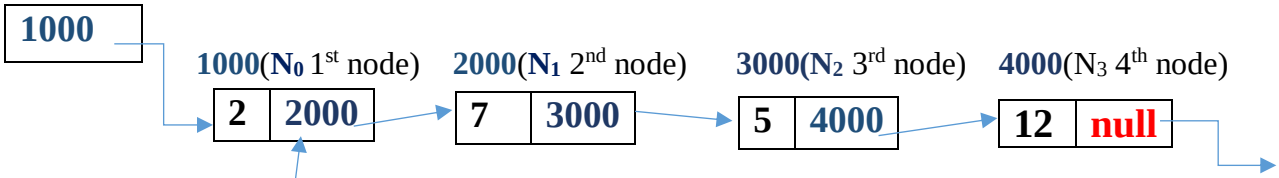
```
/*Reversing a single linked list with explanation of the code of reverse_list() method in java*/
public static void reverse_list()
{
        if(start == null) /*if list is empty don't do anything*/
       {
          System.out.println("the list is empty");
          return ;
       }
        else if ( start.link == null)/*if list conations  only one node don't do anything*/
        {
                System.out.println("the list contains one node");
                System.out.println("reverse is same as the original list");
                return ;
        }
        else
        {
/*if the list contains more than one node, then this else clause is executed to reverse the list*/
                NODE next= null;
                NODE prev=null;
                NODE curr=start;
                while(curr != null) /*this loop performs the reverse operation*/
                {
                        next= curr.link;
                        curr.link = prev;
                        prev=curr;
                         curr=next;
                }/*End of while…loop*/
                start=prev; /*updates the start pointer by final value of prev after loop*/
         }
}/*End of Reverse method */

/*Explanation of the program statements in else clause part of the above reverse_list() method*/
```

Before execution of the While…loop in the else clause

Original list before reverse operation:

start



Prev→null curr : refers to first nodes address

Next→null

Current reversed list: Its empty initially.

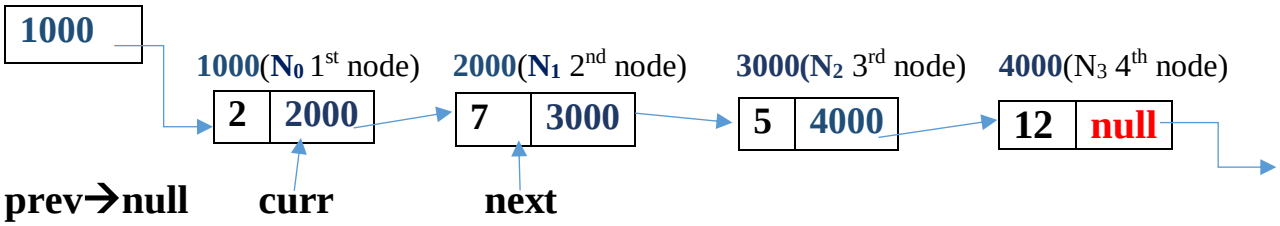
ITERATION-1 of the While…loop (curr != null) ← condition is true , curr →1st nodes addess

/*After execution of the following statement inside the while..loop:*/

next = curr.link;

Original list:

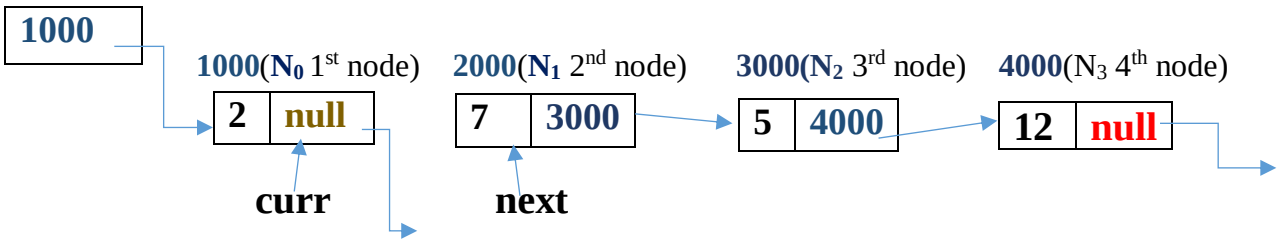
start



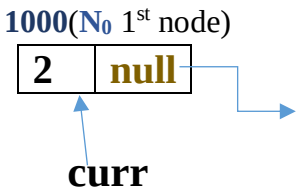
/*After execution of the following statement inside the while…loop:*/

curr.link = prev;

start



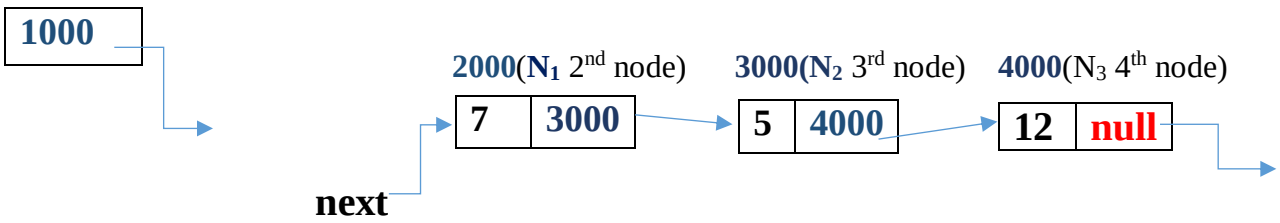
Current reversed list:



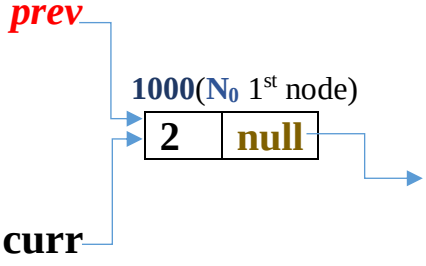
/*After execution of the following statement inside the while…loop:*/

prev = curr ;

start

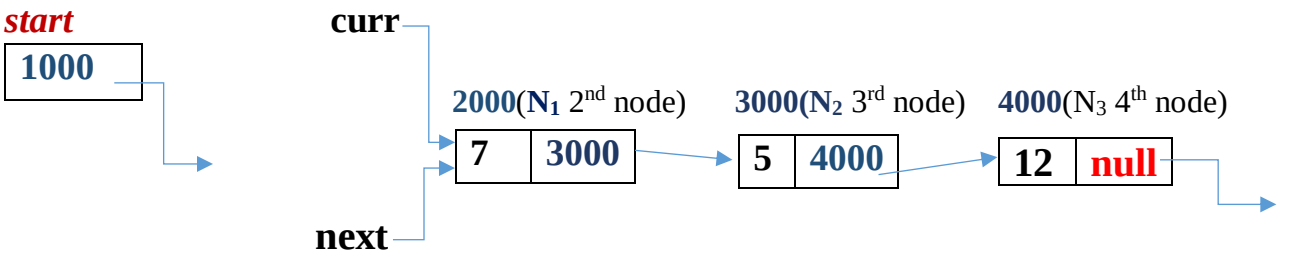


Current reversed list

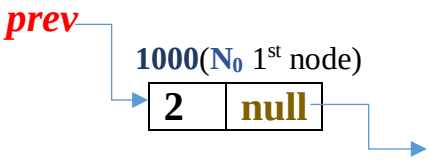


/*After execution of the following statement inside the while…loop:*/

curr = next ;



Current reversed list

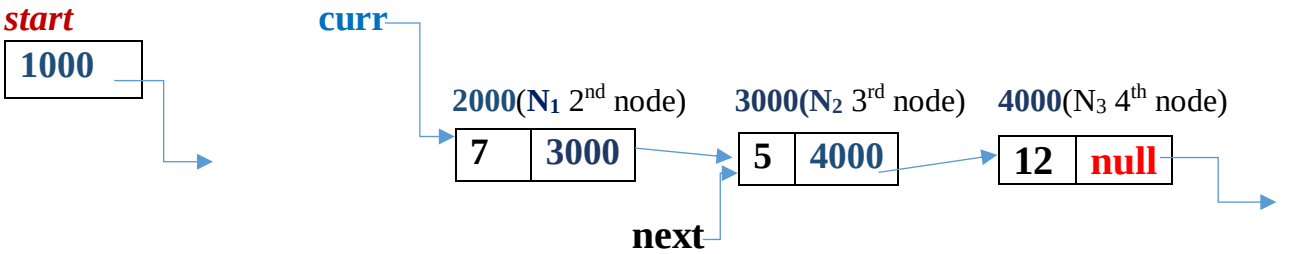


With curr→2nd node and next→2nd node the loop starts 2nd iteration

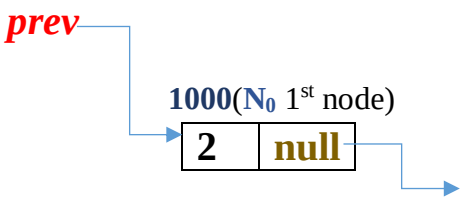
ITERATION-2 of the While…loop (curr != null) ← condition is true: curr refers to 2nd node

/*After execution of the following statement:*/

next = curr.link;

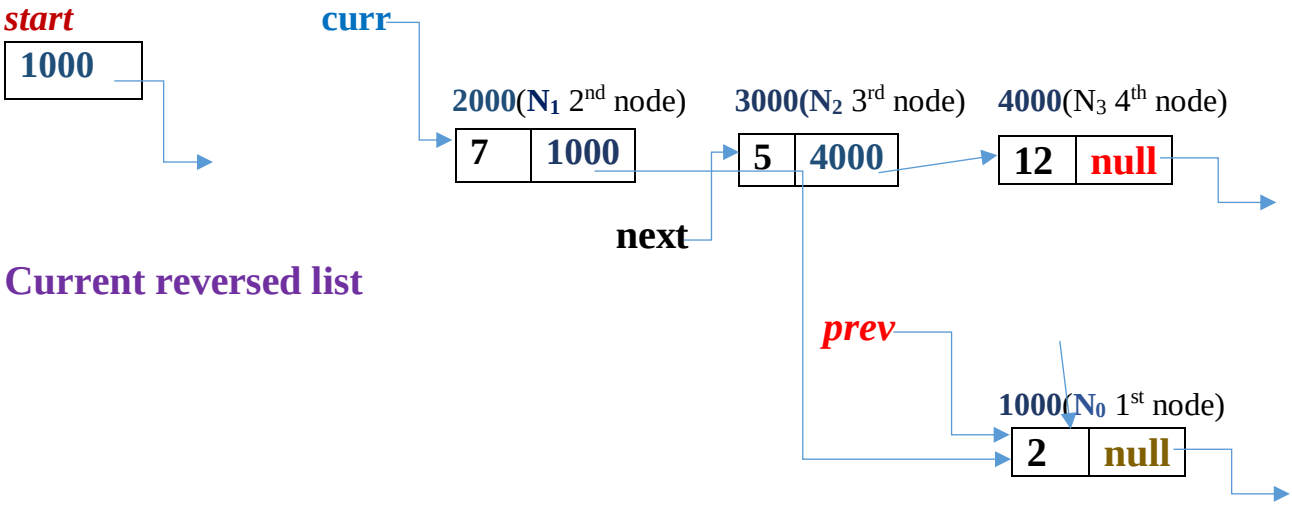


Current reversed list



/*After execution of the following statement:*/

curr.link = prev;

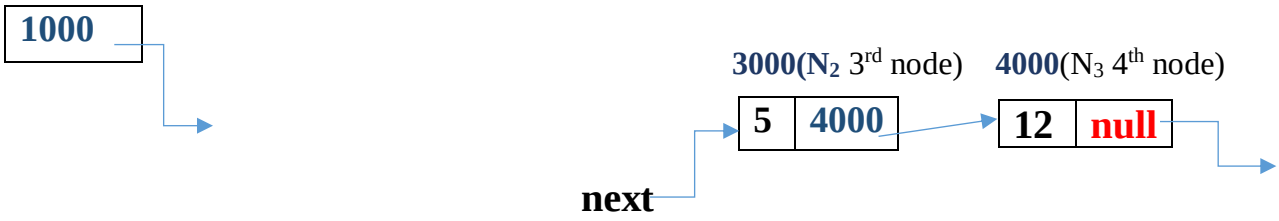


Current reversed list

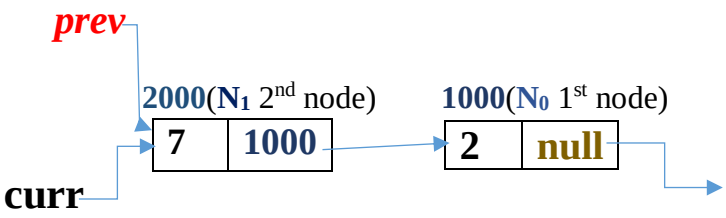
/*After execution of the following statement:*/

prev = curr ;

start



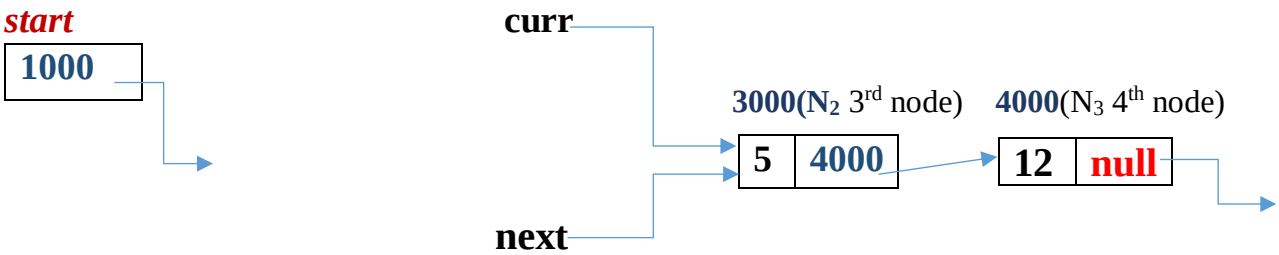
Current reversed list:



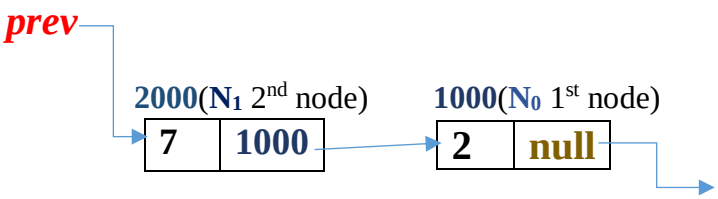
*/*After execution of the following statement:*/*

curr = next ;

start



Current reversed list:



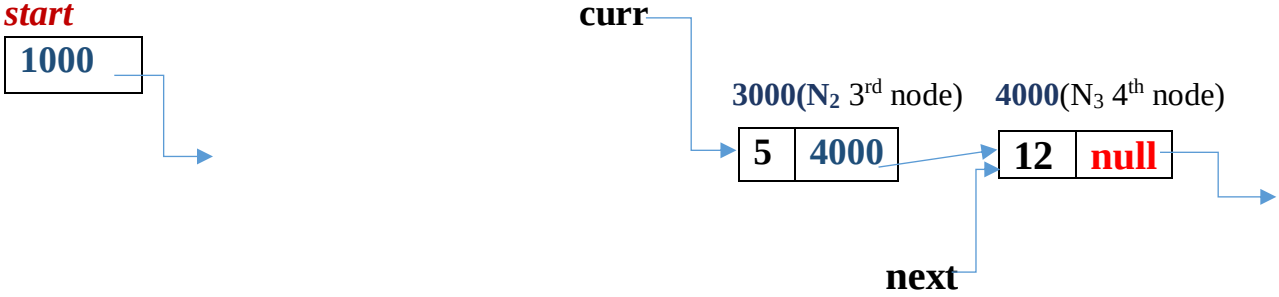
With curr→3rd node & next→3rd node 3rd iteration of while…loop starts

ITERATION-2 of the While…loop (curr != null) ← condition is true :curr refers to 3rd node

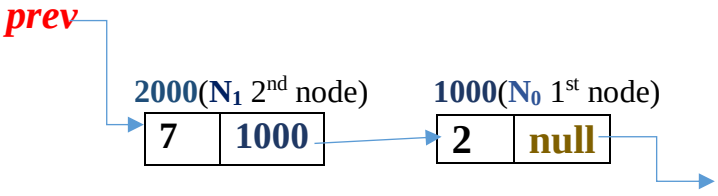
*/*After execution of the following statement:*/*

next = curr.link;

start

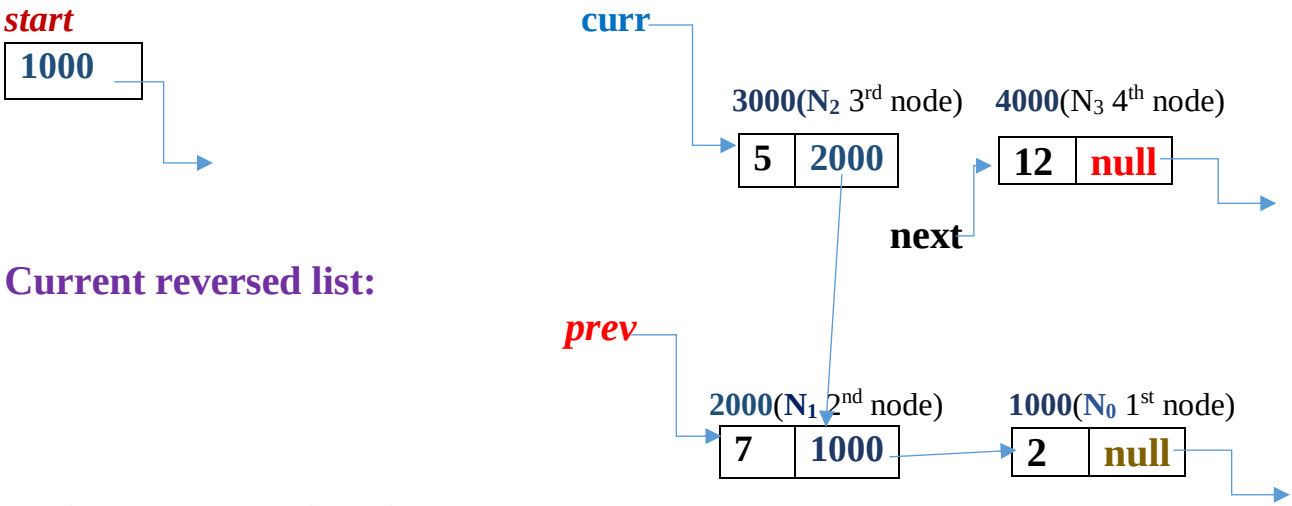


Current reversed list:



*/*After execution of the following statement:*/*

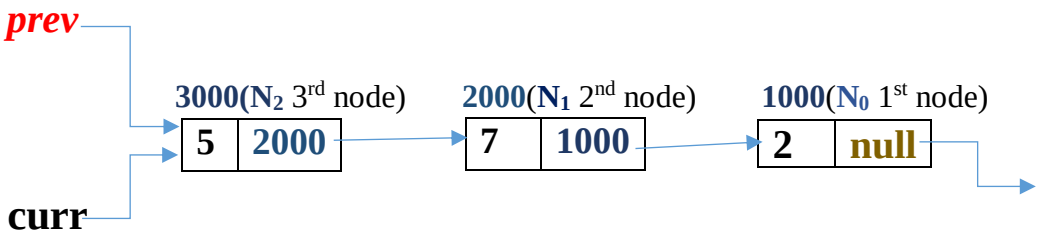
curr.link = prev;



*/*After execution of the following statement:*/*

prev = curr;

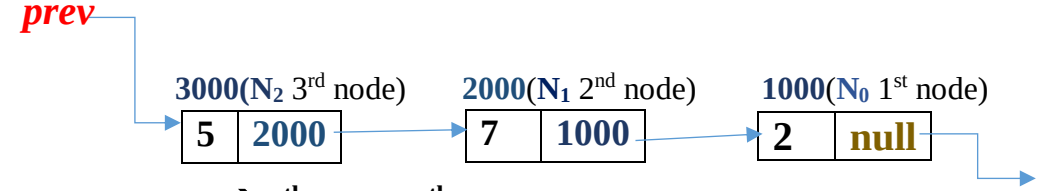
Current reversed list:



*/*After execution of the following statement:*/*

curr = next ;

Current reversed list:



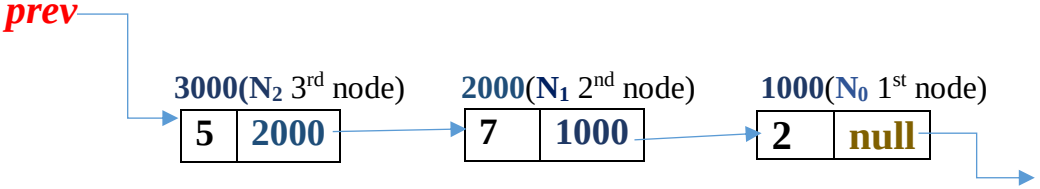
With curr→4th node & next→4th node 4th iteration of the while…loop starts

ITERATION-4of the While…loop (curr != null) ← condition is true: curr refers to 4th node

*/*After execution of the following statement:*/*

next = curr.link;

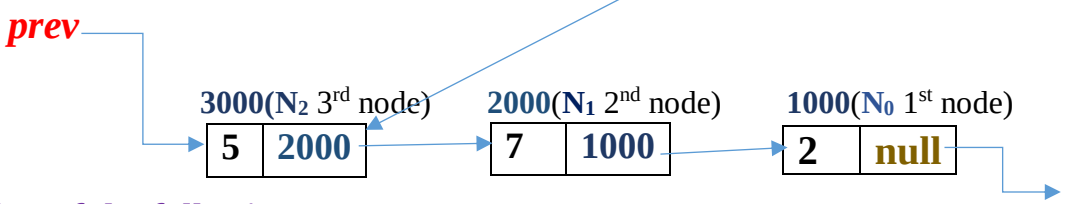
Current reversed list:



/*After execution of the following statement:*/

curr.link = prev;

Current reversed list:

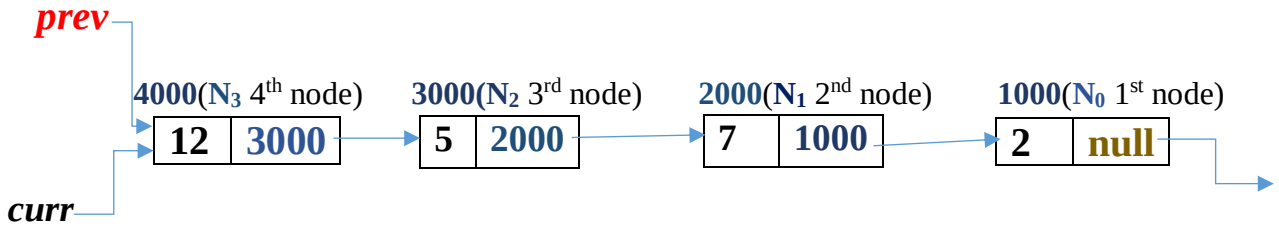


/*After execution of the following statement:*/

prev = curr ;

start **/*now the original list becomes empty*/**

Current reversed list:

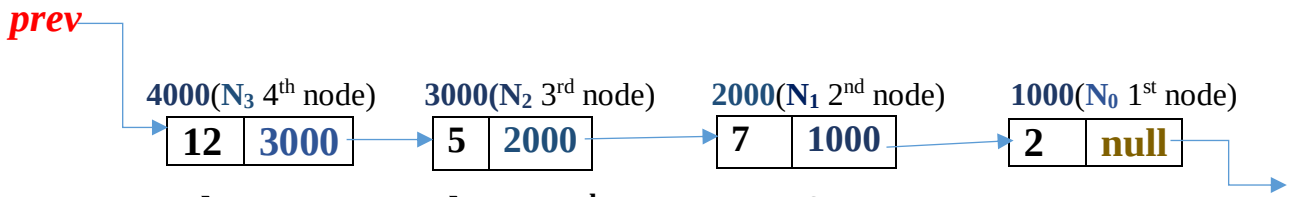


/*After execution of the following statement:*/

curr = next ;

start

Current reversed list:

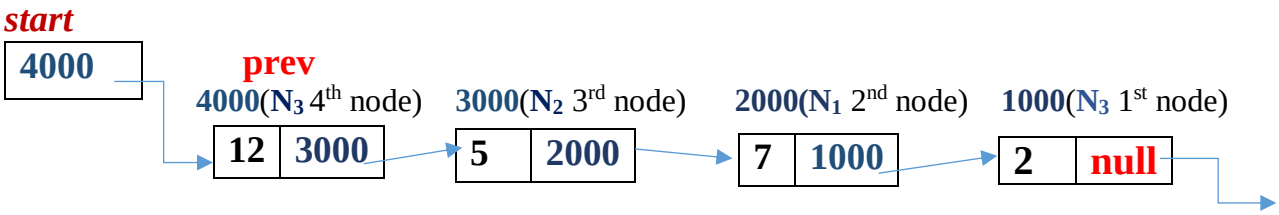


With curr→null & next→null 5th iteration of the while loop starts when all nodes of the original list get vanished

ITERATION-5 of the While…loop (curr != null) ← condition is flase: curr value is 'null'
So loop terminates.

After termination of loop prev→4th nodes address

/*After execution of last statement of the method :*/
start = prev ; /*update 'start' pointer by final value of 'prev', so 4th node becomes new first node of the reversed list as shown in the following reversed list*/



Then the method reverse_list() terminates.