The ArrayList underlying layer uses arrays to store elements. Arrays are contiguous memory spaces. Data can be obtained through indexes. The time complexity is O(1), so it is fast.

The underlying layer of LinkedList is a two-way linked list. Use for-loop to implement traversal, you need to start from the [head node of the linked list](https://medium.com/javarevisited/top-20-linked-list-coding-problems-from-technical-interviews-90b64d2df093) every time. The time complexity is O (n\*n).

1. The for loop method is faster when using [ArrayList](https://javarevisited.blogspot.com/2011/05/example-of-arraylist-in-java-tutorial.html" \t "_blank) because the for-each is implemented by the iterator, and it needs to perform concurrent modification verification.
2. When using [LinkedList](https://javarevisited.blogspot.com/2012/12/how-to-initialize-list-with-array-in-java.html), the for-each is much faster than the for-loop, because LinkedList is implemented by using a two-way linked list. Each addressing needs to start from the header node. If we need to traverse LinkedList, we need to avoid using the for-loop.
3. Using the iterator pattern, for-each does not need to care about the specific implementation of the collection. If there is a need to replace the collection, it can be easily replaced without modifying the code.