**Stream**

java.util.stream

Input -> Operations -> Terminal Operation -> Output

**Stream.of(<arrayitems>);**

Or

**private static List<Employee> empList = Arrays.asList(arrayOfEmps);**

**empList.stream();**

**Stream.concat(stream1, stream2)**

Operations

**forEach | map | collect | filter | findFirst | toArray | flatMap | collect | limit | min | Max**

List<Employee> employees = Stream.of(empIds)

.**map**(employeeRepository::findById)

.**collect**(Collectors.toList());

Employee employee = Stream.of(empIds)

.map(employeeRepository::findById)

.filter(e -> e != null)

.filter(e -> e.getSalary() > 100000)

.**findFirst()**

**.orElse(null);**

Float totalPrice = productsList.stream()

.**map**(product->product.price)

.**reduce**(0.0f,(sum, price)->sum+price);

**Get index**

OptionalInt indexOpt = IntStream.range(0, users.size())

.filter(i -> searchName.equals(users.get(i)))

.findFirst();

// **Find duplicates**

list.stream()

// Count the frequency of each element

// and filter the elements

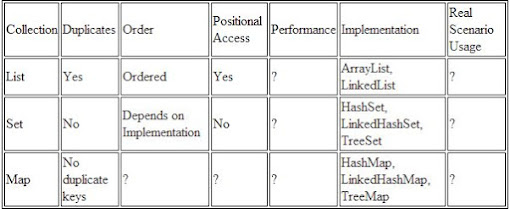
// with frequency > 1

.filter(i -> **Collections.frequency**(list, i) > 1)

// And Collect them in a Set

.collect(Collectors.toSet());

**Collection**



**Get sublist**

list.subList(2, 4); - *subLis t( fromIndex, toIndex)*

**String**

|  |  |
| --- | --- |
| **Character Extraction** | **StringBuffer** |
| char charAt(int where) | StringBuffer( ) |
| getChars( ) | StringBuffer(int size) |
| getBytes( ) | StringBuffer(String str) |
| toCharArray( ) | StringBuffer(CharSequence chars) |
| **String Comparison** | int length( ) |
| equals( ) and equalsIgnoreCase | int capacity( ) |
| regionMatches( ) |  |
| startsWith( ) and endsWith( ) | ensureCapacity( ) |
| compareTo( ) | setLength( ) |
| indexOf( ) - lastIndexOf( ) | charAt( ) and setCharAt( ) |
| substring( ) | getChars( ) |
| concat() | append( ) |
| replace() | insert() |
| trim() and strip() | reverse() |
| toUpperCase()  toLowerCase() | delete( ) and deleteCharAt( ) |
|  | replace( ) |
|  | substring( ) |

**Collections**

|  |  |
| --- | --- |
| iterator() | add(Object) |
| max() | addAll(Collection c) |
| parallelStream() | clear() |
| remove(Object o) | contains(Object o) |
| removeAll(Collection c) | containsAll(Collection c) |
| removeIf(Predicate filter) | equals(Object o) |
| retainAll(Collection c) | hashCode() |
| size() | isEmpty() |
| stream() | toArray() |

**Linkedlist**

|  |  |
| --- | --- |
| addFirst() | removeLast() |
| addLast() | getFirst() |
| removeFirst() | getLast() |

**Collections**

**-** Utility class

- Collections.addAll(arrlist, "web", "site");

- Collections.sort(arrlist);

Collections.**sort(al);**

int lastIndex = 0;

while (lastIndex != -1) {

lastIndex = mainString.indexOf(subString, lastIndex);

if (lastIndex != -1) {

System.out.println("Substring found at index: " + lastIndex);

lastIndex += subString.length();

}

}