Performance by Method

|  |  |  |
| --- | --- | --- |
| **Method** | **Linked List** | **Array** |
| Add | O(n) | O(n) |
| Remove | O(n) | O(2n) |
| Cardinality | O(1) | O(1) |
| Union | Equal Performance | |
| Difference | Equal Performance | |
| Intersection | Equal Performance | |
| hasElement | O(n) | O(n) |
| isSubset | O(n2) | O(n2) |

I understand that normally an Array would be better at specific referencing compared to linked lists but in return would be less efficient at adding or removing elements. The reason why they have the same performance for these classes is that because they are both implementations of sets, order doesn’t matter so adding and removing elements is done by simply adding to the end of the set or when removing, copying the last value of the set to the position where the target element to be removed is located. The only time where the array would be less efficient in this case would be when the array becomes full, because then we must create a new list of a bigger size and copy everything over, whereas a linked list has no maximum size.