***Circular Linked List Traversal***

In this post, traversal operation is discussed.



**Using Do While Loop:**

In a conventional linked list, we traverse the list from the head node and stop the traversal when we reach NULL. In a circular linked list, we stop traversal when we reach the first node again. Following is the C code for the linked list traversal.

C++Java

/\* Function to print nodes in a

given Circular linked list \*/

import java.util.\*;

static void printList(Node head)

{

Node temp = head;

// If linked list is not empty

if (head != null)

{

// Keep printing nodes till we reach the first node

// again

do

{

System.out.print(temp.data + " ");

temp = temp.next;

} while (temp != head);

}

}

**Time Complexity:** O(n)  
**Auxiliary Space:** O(1)

**Using For Loop:**

C++Java

import java.util.\*;

import java.io.\*;

import java.lang.\*;

class Node{

int data;

Node next;

Node(int d){

data=d;

next=null;

}

}

class Test {

public static void main(String args[])

{

Node head=new Node(10);

head.next=new Node(5);

head.next.next=new Node(20);

head.next.next.next=new Node(15);

head.next.next.next.next=head;

printlist(head);

}

public static void printlist(Node head){

if(head==null)return;

System.out.print(head.data+" ");

for(Node r=head.next;r!=head;r=r.next)

System.out.print(r.data+" ");

}

}

**Time Complexity:** O(n)  
**Auxiliary Space:** O(1)