WHEN WE ADD THE VALUE OF A NODE(ONLY NODE ONLY NODE.VAL) IT WILL ADD WHOLE list not just the value it will add right and left node and the process go on.  
  
public static Node insertAfter(Node head, int key, int newData) {

Node curr = head;

while (curr != null) {

if (curr.data == key)

break;

curr = curr.next;

}

if (curr == null) {

System.out.println("Node not found");

return head;

}

Node newNode = new Node(newData);

newNode.next = curr.next;//in this curr.next is a value and newNode.next is pointer or address to point

curr.next = newNode;

return head;

}

DIFF between doubly and singly linked list – in doubly it has both prev and next address so it become easy to perform operations.  
in circular linked list last node points to the first pointer and there are no null pointers. **Circular Linked List (CLL)**: Traversal doesn’t stop until it loops back to the head because the last node points back to the head, creating an infinite loop (unless manually controlled). tail’s next points to the head, and the head’s prev points to the tail.

POLYNOMIAL REPRESENTATION OF LINKED LIST: 3x^2+4x+6: means it has x only so 1 variable

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 2 | link | 4 | 1 | link | 6 | 0 | link |

3x^2+2xy^2+5y^2+7yz:it has tree variables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3 | 2 | 0 | 0 | link |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2 | 1 | 2 | 0 | link |

And so on…..

**TYPES OF LINKEDLIST:  
Singly  
Doubly  
Circular- singly, doubly.**