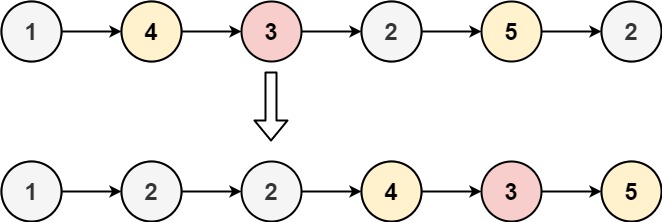
Given the head of a linked list and a value x, partition it such that all nodes **less than** x come before nodes **greater than or equal** to x.

You should **preserve** the original relative order of the nodes in each of the two partitions.

**Example 1:**



**Input:** head = [1,4,3,2,5,2], x = 3

**Output:** [1,2,2,4,3,5]

**Example 2:**

**Input:** head = [2,1], x = 2

**Output:** [1,2]

**Solution:**

/\*\*

\* Definition for singly-linked list.

\* public class ListNode {

\* int val;

\* ListNode next;

\* ListNode() {}

\* ListNode(int val) { this.val = val; }

\* ListNode(int val, ListNode next) { this.val = val; this.next = next; }

\* }

\*/

class Solution {

public ListNode partition(ListNode head, int x) {

ListNode beforeNode\_head = new ListNode(0);

ListNode before = beforeNode\_head;

ListNode afterNode\_head = new ListNode(0);

ListNode after = afterNode\_head;

while(head!=null){

if(head.val<x){

before.next = head;

before = before.next;

}else{

after.next = head;

after = after.next;

}

head = head.next;

}

before.next = afterNode\_head.next;

after.next = null;

return beforeNode\_head.next;

}

}