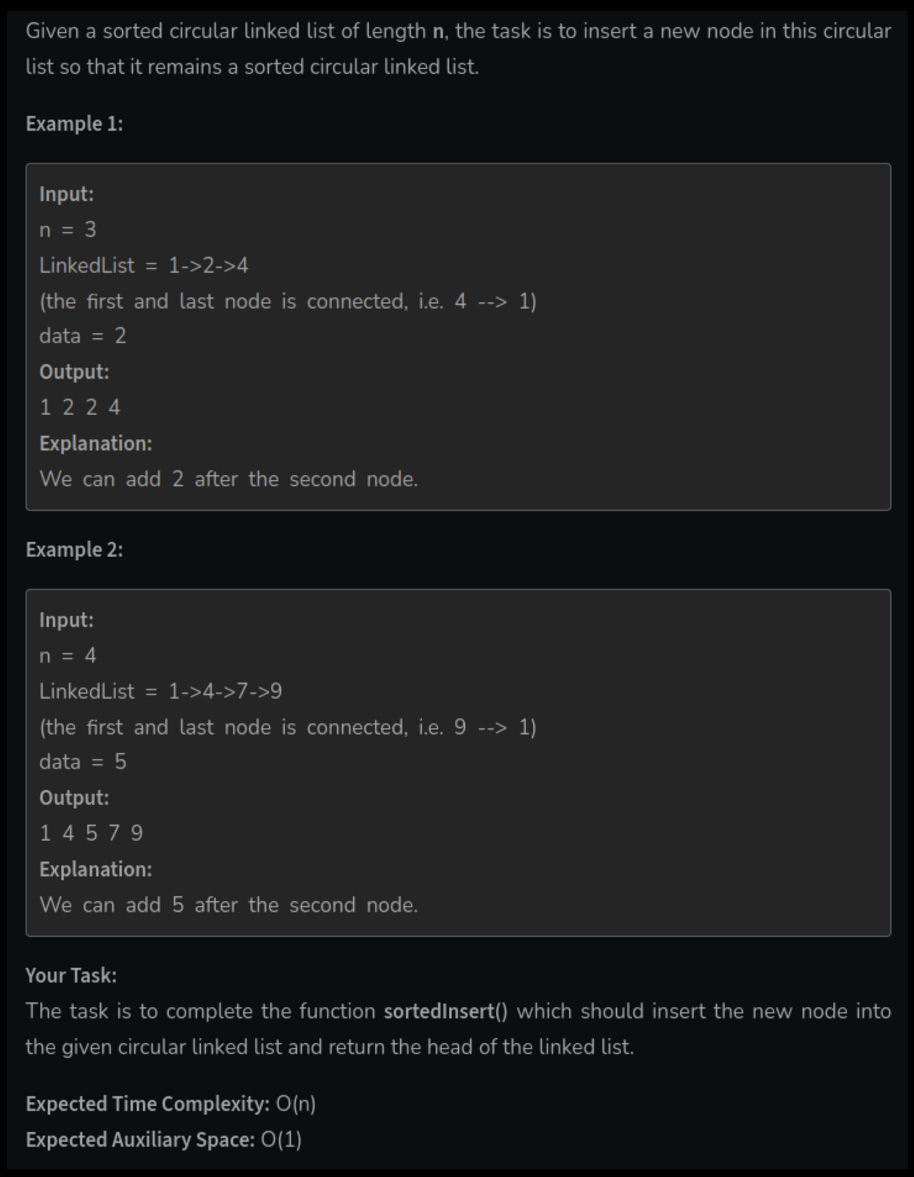
Sorted Insert In Circular Linked List



Given a sorted circular linked list of length **n**, the task is to insert a new node in this circular list so that it remains a sorted circular linked list.

Example 1:

```
Input:
n = 3
LinkedList = 1->2->4
(the first and last node is connected, i.e. 4 --> 1)
data = 2
Output:
1 2 2 4
Explanation:
We can add 2 after the second node.
```

Example 2:

```
Input:
n = 4
LinkedList = 1->4->7->9
(the first and last node is connected, i.e. 9 --> 1)
data = 5
Output:
1 4 5 7 9
Explanation:
We can add 5 after the second node.
```

Your Task:
The task is to complete the function **sortedInsert()** which should insert the new node into the given circular linked list and return the head of the linked list.

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

$$\begin{tikzcd}1 \arrow[r] & 2 \arrow[r] & 4 \arrow[ll, bend left]\end{tikzcd}$$

2.

Insert at the position where

temp → data > data.

→ Insert just before it.

```
class Solution
{
    public:
    Node *sortedInsert(Node* head, int data)
    {
       if(head==NULL){
           Node*temp=new Node(data);
           temp->next=temp;
           return temp;
       }
       if(head->data>=data){
           Node*temp=head;
           Node*ptr=new Node(data);
           while(temp->next!=head){
               temp=temp->next;
           }
           temp->next=ptr;
           ptr->next=head;
           return ptr;

       }
       Node*temp=head,*prev=NULL;
       while(temp->next!=head and temp->data<data){
           prev=temp;
           temp=temp->next;

       }
       Node*ptr=new Node(data);
       if(data>temp->data){
           ptr->next=temp->next;
           temp->next=ptr;
       }
       else{
           prev->next=ptr;
           ptr->next=temp;
       }
       return head;

    }
};
```