Complexity

For a lookup list with **m** elements and a subset list with **n** elements, the time complexity is $O(m+n)$.

Interviewing soon? We've partnered with Hired so that companies apply to you instead of you applying to them. [See how](#) ⓘ

[← Back](#)[Next →](#)

Challenge 1: A List as a Subset of Ano...

Challenge 2: Check if Lists are Disjoint



Mark as Completed



Report an Issue

