For all remember,the following four basic operations.
Create
Read
Update
Delete

Linked List -

```
template <class T>
class LinkedList {
      public :
            LinkedList * next;
            T data;
};
```

Time complexity -

1) Insert in front – O(1)
2) Insert at Back – O(n)
3) Insert in middle – O(n)
4) Delete Front – O(1)
5) Delete back – O(n)
6) Delete Middle – O(n)
7) Search – O(n)
8) Update – O(n)
9) Push – O(1)
10) Pop – O(1)

Common Operations -

0) create_node()
1) add_node() // Add a node LL
2) add_front()
3) push()
4) pop()
5) enqueue()
6) dequeue()
7) push_back()
8) pop_back()
9) insert_at()
10) walk_list()
11) delete_list()
12) sorted_insert()
13) sort()
14) remove_duplicates()

Use Linked List when -
1) You do not know the size of the list you will have
[ADD MORE]

Common Questions in Linked List -
1) Reverse a Linked List
2) Remove Duplicates in Linked List
3) Recursively travese a linked list
4) Recursively reverse a linked list
5) Recurively remove duplicates in Linked List
6) Merge two Linked List – Alternately
7) Merge sort two Linked List
8) Find a loop in Linked List
9) Find the start of the loop in the Linked List
10) Find the middle element in the linked list
11) Find the kth-element from back in the linked list
12) Divide the linked list into two lists (split_list)
13) Number in reverse order (e.g. 123 as 3->2->1)

Stacks and Queue

Common Questions -

1) Postfix expression
2) Brackets
3) DFS (Stack)
4) BFS (Queue)
5) Design stack using queue
6) Design queue using stack
7) Have O(1) operations for min(), push() and pop()
8) Stack using LL
9) Queue using LL
10) Multiple (3+) stack in one array
11) Multiple (3+) queue in one array
12) Infix to Postfix
13) Given stacks "S" and "T", and a variable "v" how would you reverse the order of elements in "S"

*14) Reverse a stack without using extra space