Feature/Aspect	ArrayList	LinkedList	Vector	Stack	PriorityQueue	ArrayDeque	HashSet	LinkedHashSet	TreeSet
Ordering	Insertion order	Insertion order	Insertion order	LIFO (Last In First Out)	Natural order or custom comparator	Insertion order	No specific order	Insertion order	Sorted order (natural or comparator)
Duplicates	Allowed	Allowed	Allowed	Allowed	Not allowed	Allowed	Not allowed	Not allowed	Not allowed
Null elements	Allowed	Allowed	Allowed	Allowed	Not allowed	Not allowed	Allowed	Allowed	Not allowed
Performance	Fast random access, slow insert/remove at beginning or middle	Fast insert/remove at beginning or middle, slower random access	Synchronized, slower than ArrayList	Same as Vector	O(log n) for insertion and retrieval	Efficient at both ends, no indexed access	O(1) for basic operations	O(1) for basic operations	O(log n) for basic operations
Thread safety	Not thread-safe	Not thread-safe	Thread-safe	Thread-safe	Not thread-safe	Not thread-safe	Not thread-safe	Not thread-safe	Not thread-safe
Underlying structure	Resizable array	Doubly linked list	Resizable array	Resizable array	Binary heap	Resizable array	Hash table	Hash table with linked list	Red-Black tree
Iteration order	Maintains insertion order	Maintains insertion order	Maintains insertion order	Maintains insertion order (LIFO)	Natural order or custom comparator	Maintains insertion order	No order	Maintains insertion order	Sorted order
Key Operations	Fast random access, good for large reads	Fast inserts/removals at both ends	Similar to ArrayList but synchronized	Push, pop, peek, search	Offer, poll, peek	Add/remove at both ends, supports stack/queue	Add, remove, contains	Add, remove, contains	Add, remove, contains
Best use case	Frequent access by index, minimal inserts/removals	Frequent inserts/removals, minimal random access	Thread-safe version of ArrayList	LIFO stack operations	Priority-based ordering	Double-ended queue, stack/queue usage	Fast lookup, no duplicates, no order	Fast lookup, predictable iteration order	Sorted unique elements