**Return Kth to last**

Implement an algorithm to find the kth to last element of a singly list

//BestImplementation

public void kthToLast**(**Node root**,** int k**){**

Node p1 **=** root**;**

Node p2 **=** root**;**

**for(**int i **=**0**;** i**<**k**;**i**++){**

**if(**p1**==**null**)**

**return** null**;**

p1**=** p1**.**next**;**

**}**

**while(**p1 **!=** null**){**

p1 **=** p1**.**next**;**

p2 **=** p2**.**next**;**

**}**

**return** p2**;**

**}**

//Recursive implementation

public void kthToLast**(**Node root**,** int k**){**

**if(**root**==null)**

**return** 0**;**

int index **=** kthToLast**(**root**.**next**,**k**)+**1**; //Making the count backward**

**if(**root**.**next **==** k**){**

System**.**out**.**println**(**"The "**+** k **+**" Element is "**+** root**.**data**);**

**}**

**return** index**;**

**}**

**//This implementation is in C++**

node**\*** nthToLast**(**node**\*** head**,** int k**,** int**&** i**){**

**if(**head **==** **NULL){**

**return** null**;**

**}**

node**\*** nd **=** nthToLast**(**head**->**next**,** k**,** i**);**

i **=** i **+** 1**;**

**if(**i **==** k**){**

**return** nd**;**

**}**

**return** nd**;**

**}**

node**\*** nthTolast**(**node**\*** head**,** int k**){**

int i **=** 0**;**

**return** nthToLast**(**head**,** k**,** i**);**

**}**