

Suppose you are given two circularly linked lists, L and M. Describe an algorithm for telling if L and M store the same sequence of elements (but perhaps with different starting points).

Input :

Circular Linked List L : 10 → 20 → 30 → 40 → head

Circular Linked List M : 20 → 30 → 40 → 10 → head

SameSequence(L,M)

Start1=L.tail

Start2=M.tail

Exitcond=M.tail

SameSequence(M,L)

While(start1.data != start2.data)

start2=start2.next

if(Exitcond= start2)

return false;

break;

end while

While(start1.next != L.tail)

If(start1.data != start2.data)

return false;

break;

start2=start2.next

start1=start1.next

return true

Given a circularly linked list L containing an even number of nodes, describe how to split L into two circularly linked lists of half the size.

Circular Linked List L : 10 → 20 → 30 → 40 → 50 → 60 → head

splitHalf(LL)

L,M

st=tail

count=1

while(st.next!=tail)

count++

st=st.next

end while

st=tail.next

for i from 1 to count/2

L.addLast(st)

st=st.next

end for

for i from 1 to count/2

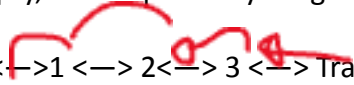
M.addLast(st)

st=st.next

end for

end

*Given a pointer to the head of a Doubly-linked list, print each value from the reversed list. If the given list is empty, do not print anything.

Linked List : Head  1 <-> 2 <-> 3 <-> Trailer

Example Output Reversed Linked list : 3 —> 2—> 1

```
printreverse()
  current = trailer.prev
  while current != header
    print(current.data)
    current=current.prev
  end while
```

A Doubly Linked List that contains a string , write a function that checks and returns “True” if the string is a palindrome. A string that is equal to its reverse is a palindrome; i.e. if you reverse the letters the word stays the same.

Example :

Input : madam DLL: head <-> m <-> a <-> d <-> a <-> m <->tail Output : True Input : civic DLL: head <-> c <-> i <-> v <-> i <-> c <->tail-> Output : True Input : kayak Output : True

isPalindrome (DLL):

```
s=DLL.header.next
e= DLL.tailer.prev
WHILE s!=e:
  if s.data!=e.data
    return false
  end if
  s =s.next
  e =e.prev
end while
return true
```

How do you insert a node at the beginning of the DLL list?

a)

```
public class insertFront(int data)
{
    ✓ Node node = new Node(data, head, head.getNext());
    node.getNext().setPrev(node);
    head.setNext(node);
    size++;
}
```

Write an algorithm that prints alternant nodes first from head to tail , second from tail to head
Example:

if the given list is : 10 -> 40 -> 30 -> 60 Output 1 (from head to tail) : 10 , 30 Output 2 (from tail to head) : 60 , 40

```
printAlter()
c=head
i=1
WHILE c!= NULL:
    if i mod 2==1
        print (c.data
    c=c.next
    i=i+1
end while

-----
printAlterFromtail(head,size)
if head=null
    return
else
    printAlterFromtail(head.next,size-1)
    if size mod 2==1
        print(head.data)
```

end if