1. **push_front(int Val)**

   Inserts a new node at the beginning of the list.

   - Creates a new node with the given value.
   - If the list is empty, the new node becomes both Head and Tail.
   - Otherwise:
     - The new node's Next pointer is set to the current Head.
     - The current Head's Prev pointer is set to the new node.
     - The Head is updated to the new node.

2. **push_back(int Val)**

   Inserts a new node at the end of the list.

   - Creates a new node with the given value.
   - If the list is empty, the new node becomes both Head and Tail.
   - Otherwise:
     - The current Tail's Next pointer is set to the new node.
     - The new node's Prev pointer is set to the current Tail.
     - The Tail is updated to the new node.

3. **insert(int Val, int Pos)**

   Inserts a new node at a specified position.

   - If the position is invalid (< 0), it prints an error.
   - If inserting at position 0, it calls push_front(Val).
   - Otherwise:
     - Traverses the list to find the node at Pos - 1.
     - Creates a new node.
     - Updates pointers:
       - The new node's Next points to the next node.
       - The new node's Prev points to the current node.

- If there is a next node, update its Prev to point to the new node.
- The current node's Next is updated to the new node.

4. insertCenter(int Val)

Inserts a new node in the middle of the list.

- If the list is empty, it calls push_front(Val).
- Uses the **slow & fast pointer technique**:
  - Slow moves one step at a time.
  - Fast moves two steps at a time.
  - When Fast reaches the end, Slow is at the middle.
- Inserts a new node **after** Slow, updating pointers accordingly.

5. displayForward()

Displays the list from Head to Tail.

- Starts from Head and traverses using Next pointers.
- Prints each node's value until NULL is reached.

6. displayReverse()

Displays the list from Tail to Head.

- Starts from Tail and traverses using Prev pointers.
- Prints each node's value until NULL is reached.

Output:

```
3->2->1->NULL
3->2->1->4->NULL
5->3->2->1->4->NULL
After inserting at center: 5->3->2->6->1->4->NULL
Display in reverse order: 4->1->6->2->3->5->NULL
```