List and Arraylist

List interface in Java is a part of the java.util package and is a sub-interface of the collection interface. It provides a way to store an ordered collection of elements(Known as sequence). List allows for precise control over where elements are inserted and can contain duplicate elements.

Note: List will extend Collections. So the list has all the functionalities of the Collection.

List interface is implemented by several classes in the Java Collection Framework such as: Arraylist, LinkedList, Vector and Stack.

Key Features of List Interface:
- Order Preservation
- Indexed-Based Access
- Allow Duplicates.

ArrayList: An Arraylist is a resizable array implementation of the list interface. Unlike arrays in Java, which have fixed size, an arraylist can change its size dynamically as elements are added or removed. This flexibility makes it a popular choice when no. of elements in a list isn't known in advance.

Functionalities like: add, add with index, get, set, size, contain, remove and many more will be implemented in code and take a reference of it.

Internal Working of Arraylist: Unlike regular arrays, which have fixed size, an arraylist can grow and shrink as elements are added or removed. This dynamic resizing is achieved by creating a new array when the current array is full and copying the elements to the new array.

Note: When you create an arraylist, it has an initial capacity (default is 10). The capacity refers to the size of the internal array that can hold elements before needing to resize.

Adding Element: When we add an element to an arraylist, the following steps occur.
- Check Capacity: Before adding the new element, Arraylist checks if there is enough space in the internal array(Element data array),. If the array is full, it needs to be resized.
- Resize if necessary: If the internal array is full, the arraylist will create a new array with larger capacity(usually 1.5 times of the current capacity) and copy the elements from the old array to the new array.
- Add the Element: The new element is then added to the internal array at the appropriate index, and the size is incremented.

Resizing the Elements:
Initial Capacity: By default, the initial capacity is 10. This means the internal array can hold 10 elements before it needs to grow.

Growth factor: When the internal array is full, a new array is created with size 1.5 times the old array. This growth factor balances memory efficiency and resizing cost.
Copying Elements: When resizing occurs, all the elements from the old array are copied to the new array, which is an O(n) operation, where n is the number of elements in the arraylist.

Removing Elements:
- Check Bounds: The arraylist first checks if the index is within the valid range.
- Remove the element: The element is removed, and all elements to the right are removed element are shifted one position to the left to fill the gap
- Reduce the size: The size is decremented by 1.

Convert Arraylist to array: Take reference of code.

Time Complexity:
- Access by index - O(1)
- Adding an element - O(n)
- Removing element - O(n)
- Iteration - O(n)