![Text, letter

Description automatically generated]()

Here we will start from 6 in reverse manner and check in stack is there any smaller than 6 then pop .if not like in 6 above nothing put-1 in linked list.then put 6 in stack.

![A picture containing text, whiteboard

Description automatically generated]()

Here we can see a pattern forming as stack contain element in increasing order.

Now for 3 check is there smaller element in stack .hence remove 1 and 2 from stack. Then put 4 at 3 elementwise remove 1 and 2 and it cannot be the answer as from left to right first 3 will come not 1 and 2 that come later. ![Text

Description automatically generated]()

![A picture containing text, whiteboard

Description automatically generated]()

Code:most efficient:

class Solution {

public:

ListNode\* reverse(ListNode\* head)

{

ListNode\* prevptr= NULL;

ListNode\* currptr= head;

ListNode\* nextptr;

while(currptr!= NULL)

{

nextptr= currptr->next;

currptr->next= prevptr;

prevptr= currptr;

currptr= nextptr;

}

return prevptr;

}

vector<int> nextLargerNodes(ListNode\* head)

{

vector<int> ans;

ListNode\* newHead= reverse(head);

stack<int> st;

ListNode\* temp= newHead;

while(temp!= NULL)

{

// if stack is not empty, then pop an element from stack. If the popped element is smaller than next,

// then keep popping while elements are smaller and stack is not empty.

if(!st.empty())

{

while(!st.empty() && st.top()<= temp->val)

{

st.pop();

}

}

ans.push\_back(st.empty() ? 0 : st.top());

st.push(temp->val);

temp= temp->next;

}

std::reverse(ans.begin(), ans.end());

return ans;

}

};