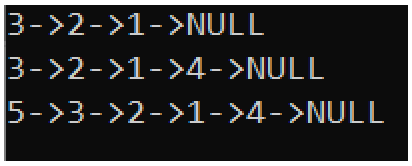
Logical Explanation of the code

1. **Checks for invalid positions** – If the given position is **negative**, it prints an error message and exits.
2. **Handles insertion at the head** – If the position is **0**, it calls insert_at_start(val) to insert the node at the beginning.
3. **Traverses the list** – It moves through the list until it reaches the **(pos - 1)th node** (the node before the insertion point).
4. **Inserts the new node** – It creates a new node, links it to the next node, and updates the previous node's next pointer to maintain the list structure.

Output:



```
3->2->1->NULL
3->2->1->4->NULL
5->3->2->1->4->NULL
```

Code:

```
//Function to insert at any position.
    void insert(int val, int pos)
{
            if(pos<0){
            cerr << "Invalid pos\n";
            return;
            }
            if(pos == 0)
            {
            push_front(val);
            return;
            }
            Node* temp = head;
            for(int i = 0; i<pos-1; i++){
            temp = temp->next;
            }
            Node* newNode = new Node(val);
            newNode->next = temp->next;
            temp->next = newNode;
}
```