LinkedList in Java

Last Updated: 06-07-2020

Linked List is a part of the [Collection framework](https://www.geeksforgeeks.org/collections-in-java-2/) present in [java.util package](https://www.geeksforgeeks.org/java-util-package-java/). This class is an implementation of the [LinkedList data structure](https://www.geeksforgeeks.org/data-structures/linked-list/) which is a linear data structure where the elements are not stored in contiguous locations and every element is a separate object with a data part and address part. The elements are linked using pointers and addresses. Each element is known as a node. Due to the dynamicity and ease of insertions and deletions, they are preferred over the arrays. It also has few disadvantages like the nodes cannot be accessed directly instead we need to start from the head and follow through the link to reach to a node we wish to access.

**Example:** The following implementation demonstrates how to create and use a linked list.

|  |
| --- |
| import java.util.\*;    public class Test {        public static void main(String args[])      {          // Creating object of the          // class linked list          LinkedList<String> ll              = new LinkedList<String>();            // Adding elements to the linked list          ll.add("A");          ll.add("B");          ll.addLast("C");          ll.addFirst("D");          ll.add(2, "E");            System.out.println(ll);            ll.remove("B");          ll.remove(3);          ll.removeFirst();          ll.removeLast();            System.out.println(ll);      }  } |

**Output:**

[D, A, E, B, C]

[A]

**Performing Various Operations on ArrayList**

Let’s see how to perform some basics operations on the LinkedList.

**1. Adding Elements:** In order to add an element to an ArrayList, we can use the [add() method](https://www.geeksforgeeks.org/java-util-linkedlist-add-method-in-java/). This method is overloaded to perform multiple operations based on different parameters. They are:

* **add(Object):** This method is used to add an element at the end of the LinkedList.
* **add(int index, Object):** This method is used to add an element at a specific index in the LinkedList.

|  |
| --- |
| // Java program to add elements  // to a LinkedList    import java.util.\*;    public class GFG {        public static void main(String args[])      {          LinkedList<String> ll = new LinkedList<>();            ll.add("Geeks");          ll.add("Geeks");          ll.add(1, "For");            System.out.println(ll);      }  } |

**Output:**

[Geeks, For, Geeks]

**2. Changing Elements:** After adding the elements, if we wish to change the element, it can be done using the [set() method](https://www.geeksforgeeks.org/linkedlist-set-method-in-java/). Since a LinkedList is indexed, the element which we wish to change is referenced by the index of the element. Therefore, this method takes an index and the updated element which needs to be inserted at that index.

filter\_none

edit

play\_arrow

brightness\_4

|  |
| --- |
| // Java program to change elements  // in a LinkedList    import java.util.\*;    public class GFG {        public static void main(String args[])      {          LinkedList<String> ll = new LinkedList<>();            ll.add("Geeks");          ll.add("Geeks");          ll.add(1, "Geeks");            System.out.println("Initial LinkedList " + ll);            ll.set(1, "For");            System.out.println("Updated LinkedList " + ll);      }  } |

**Output:**

Initial LinkedList [Geeks, Geeks, Geeks]

Updated LinkedList [Geeks, For, Geeks]

**3. Removing Elements:** In order to remove an element from a LinkedList, we can use the [remove() method](https://www.geeksforgeeks.org/linkedlist-remove-method-in-java/). This method is overloaded to perform multiple operations based on different parameters. They are:

* **remove(Object):** This method is used to simply remove an object from the LinkedList. If there are multiple such objects, then the first occurrence of the object is removed.
* **remove(int index):** Since a LinkedList is indexed, this method takes an integer value which simply removes the element present at that specific index in the LinkedList. After removing the element, all the elements are moved to the left to fill the space and the indices of the objects are updated.

filter\_none

edit

play\_arrow

brightness\_4

|  |
| --- |
| // Java program to remove elements  // in a LinkedList    import java.util.\*;    public class GFG {        public static void main(String args[])      {          LinkedList<String> ll = new LinkedList<>();            ll.add("Geeks");          ll.add("Geeks");          ll.add(1, "For");            System.out.println(              "Initial LinkedList " + ll);            ll.remove(1);            System.out.println(              "After the Index Removal " + ll);            ll.remove("Geeks");            System.out.println(              "After the Object Removal " + ll);      }  } |

**Output:**

Initial LinkedList [Geeks, For, Geeks]

After the Index Removal [Geeks, Geeks]

After the Object Removal [Geeks]

**4. Iterating the LinkedList:** There are multiple ways to iterate through the LinkedList. The most famous ways are by using the basic for loop in combination with a [get() method](https://www.geeksforgeeks.org/linkedlist-get-method-in-java/) to get the element at a specific index and the advanced for loop.

filter\_none

edit

play\_arrow

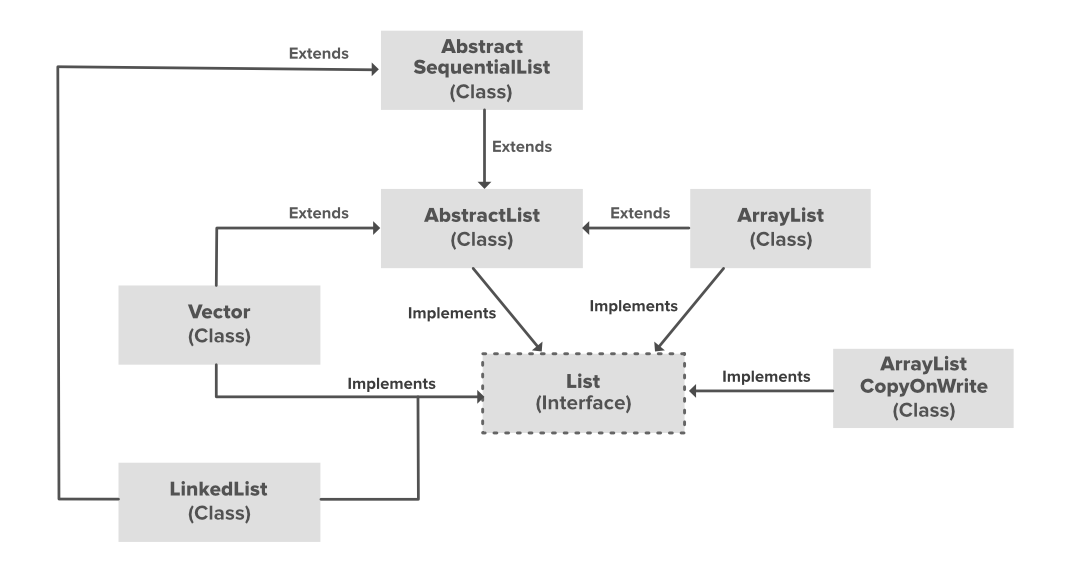
brightness\_4

|  |
| --- |
| // Java program to iterate the elements  // in an LinkedList    import java.util.\*;    public class GFG {        public static void main(String args[])      {          LinkedList<String> ll              = new LinkedList<>();            ll.add("Geeks");          ll.add("Geeks");          ll.add(1, "For");            // Using the Get method and the          // for loop          for (int i = 0; i < ll.size(); i++) {                System.out.print(ll.get(i) + " ");          }            System.out.println();            // Using the for each loop          for (String str : ll)              System.out.print(str + " ");      }  } |

**Output:**

Geeks For Geeks

Geeks For Geeks

[](https://media.geeksforgeeks.org/wp-content/uploads/20200624224531/List-ArrayList-in-Java-In-Depth-Study.png)

In the above illustration, [AbstractList](https://www.geeksforgeeks.org/abstractlist-in-java-with-examples/), [CopyOnWriteArrayList](https://www.geeksforgeeks.org/copyonwritearraylist-in-java/) and the [AbstractSequentialList](https://www.geeksforgeeks.org/abstractsequentiallist-in-java-with-examples/) are the classes which implement the list interface. A separate functionality is implemented in each of the mentioned classes. They are:

1. **AbstractList:** This class is used to implement an unmodifiable list, for which one needs to only extend this AbstractList Class and implement only the get() and the size() methods.
2. **CopyOnWriteArrayList:** This class implements the list interface. It is an enhanced version of ArrayList in which all the modifications(add, set, remove, etc.) are implemented by making a fresh copy of the list.
3. **AbstractSequentialList:** This class implements the Collection interface and the AbstractCollection class. This class is used to implement an unmodifiable list, for which one needs to only extend this AbstractList Class and implement only the get() and the size() methods.

**How LinkedList work Internally?**

Since a LinkedList acts as a dynamic array and we do not have to specify the size while creating it, the size of the list automatically increases when we dynamically add and remove items. And also, the elements are not stored in a continuous fashion. Therefore, there is no need to increase the size. Internally, the LinkedList is implemented using the [doubly linked list data structure](https://www.geeksforgeeks.org/doubly-linked-list/). The main difference between a normal linked list and a doubly LinkedList is that a doubly linked list contains an extra pointer, typically called the previous pointer, together with the next pointer and data which are there in the singly linked list.

**Constructors in the LinkedList**

In order to create a LinkedList, we need to create an object of the LinkedList class. The LinkedList class consists of various constructors that allow the possible creation of the list. The following are the constructors available in this class:

1. **LinkedList():** This constructor is used to create an empty linked list. If we wish to create an empty LinkedList with the name ll, then, it can be created as:

*LinkedList ll = new LinkedList();*

1. **LinkedList(Collection C):** This constructor is used to create an ordered list which contains all the elements of a specified collection, as returned by the collection’s iterator. If we wish to create a linkedlist with the name ll, then, it can be created as:

*LinkedList ll = new LinkedList(C);*

**Methods for Java LinkedList**

|  |  |
| --- | --- |
| **METHOD** | **DESCRIPTION** |
| [**add(int index, E element)**](https://www.geeksforgeeks.org/java-util-linkedlist-add-method-in-java/) | This method Inserts the specified element at the specified position in this list. |
| [**add(E e)**](https://www.geeksforgeeks.org/java-util-linkedlist-add-method-in-java/) | This method Appends the specified element to the end of this list. |
| [**addAll(int index, Collection<E> c)**](https://www.geeksforgeeks.org/java-util-linkedlist-addall-method-in-java/) | This method Inserts all of the elements in the specified collection into this list, starting at the specified position. |
| [**addAll(Collection<E> c)**](https://www.geeksforgeeks.org/java-util-linkedlist-addall-method-in-java/) | This method Appends all of the elements in the specified collection to the end of this list, in the order that they are returned by the specified collection’s iterator. |
| [**addFirst(E e)**](https://www.geeksforgeeks.org/linkedlist-addfirst-method-in-java/) | This method Inserts the specified element at the beginning of this list. |
| [**addLast(E e)**](https://www.geeksforgeeks.org/linkedlist-addlast-method-in-java/) | This method Appends the specified element to the end of this list. |
| [**clear()**](https://www.geeksforgeeks.org/linkedlist-clear-method-in-java/) | This method removes all of the elements from this list. |
| [**clone()**](https://www.geeksforgeeks.org/linkedlist-clone-method-in-java/) | This method returns a shallow copy of this LinkedList. |
| [**contains(Object o)**](https://www.geeksforgeeks.org/linkedlist-contains-method-in-java/) | This method returns true if this list contains the specified element. |
| [**descendingIterator()**](https://www.geeksforgeeks.org/linkedlist-descendingiterator-method-in-java-with-examples/) | This method returns an iterator over the elements in this deque in reverse sequential order. |
| [**element()**](https://www.geeksforgeeks.org/linkedlist-element-method-in-java-with-%20examples/) | This method retrieves, but does not remove, the head (first element) of this list. |
| [**get(int index)**](https://www.geeksforgeeks.org/linkedlist-get-method-in-java/) | This method returns the element at the specified position in this list. |
| [**getFirst()**](https://www.geeksforgeeks.org/java-util-linkedlist-get-getfirst-getlast-java/) | This method returns the first element in this list. |
| [**getLast()**](https://www.geeksforgeeks.org/linkedlist-getlast-method-in-java/) | This method returns the last element in this list. |
| [**indexOf(Object o)**](https://www.geeksforgeeks.org/linkedlist-indexof-method-in-java/) | This method returns the index of the first occurrence of the specified element in this list, or -1 if this list does not contain the element. |
| [**lastIndexOf(Object o)**](https://www.geeksforgeeks.org/linkedlist-lastindexof-method-in-java/) | This method returns the index of the last occurrence of the specified element in this list, or -1 if this list does not contain the element. |
| [**listIterator(int index)**](https://www.geeksforgeeks.org/linkedlist-listiterator-method-in-java/) | This method returns a list-iterator of the elements in this list (in proper sequence), starting at the specified position in the list. |
| [**offer(E e)**](https://www.geeksforgeeks.org/java-util-linkedlist-offer-offerfirst-offerlast-java/) | This method Adds the specified element as the tail (last element) of this list. |
| [**offerFirst(E e)**](https://www.geeksforgeeks.org/java-util-linkedlist-offer-offerfirst-offerlast-java/) | This method Inserts the specified element at the front of this list. |
| [**offerLast(E e)**](https://www.geeksforgeeks.org/java-util-linkedlist-offer-offerfirst-offerlast-java/) | This method Inserts the specified element at the end of this list. |
| [**peek()**](https://www.geeksforgeeks.org/java-util-linkedlist-peek-peekfirst-peeklast-java/) | This method retrieves, but does not remove, the head (first element) of this list. |
| [**peekFirst()**](https://www.geeksforgeeks.org/java-util-linkedlist-peek-peekfirst-peeklast-java/) | This method retrieves, but does not remove, the first element of this list, or returns null if this list is empty. |
| [**peekLast()**](https://www.geeksforgeeks.org/java-util-linkedlist-peek-peekfirst-peeklast-java/) | This method retrieves, but does not remove, the last element of this list, or returns null if this list is empty. |
| [**poll()**](https://www.geeksforgeeks.org/java-util-linkedlist-poll-pollfirst-polllast-%20examples-java/) | This method retrieves and removes the head (first element) of this list. |
| [**pollFirst()**](https://www.geeksforgeeks.org/java-util-linkedlist-poll-pollfirst-polllast-%20examples-java/) | This method retrieves and removes the first element of this list, or returns null if this list is empty. |
| [**pollLast()**](https://www.geeksforgeeks.org/java-util-linkedlist-poll-pollfirst-polllast-%20examples-java/) | This method retrieves and removes the last element of this list, or returns null if this list is empty. |
| [**pop()**](https://www.geeksforgeeks.org/linkedlist-pop-method-in-java/) | This method Pops an element from the stack represented by this list. |
| [**push(E e)**](https://www.geeksforgeeks.org/linkedlist-push-method-in-java/) | This method Pushes an element onto the stack represented by this list. |
| [**remove()**](https://www.geeksforgeeks.org/linkedlist-remove-method-in-java/) | This method retrieves and removes the head (first element) of this list. |
| [**remove(int index)**](https://www.geeksforgeeks.org/linkedlist-remove-method-in-java/) | This method removes the element at the specified position in this list. |
| [**remove(Object o)**](https://www.geeksforgeeks.org/linkedlist-remove-method-in-java/) | This method removes the first occurrence of the specified element from this list, if it is present. |
| [**removeFirst()**](https://www.geeksforgeeks.org/linkedlist-removefirst-method-in-java/) | This method removes and returns the first element from this list. |
| [**removeFirstOccurrence(Object o)**](https://www.geeksforgeeks.org/linkedlist-removefirstoccurrence-method-in-%20java/) | This method removes the first occurrence of the specified element in this list (when traversing the list from head to tail). |
| [**removeLast()**](https://www.geeksforgeeks.org/linkedlist-removelast-method-in-java/) | This method removes and returns the last element from this list. |
| [**removeLastOccurrence(Object o)**](https://www.geeksforgeeks.org/linkedlist-removelastoccurrence-method-in-java-with-example/) | This method removes the last occurrence of the specified element in this list (when traversing the list from head to tail). |
| [**set(int index, E element)**](https://www.geeksforgeeks.org/linkedlist-set-method-in-java/) | This method replaces the element at the specified position in this list with the specified element. |
| [**size()**](https://www.geeksforgeeks.org/linkedlist-size-method-in-java/) | This method returns the number of elements in this list. |
| [**spliterator()**](https://www.geeksforgeeks.org/linkedlist-spliterator-method-in-java/) | This method Creates a late-binding and fail-fast Spliterator over the elements in this list. |
| [**toArray()**](https://www.geeksforgeeks.org/linkedlist-toarray-method-in-java-with-example/) | This method returns an array containing all of the elements in this list in proper sequence (from first to last element). |
| [**toArray(T[] a)**](https://www.geeksforgeeks.org/linkedlist-toarray-method-in-java-with-example/) | This method returns an array containing all of the elements in this list in proper sequence (from first to last element); the runtime type of the returned array is that of the specified array. |
| **toString()** | This method returns a String containing all of the elements in this list in proper sequence (from first to last element), each element is separated by commas and the String is enclosed in square brackets. |

**Topics :**

* [Singly Linked List](https://www.geeksforgeeks.org/data-structures/linked-list/#singlyLinkedList)
* [Circular Linked List](https://www.geeksforgeeks.org/data-structures/linked-list/#circularLinkedList)
* [Doubly Linked List](https://www.geeksforgeeks.org/data-structures/linked-list/#doublyLinkedList)
* [Misc](https://www.geeksforgeeks.org/data-structures/linked-list/#misc)
* [Quick Links](https://www.geeksforgeeks.org/data-structures/linked-list/#quick)

**Singly Linked List :**

1. [Introduction to Linked List](http://geeksquiz.com/linked-list-set-1-introduction/)
2. [Linked List vs Array](https://www.geeksforgeeks.org/linked-list-vs-array/)
3. [Linked List Insertion](http://geeksquiz.com/linked-list-set-2-inserting-a-node/)
4. [Linked List Deletion (Deleting a given key)](http://geeksquiz.com/linked-list-set-3-deleting-node/)
5. [Linked List Deletion (Deleting a key at given position)](http://geeksquiz.com/delete-a-linked-list-node-at-a-given-position/)
6. [Write a function to delete a Linked List](https://www.geeksforgeeks.org/write-a-function-to-delete-a-linked-list/)
7. [Find Length of a Linked List (Iterative and Recursive)](http://geeksquiz.com/find-length-of-a-linked-list-iterative-and-recursive/)
8. [Search an element in a Linked List (Iterative and Recursive)](http://geeksquiz.com/search-an-element-in-a-linked-list-iterative-and-recursive/)
9. [Write a function to get Nth node in a Linked List](https://www.geeksforgeeks.org/write-a-function-to-get-nth-node-in-a-linked-list/)
10. [Nth node from the end of a Linked List](https://www.geeksforgeeks.org/nth-node-from-the-end-of-a-linked-list/)
11. [Print the middle of a given linked list](https://www.geeksforgeeks.org/write-a-c-function-to-print-the-middle-of-the-linked-list/)
12. [Write a function that counts the number of times a given int occurs in a Linked List](https://www.geeksforgeeks.org/write-a-function-that-counts-the-number-of-times-a-given-int-occurs-in-a-linked-list/)
13. [Detect loop in a linked list](https://www.geeksforgeeks.org/write-a-c-function-to-detect-loop-in-a-linked-list/)
14. [Find length of loop in linked list](https://www.geeksforgeeks.org/find-length-of-loop-in-linked-list/)
15. [Function to check if a singly linked list is palindrome](https://www.geeksforgeeks.org/function-to-check-if-a-singly-linked-list-is-palindrome/)
16. [Remove duplicates from a sorted linked list](https://www.geeksforgeeks.org/remove-duplicates-from-a-sorted-linked-list/)
17. [Remove duplicates from an unsorted linked list](https://www.geeksforgeeks.org/remove-duplicates-from-an-unsorted-linked-list/)
18. [Swap nodes in a linked list without swapping data](https://www.geeksforgeeks.org/swap-nodes-in-a-linked-list-without-swapping-data/)
19. [Pairwise swap elements of a given linked list](https://www.geeksforgeeks.org/pairwise-swap-elements-of-a-given-linked-list/)
20. [Move last element to front of a given Linked List](https://www.geeksforgeeks.org/move-last-element-to-front-of-a-given-linked-list/)
21. [Intersection of two Sorted Linked Lists](https://www.geeksforgeeks.org/intersection-of-two-sorted-linked-lists/)
22. [Intersection point of two Linked Lists.](https://www.geeksforgeeks.org/write-a-function-to-get-the-intersection-point-of-two-linked-lists/)
23. [QuickSort on Singly Linked List](https://www.geeksforgeeks.org/quicksort-on-singly-linked-list/)
24. [Segregate even and odd nodes in a Linked List](https://www.geeksforgeeks.org/segregate-even-and-odd-elements-in-a-linked-list/)
25. [Reverse a linked list](https://www.geeksforgeeks.org/write-a-function-to-reverse-the-nodes-of-a-linked-list/)

[More >>](https://www.geeksforgeeks.org/data-structures/linked-list/singly-linked-list/)

**Circular Linked List :**

1. [Circular Linked List Introduction and Applications,](http://geeksquiz.com/circular-linked-list/)
2. [Circular Linked List Traversal](http://geeksquiz.com/circular-linked-list-set-2-traversal/)
3. [Split a Circular Linked List into two halves](https://www.geeksforgeeks.org/split-a-circular-linked-list-into-two-halves/)
4. [Sorted insert for circular linked list](https://www.geeksforgeeks.org/sorted-insert-for-circular-linked-list/)
5. [Check if a linked list is Circular Linked List](https://www.geeksforgeeks.org/check-if-a-linked-list-is-circular-linked-list/)
6. [Convert a Binary Tree to a Circular Doubly Link List](https://www.geeksforgeeks.org/convert-a-binary-tree-to-a-circular-doubly-link-list/)
7. [Circular Singly Linked List | Insertion](https://www.geeksforgeeks.org/circular-singly-linked-list-insertion/)
8. [Deletion from a Circular Linked List](https://www.geeksforgeeks.org/deletion-circular-linked-list/)
9. [Circular Queue | Set 2 (Circular Linked List Implementation)](https://www.geeksforgeeks.org/circular-queue-set-2-circular-linked-list-implementation/)
10. [Count nodes in Circular linked list](https://www.geeksforgeeks.org/count-nodes-circular-linked-list/)
11. [Josephus Circle using circular linked list](https://www.geeksforgeeks.org/josephus-circle-using-circular-linked-list/)
12. [Convert singly linked list into circular linked list](https://www.geeksforgeeks.org/convert-singly-linked-list-circular-linked-list/)
13. [Circular Linked List | Set 1 (Introduction and Applications)](https://www.geeksforgeeks.org/circular-linked-list/)
14. [Circular Linked List | Set 2 (Traversal)](https://www.geeksforgeeks.org/circular-linked-list-set-2-traversal/)
15. [Implementation of Deque using circular array](https://www.geeksforgeeks.org/implementation-deque-using-circular-array/)
16. [Exchange first and last nodes in Circular Linked List](https://www.geeksforgeeks.org/exchange-first-last-node-circular-linked-list/)

[More >>](https://www.geeksforgeeks.org/tag/circular-linked-list/)

**Doubly Linked List :**

1. [Doubly Linked List Introduction and Insertion](http://geeksquiz.com/doubly-linked-list/)
2. [Delete a node in a Doubly Linked List](https://www.geeksforgeeks.org/delete-a-node-in-a-doubly-linked-list/)
3. [Reverse a Doubly Linked List](https://www.geeksforgeeks.org/reverse-a-doubly-linked-list/)
4. [The Great Tree-List Recursion Problem.](https://www.geeksforgeeks.org/the-great-tree-list-recursion-problem/)
5. [Copy a linked list with next and arbit pointer](https://www.geeksforgeeks.org/a-linked-list-with-next-and-arbit-pointer/)
6. [QuickSort on Doubly Linked List](https://www.geeksforgeeks.org/quicksort-for-linked-list/)
7. [Swap Kth node from beginning with Kth node from end in a Linked List](https://www.geeksforgeeks.org/swap-kth-node-from-beginning-with-kth-node-from-end-in-a-linked-list/)
8. [Merge Sort for Doubly Linked List](https://www.geeksforgeeks.org/merge-sort-for-doubly-linked-list/)
9. [Create a Doubly Linked List from a Ternary Tree](https://www.geeksforgeeks.org/create-doubly-linked-list-ternary-ree/)
10. [Find pairs with given sum in doubly linked list](https://www.geeksforgeeks.org/find-pairs-given-sum-doubly-linked-list/)
11. [Insert value in sorted way in a sorted doubly linked list](https://www.geeksforgeeks.org/insert-value-sorted-way-sorted-doubly-linked-list/)
12. [Delete a Doubly Linked List node at a given position](https://www.geeksforgeeks.org/delete-doubly-linked-list-node-given-position/)
13. [Count triplets in a sorted doubly linked list whose sum is equal to a given value x](https://www.geeksforgeeks.org/count-triplets-sorted-doubly-linked-list-whose-sum-equal-given-value-x/)
14. [Remove duplicates from a sorted doubly linked list](https://www.geeksforgeeks.org/remove-duplicates-sorted-doubly-linked-list/)
15. [Delete all occurrences of a given key in a doubly linked list](https://www.geeksforgeeks.org/delete-occurrences-given-key-doubly-linked-list/)
16. [Remove duplicates from an unsorted doubly linked list](https://www.geeksforgeeks.org/remove-duplicates-unsorted-doubly-linked-list/)
17. [Sort the biotonic doubly linked list](https://www.geeksforgeeks.org/sort-biotonic-doubly-linked-list/)
18. [Sort a k sorted doubly linked list](https://www.geeksforgeeks.org/sort-k-sorted-doubly-linked-list/)
19. [Convert a given Binary Tree to Doubly Linked List | Set](https://www.geeksforgeeks.org/convert-a-given-binary-tree-to-doubly-linked-list-set-4/)
20. [Program to find size of Doubly Linked List](https://www.geeksforgeeks.org/program-find-size-doubly-linked-list/)
21. [Sorted insert in a doubly linked list with head and tail pointers](https://www.geeksforgeeks.org/create-doubly-linked-list-using-double-pointer-inserting-nodes-list-remains-ascending-order/)
22. [Large number arithmetic using doubly linked list](https://www.geeksforgeeks.org/large-number-arithmetic-using-doubly-linked-list/)
23. [Rotate Doubly linked list by N nodes](https://www.geeksforgeeks.org/rotate-doubly-linked-list-n-nodes/)
24. [Priority Queue using doubly linked list](https://www.geeksforgeeks.org/priority-queue-using-doubly-linked-list/)
25. [Reverse a doubly linked list in groups of given size](https://www.geeksforgeeks.org/reverse-doubly-linked-list-groups-given-size/)
26. [Doubly Circular Linked List | Set 1 (Introduction and Insertion)](https://www.geeksforgeeks.org/doubly-circular-linked-list-set-1-introduction-and-insertion/)
27. [Doubly Circular Linked List | Set 2 (Deletion)](https://www.geeksforgeeks.org/doubly-circular-linked-list-set-2-deletion/)

[More >>](https://www.geeksforgeeks.org/data-structures/linked-list/doubly-linked-list/)

**Misc :**

1. [Skip List | Set 1 (Introduction)](https://www.geeksforgeeks.org/skip-list/)
2. [Skip List | Set 2 (Insertion)](https://www.geeksforgeeks.org/skip-list-set-2-insertion/)
3. [Skip List | Set 3 (Searching and Deletion)](https://www.geeksforgeeks.org/skip-list-set-3-searching-deletion/)
4. [Reverse a stack without using extra space in O(n)](https://www.geeksforgeeks.org/reverse-stack-without-using-extra-space/)
5. [An interesting method to print reverse of a linked list](https://www.geeksforgeeks.org/an-interesting-method-to-print-reverse-of-a-linked-list/)
6. [Linked List representation of Disjoint Set Data Structures](https://www.geeksforgeeks.org/linked-list-representation-disjoint-set-data-structures/)
7. [Sublist Search (Search a linked list in another list)](https://www.geeksforgeeks.org/sublist-search-search-a-linked-list-in-another-list/)
8. [How to insert elements in C++ STL List ?](https://www.geeksforgeeks.org/insert-elements-c-stl-list/)
9. [Unrolled Linked List | Set 1 (Introduction)](https://www.geeksforgeeks.org/unrolled-linked-list-set-1-introduction/)
10. [A Programmer’s approach of looking at Array vs. Linked List](http://geeksquiz.com/programmers-approach-looking-array-vs-linked-list/)
11. [How to write C functions that modify head pointer of a Linked List?](https://www.geeksforgeeks.org/how-to-write-functions-that-modify-the-head-pointer-of-a-linked-list/)
12. [Given a linked list which is sorted, how will you insert in sorted way](https://www.geeksforgeeks.org/given-a-linked-list-which-is-sorted-how-will-you-insert-in-sorted-way/)
13. [Can we reverse a linked list in less than O(n)?](http://geeksquiz.com/can-we-reverse-a-linked-list-in-less-than-on/)
14. [Practice questions for Linked List and Recursion](https://www.geeksforgeeks.org/practice-questions-for-linked-list-and-recursion/)
15. [Construct a Maximum Sum Linked List out of two Sorted Linked Lists having some Common nodes](https://www.geeksforgeeks.org/maximum-sum-linked-list-two-sorted-linked-lists-common-nodes/)
16. [Given only a pointer to a node to be deleted in a singly linked list, how do you delete it?](https://www.geeksforgeeks.org/given-only-a-pointer-to-a-node-to-be-deleted-in-a-singly-linked-list-how-do-you-delete-it/)
17. [Why Quick Sort preferred for Arrays and Merge Sort for Linked Lists?](https://www.geeksforgeeks.org/why-quick-sort-preferred-for-arrays-and-merge-sort-for-linked-lists/)
18. [Squareroot(n)-th node in a Linked List](https://www.geeksforgeeks.org/squarerootnth-node-in-a-linked-list/)
19. [Find the fractional (or n/k – th) node in linked list](https://www.geeksforgeeks.org/find-fractional-nk-th-node-linked-list/)
20. [Find modular node in a linked list](https://www.geeksforgeeks.org/find-modular-node-linked-list/)
21. [Construct a linked list from 2D matrix](https://www.geeksforgeeks.org/construct-linked-list-2d-matrix/)
22. [Find smallest and largest elements in singly linked list](https://www.geeksforgeeks.org/find-smallest-largest-elements-singly-linked-list/)
23. [Arrange consonants and vowels nodes in a linked list](https://www.geeksforgeeks.org/arrange-consonants-vowels-nodes-linked-list/)
24. [Partitioning a linked list around a given value and If we don’t care about making the elements of the list “stable”](https://www.geeksforgeeks.org/partitioning-linked-list-around-given-value-dont-care-making-elements-list-stable/)
25. [Modify contents of Linked List](https://www.geeksforgeeks.org/modify-contents-linked-list/)

**Quick Links :**

* [‘Practice Problems’ on Linked List](https://practice.geeksforgeeks.org/topics/Linked-List/)
* [‘Videos’ on Linked List](https://www.youtube.com/playlist?list=PLqM7alHXFySH41ZxzrPNj2pAYPOI8ITe7)
* [‘Quizzes’ on Linked List](https://www.geeksforgeeks.org/data-structure-gq/linked-list-gq/)

Top 20 Linked List Interview Question

1. [Print the Middle of a given linked list](https://www.geeksforgeeks.org/write-a-c-function-to-print-the-middle-of-the-linked-list/)
2. [Flattening a linked list](https://www.geeksforgeeks.org/flattening-a-linked-list/)
3. [Delete the elements in an linked list whose sum is equal to zero](https://www.careercup.com/question?id=5717797377146880)
4. [Delete middle of linked list](https://practice.geeksforgeeks.org/problem-page.php?pid=700175)
5. [Remove duplicate elements from sorted linked list](https://practice.geeksforgeeks.org/problem-page.php?pid=700196)
6. [Add 1 to a number represented as a linked list](https://www.geeksforgeeks.org/add-1-number-represented-linked-list/)
7. [Reverse a linked list in groups of given size](https://practice.geeksforgeeks.org/problem-page.php?pid=700013)
8. [Detect loop in linked list](https://practice.geeksforgeeks.org/problem-page.php?pid=700099)
9. [Remove loop in linked list](https://practice.geeksforgeeks.org/problems/remove-loop-in-linked-list/1)
10. [Find nth node from the end of linked list](https://practice.geeksforgeeks.org/problems/nth-node-from-end-of-linked-list/1)
11. [Function to check if a singly linked list is a palindrome](https://www.geeksforgeeks.org/function-to-check-if-a-singly-linked-list-is-palindrome/)
12. [Reverse alternate k node in a singly linked list](https://www.geeksforgeeks.org/reverse-alternate-k-nodes-in-a-singly-linked-list/)
13. [Delete last occurrence of an item from linked list](http://quiz.geeksforgeeks.org/delete-last-occurrence-of-an-item-from-linked-list/)
14. [Rotate a linked list.](https://www.geeksforgeeks.org/rotate-a-linked-list/)
15. [Delete n nodes after m nodes of a linked list.](https://www.geeksforgeeks.org/delete-n-nodes-after-m-nodes-of-a-linked-list/)
16. [Merge a linked list into another linked list at alternate positions.](https://www.geeksforgeeks.org/merge-a-linked-list-into-another-linked-list-at-alternate-positions/)
17. [Write a function to delete a linked list.](https://www.geeksforgeeks.org/write-a-function-to-delete-a-linked-list/)
18. [Write a function to reverse the nodes of a linked list.](https://www.geeksforgeeks.org/write-a-function-to-reverse-the-nodes-of-a-linked-list/)
19. [Why quicksort is preferred for arrays and merge sort for linked lists.](https://www.geeksforgeeks.org/why-quick-sort-preferred-for-arrays-and-merge-sort-for-linked-lists/)
20. [linked list in java](http://quiz.geeksforgeeks.org/linked-list-in-java/)