Java Methods Cheat Sheet for DSA Problems

String Methods

Basic Operations

- `length()` – Returns the length (number of characters) in the string.
- `charAt(int index)` – Returns the character at the specified index.
- `substring(int beginIndex)` – Returns a new string starting from the specified index to the end.
- `substring(int beginIndex, int endIndex)` – Returns a new string from beginIndex to endIndex-1.
- `isEmpty()` – Returns true if the string length is 0.
- `toCharArray()` – Converts the string into a character array.

String Comparison

- `equals(Object obj)` – Compares the content of strings (case-sensitive).
- `equalsIgnoreCase(String str)` – Compares two strings ignoring case.
- `compareTo(String str)` – Lexicographically compares strings; returns an int.
- `startsWith(String prefix)` – Checks if the string starts with the given prefix.
- `endsWith(String suffix)` – Checks if the string ends with the given suffix.
- `contains(CharSequence seq)` – Checks if the string contains the specified sequence.

String Modification

- `toLowerCase()` – Converts all characters to lowercase.
- `toUpperCase()` – Converts all characters to uppercase.
- `trim()` – Removes leading and trailing whitespace.
- `replace(char oldChar, char newChar)` – Replaces all occurrences of oldChar.
- `replace(CharSequence target, CharSequence replacement)` – Replaces target sequence with replacement.
- `replaceAll(String regex, String replacement)` – Replaces matching substrings based on regex.

Searching

- `indexOf(String str)` – Returns index of first occurrence of str.
- `indexOf(String str, int fromIndex)` – Same as above, starting from fromIndex.
- `lastIndexOf(String str)` – Returns last occurrence of the specified string.
- `lastIndexOf(String str, int fromIndex)` – Last occurrence before a given index.
- `matches(String regex)` – Returns true if the string matches the given regex pattern.

Splitting and Joining

- `split(String regex)` – Splits string into array based on regex.
- `split(String regex, int limit)` – Splits with a limit on resulting array size.

- `join(CharSequence delimiter, CharSequence... elements)` – Joins elements with a delimiter.
- `concat(String str)` – Concatenates the specified string.

StringBuilder Methods

Basic Operations

- `StringBuilder()` – Creates an empty builder with capacity 16.
- `StringBuilder(String str)` – Initializes builder with the given string.
- `length()` – Returns the number of characters.
- `capacity()` – Returns the current capacity.
- `charAt(int index)` – Returns character at specified index.
- `substring(int start)` – Returns substring from start to end.
- `substring(int start, int end)` – Returns substring between given indices.

Modification

- `append(X x)` – Appends representation of x to builder.
- `insert(int offset, X x)` – Inserts representation at specified offset.
- `delete(int start, int end)` – Removes characters in the range.
- `deleteCharAt(int index)` – Deletes character at specified index.
- `replace(int start, int end, String str)` – Replaces substring with specified string.
- `setCharAt(int index, char ch)` – Sets character at index.
- `reverse()` – Reverses the character sequence.
- `setLength(int newLength)` – Changes the length (adds or truncates).
- `toString()` – Converts builder to string.

Arrays (java.util.Arrays)

Basic Operations

- `Arrays.toString(array)` – Returns string representation.
- `Arrays.deepToString(Object[][] array)` – For 2D arrays.
- `Arrays.equals(array1, array2)` – Compares two arrays.
- `Arrays.deepEquals(Object[][] a1, Object[][] a2)` – For 2D array equality.
- `Arrays.hashCode(array)` – Returns hashcode.
- `Arrays.deepHashCode(Object[][] array)` – Hashcode for 2D arrays.

Searching and Sorting

- `Arrays.sort(array)` – Sorts array in ascending order.
- `Arrays.sort(array, fromIndex, toIndex)` – Sorts a range in array.
- `Arrays.sort(array, Comparator c)` – Custom sort with comparator.
- `Arrays.binarySearch(array, key)` – Searches for key in sorted array.
- `Arrays.binarySearch(array, fromIndex, toIndex, key)` – Search within range.

- `Arrays.fill(array, val)` – Fills array with val.
- `Arrays.fill(array, fromIndex, toIndex, val)` – Fills range with val.

Array Conversion

- `Arrays.asList(T... a)` – Converts array to List.
- `Arrays.copyOf(original, newLength)` – Copies and resizes.
- `Arrays.copyOfRange(original, from, to)` – Copies specific range.

Java 8 Enhancements

- `Arrays.stream(array)` – Converts to Stream.
- `Arrays.parallelSort(array)` – Parallel sorting using threads.

ArrayList Methods

Basic Operations

- `ArrayList<>()` – Creates empty list.
- `ArrayList(Collection c)` – Creates list from collection.
- `size()` – Number of elements.
- `isEmpty()` – Checks if list is empty.
- `get(int index)` – Returns element at index.
- `set(int index, E element)` – Replaces element.
- `add(E e)` – Appends element.
- `add(int index, E element)` – Inserts at index.
- `remove(int index)` – Removes by index.
- `remove(Object o)` – Removes first occurrence.
- `clear()` – Removes all elements.

Searching

- `contains(Object o)` – Checks presence.
- `indexOf(Object o)` – First occurrence.
- `lastIndexOf(Object o)` – Last occurrence.

Bulk Operations

- `addAll(Collection c)` – Appends collection.
- `addAll(int index, Collection c)` – Inserts collection.
- `removeAll(Collection c)` – Removes all in c.
- `retainAll(Collection c)` – Retains only in c.
- `containsAll(Collection c)` – Checks all in c.

List Views

- `subList(fromIndex, toIndex)` – Returns portion.

Java 8 Enhancements

- `stream()` – Converts to Stream.
- `forEach(action)` – Applies action to elements.
- `removeIf(filter)` – Removes matching elements.
- `sort(comparator)` – Sorts list.

HashSet Methods

Basic Operations

- `HashSet<>()` – Creates empty set.
- `HashSet(Collection c)` – Initializes from collection.
- `size()` – Number of elements.
- `isEmpty()` – Checks emptiness.
- `add(E e)` – Adds if not already present.
- `remove(Object o)` – Removes specified object.
- `clear()` – Removes all.
- `contains(Object o)` – Checks existence.

Bulk Operations

- `addAll(Collection c)` – Adds all.
- `removeAll(Collection c)` – Removes all in c.
- `retainAll(Collection c)` – Keeps only in c.
- `containsAll(Collection c)` – Checks all present.

Iteration

- `iterator()` – Returns iterator.

Java 8 Enhancements

- `stream()` – Sequential stream.
- `forEach(action)` – Action per element.
- `removeIf(filter)` – Conditional removal.

HashMap Methods

Basic Operations

- `HashMap<>()` – Creates empty map.
- `HashMap(Map m)` – Initializes from map.
- `size()` – Number of mappings.
- `isEmpty()` – Checks if empty.
- `put(K key, V value)` – Adds/updates mapping.
- `get(Object key)` – Returns value or null.

- `getOrDefault(key, defaultValue)` – Value or default.
- `remove(Object key)` – Deletes key.
- `clear()` – Clears map.

Checking Map Contents

- `containsKey(Object key)` – Checks key.
- `containsValue(Object value)` – Checks value.

Bulk Operations

- `putAll(Map m)` – Adds all entries.
- `putIfAbsent(K key, V value)` – Adds if not present.

Map Views

- `keySet()` – Returns key view.
- `values()` – Returns value view.
- `entrySet()` – Returns entry view.

Java 8 Enhancements

- `forEach(action)` – Performs action on each entry.
- `replaceAll(function)` – Replaces all values.
- `compute(key, remappingFunction)` – Computes new value.
- `computeIfAbsent(key, mappingFunction)` – Computes if absent.
- `computeIfPresent(key, remappingFunction)` – Computes if present.
- `merge(key, value, remappingFunction)` – Merges values.

Queue and PriorityQueue Methods

Basic Operations

- `add(E e) / offer(E e)` – Inserts element.
- `remove() / poll()` – Removes head (poll returns null if empty).
- `element() / peek()` – Retrieves head (peek returns null).
- `size()` – Number of elements.
- `isEmpty()` – Checks if empty.

PriorityQueue Specific

- `PriorityQueue<>()` – Default min heap.
- `PriorityQueue(Comparator c)` – Custom comparator.

Stack Methods

Basic Operations

- `push(E item)` – Pushes item to top.
- `pop()` – Removes and returns top item.
- `peek()` – Returns top without removing.
- `empty()` – Checks if empty.
- `search(Object o)` – Returns 1-based position.

Deque Methods (LinkedList, ArrayDeque)

Basic Operations

- `addFirst(E e) / offerFirst(E e)` – Inserts at front.
- `addLast(E e) / offerLast(E e)` – Inserts at end.
- `removeFirst() / pollFirst()` – Removes from front.
- `removeLast() / pollLast()` – Removes from end.
- `getFirst() / peekFirst()` – Retrieves first.
- `getLast() / peekLast()` – Retrieves last.