**ArrayLists;**

1. \*\*Adding Elements:\*\*

- `add(element)`: Adds an element to the end of the `ArrayList`.

- `add(index, element)`: Inserts an element at a specific index in the `ArrayList`, shifting existing elements to accommodate the new element.

- `addAll(collection)`: Adds all elements from a collection to the end of the `ArrayList`.

2. \*\*Accessing Elements:\*\*

- `get(index)`: Retrieves the element at the specified index in the `ArrayList`.

- `indexOf(element)`: Returns the index of the first occurrence of the specified element in the `ArrayList`.

- `lastIndexOf(element)`: Returns the index of the last occurrence of the specified element in the `ArrayList`.

3. \*\*Updating Elements:\*\*

- `set(index, element)`: Replaces the element at the specified index in the `ArrayList` with a new element.

4. Removing Elements:\*\*

- `remove(index)`: Removes the element at the specified index from the `ArrayList` and shifts subsequent elements.

- `remove(element)`: Removes the first occurrence of the specified element from the `ArrayList`.

- `clear()`: Removes all elements from the `ArrayList`, leaving it empty.

5. \*\*Checking Size and Empty Status:\*\*

- `size()`: Returns the number of elements in the `ArrayList`.

- `isEmpty()`: Checks if the `ArrayList` is empty.

6. \*\*Checking Element Existence:\*\*

- `contains(element)`: Checks if the `ArrayList` contains the specified element.

**LinkedLists**

Same operations as arrayLists jst this few changes:

**Accessing Elements:**

* **getFirst()**: Retrieves the first element of the linked list.
* **getLast()**: Retrieves the last element of the linked list.
* **get(index)**: Retrieves the element at the specified index in the linked list.

**Removing Elements:**

* **removeFirst()**: Removes and returns the first element from the linked list.
* **removeLast()**: Removes and returns the last element from the linked list.
* **remove(index)**: Removes and returns the element at the specified index in the linked list.

**Arrays**

In Java, there are several operations commonly used to manipulate data in an array. Here are some of the most frequently used operations:

1. \*\*Accessing Elements:\*\*

- Accessing individual elements in an array is done using the square bracket notation `array[index]`, where `array` is the name of the array and `index` is the position of the element.

2. \*\*Updating Elements:\*\*

- To update an element in an array, assign a new value to the desired index using the assignment operator `=`.

3. \*\*Iterating through the Array:\*\*

- You can iterate through the elements of an array using loops such as `for` loop or `foreach` loop.

4. \*\*Copying Arrays:\*\*

- To copy the contents of one array into another, you can use the `System.arraycopy()` method or create a new array and manually copy the elements.

5. \*\*Sorting Elements:\*\*

- You can sort the elements of an array using the `Arrays.sort()` method, which implements a quicksort algorithm.

6. \*\*Finding Maximum and Minimum:\*\*

- To find the maximum or minimum element in an array, you can iterate through the array and keep track of the current maximum or minimum value.

7. \*\*Searching for an Element:\*\*

- To search for a specific element in an array, you can iterate through the array and check each element against the desired value.

8. \*\*Checking Length:\*\*

- The length of an array can be obtained using the `length` property, such as `array.length`.

**Stack(LIFO)**

1.Pushing Elements:

push(element): Inserts an element onto the top of the stack.

2.Popping Elements:

pop(): Removes the last element from in the stack.

3.Peeking Elements:

peek(): Returns the last element of the stack without removing it.

4.Checking if Stack is Empty:

isEmpty(): Checks if the stack is empty. Returns true if the stack is empty; otherwise, returns false.

5.Checking the Size of the Stack:

size(): Returns the number of elements in the stack.

**Queue(FIFO)**

Same operations as stack just these new parts:

1.Adding Elements:

offer(element): Adds an element to the end of the queue.

add(element): Adds an element to the end of the queue. Throws an exception if the operation fails.

2.Removing Elements:

poll(): Removes and returns the element at the front of the queue. Returns null if the queue is empty.

remove(): Removes and returns the element at the front of the queue. Throws an exception if the queue is empty.

3.Peeking Elements:

peek(): Returns the element at the front of the queue without removing it. Returns null if the queue is empty.

element(): Returns the element at the front of the queue without removing it. Throws an exception if the queue is empty.