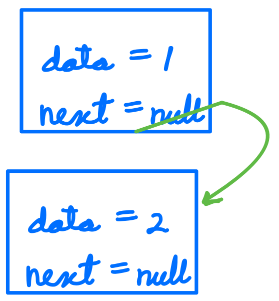
```
class Node {
    int data;
    Node next;
    Node (int x) {
        data = x;
        next = null;
    }
}
```

<u>linked list</u>

```
Node a = new Node(1);

Nod b = new Node(2);

a.next = b;
```



$$\begin{tikzcd}\boxed{1} \arrow[r] & \boxed{2} \arrow[r] & \boxed{3} \arrow[r] & \boxed{4} \arrow[r] & \boxed{5} \arrow[r] & \text{null}\end{tikzcd}$$

<u>Head</u>

Q → Create a linked list with N nodes having data 1 to N.
Return the <u>head</u> of linked list.

$N = 5$

temp

$$\begin{tikzcd}\boxed{1} \arrow[r] & \boxed{2} \arrow[r] & \boxed{3} \arrow[r] & \boxed{4} \arrow[r] & \boxed{5} \arrow[r] & \text{null}\end{tikzcd}$$

<u>Head</u>

```
Node create(int N) {
    if (N <= 0)
        return null;
    Node Head = new Node(1);
    Node temp = Head; // shallow copy
    for i → 2 to N
        Node x = new Node(i);
        temp.next = x;
        temp = x;      // shallow copy
    return Head;
}
```

$i = \cancel{2}\ 3$

$N = 3$

temp

$$\begin{tikzcd}\boxed{1} \arrow[r] & \boxed{2} \arrow[r] & \boxed{3} \arrow[r] & \text{null}\end{tikzcd}$$

<u>Head</u>

$\boxed{\text{Never to upate } \underline{\text{Head}}.}$

$N = 0$ Head = null

$TC = O(N)$

$SC = \underline{O(1)}$

Q → Given the Head of linked list, return the first node with data = X. If X is not present, return null.

$$\begin{tikzcd}\boxed{8} \arrow[r] & \boxed{3} \arrow[r] & \boxed{12} \arrow[r] & \boxed{2} \arrow[r] & \boxed{5} \arrow[r] & \text{null}\end{tikzcd}$$

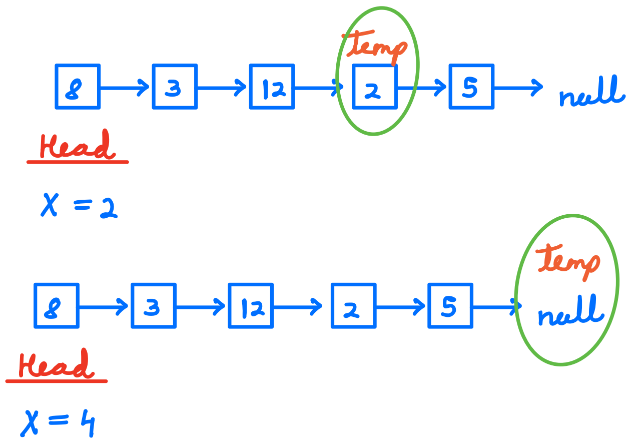
<u>Head</u> $X = 2$ ↑

$X = 4$ Ans = <u>null</u>

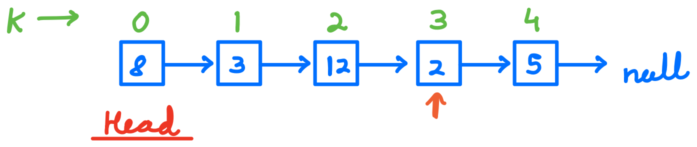
```
Node search( Node Head, int x) {
    Node temp = Head;
    while ( temp != null ) {
        if( temp.data == x)
            return temp;
        temp = temp.next;
    }
    return null;
}
```

TC = O(N)

SC = O(1)



Q → Given the Head of the linked list.
Return the data present at Kth node in linked list
s.t K = 0 → Head node & $\boxed{K >= 0}$



K = 3 Ans = 2

K = 6 Null Pointer Exception ✓

K = 0 , Head = null

```
int access (Node Head, int K) {
    Node temp = Head;
    for i → 1 to K
        if (temp == null)
            return -1;
        temp = temp.next;
    return temp != null ? temp.data : -1;
}
```

-1 ⇒ Kth position is not present.

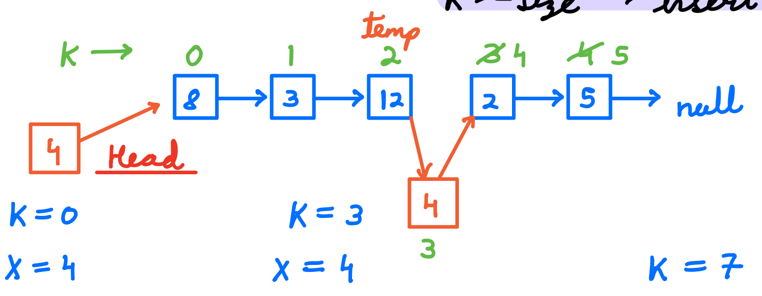
TC = O(K)

SC = O(1)

Access Kth element in array → A[K] TC = O(1)

10 : 30 PM

Q → Given a linked list, insert a node at position K with data = X. K = 0 → insert as head node.
K >= size → insert as last node. $\boxed{K>=0}$



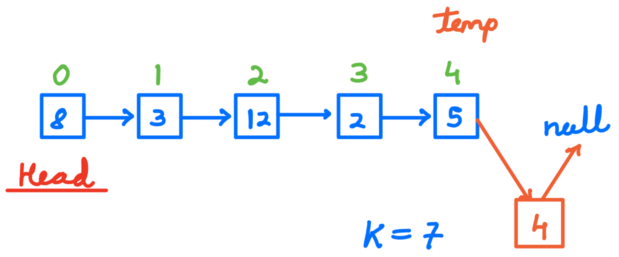
Return the head of updated linked list.

Null Pointer Exception ✓

```
Node insert(Node Head, int K, int X) {
    Node nodeX = new Node(X);
    if (K == 0 || Head == null) {
        nodeX.next = Head;
        Head = nodeX;      } return nodeX; ✓
        return Head;
    }
    Node temp = Head;
    for i → 1 to (K-1)
        if (temp.next == null)
            break;
        temp = temp.next;

    nodeX.next = temp.next;
    temp.next = nodeX;
    return Head;
}
```

K > 0, Head = null
Null Pointer Exception ✓



K = 7
X = 4

$\boxed{\begin{matrix}K=0, X=4 \\ Head = null\end{matrix}}$ ✓

TC = O(K)
SC = O(1)

Q → Delete K th node of the linked list. $\boxed{0 <= K < N}$ → H.W
Q → Count the no. of nodes in linked list. → H.W
