



## **WORKSHEET4**

NAME- NABEEN VERMA

**SEC-DWWC 43** 

**UID:20BCS7422** 

Date- 06/01/2023

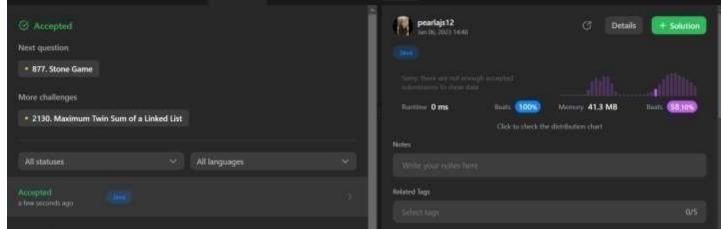
Q1) ADD TWO NUMBERS https://leetcode.com/problems/add-two-numbers/description/

```
addTwoNumbers(ListNode 11, ListNode 12)
                          111=11;
        {ListNode
        ListNode 112=12;
      ListNode dummy=new ListNode(0);
    ListNode d=new ListNode();
    d=dummy;
    int carry=0; while(ll1!=null
    || 112!=null)
        int x = (111 != null) ? 111.val : 0;
        int y = (112 != null) ? 112.val : 0;
        int sum = carry + x + y;
        d.next=new ListNode(sum%10);
        carry=sum/10;
        if(ll1 != null)
            ll1=ll1.next;
        if(112 != null)
        112=112.next;
        d=d.next;
     if (carry > 0) {
        d.next = new
    ListNode(carry); } return
    dummy.next;
```









```
O2)
class Solution
 ListNode getMid(ListNode head) { ListNode
               slow = head, fast = head;
               while (fast != null) {
                      slow = slow.next;
                      fast = fast.next == null ? null : fast.next.next;
               } return
               slow;
       }
       ListNode reverse(ListNode head) {
               ListNode prev = null, curr = head, next = head.next;
               while (curr != null) { curr.next = prev; prev = curr;
               curr = next; if (next != null) next = next.next;
               } return
               prev;
       }
       boolean isPalindrome(ListNode head)
               {if (head == null) return false;
               ListNode mid = getMid(head);
               if (mid != null) // this is to handle when there is only 1 element
                      mid = reverse(mid);
               ListNode pointer_1 = head, pointer_2 = mid;
               while (pointer_1 != null && pointer_2 != null)
               { if (pointer_1.val != pointer_2.val) return false;
                      pointer_1 = pointer_1.next;
                      pointer_2 = pointer_2.next;
               }
               return true;
       }
```



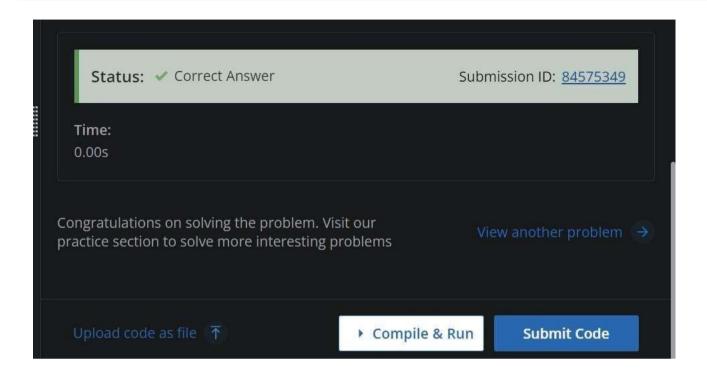


}

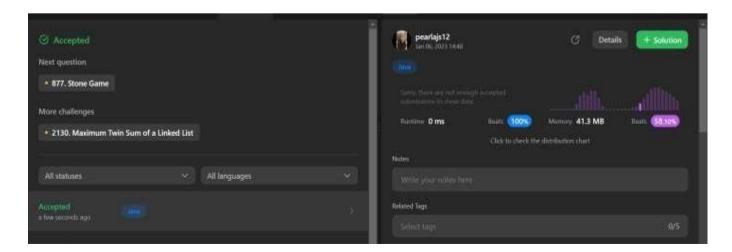
```
Q3) TEMPLE LAND
Ans) #include
<br/>
<br/>
dits/stdc++.h> using
namespace std;
int main() {
       // ASHISH RANA
       int t; cin>>t; while(t--){
               n;
                       cin>>n;
       vector<int>a(n);
       for(auto &i:a)cin>>i;
          if(n&1){ bool
            flag=1;
            for(int
               i=0;i<=n/2;i++)\{if(i+
              1!=a[i])flag=0;
            }
            for(int
            i=n/2+1; i< n; i++) \{if(ni!=
            a[i]) flag=0; }
            cout<<(flag?"yes":"no")<<'\n';
          } else
       cout << "no \n"; }
       return 0;
}
```







## Q4) MIDDLE OF LINKED LIST









## Q5) https://leetcode.com/problems/sort-list/

```
class
        Solution
                         public
                                   ListNode
    sortList(ListNode head) { if (head ==
    null || head.next == null) return head;
        ListNode mid = getMid(head);
        ListNode left = sortList(head);
        ListNode right = sortList(mid);
        return merge(left, right);
    ListNode merge(ListNode list1, ListNode list2)
      {if (list1 == null) { return
            list2;
        } if (list2 ==
        null)
            {return list1;
        ListNode head1=list1;
        ListNode head2=list2;
        ListNode dummy;
        ListNode head3;
            if(head1.val<head2.val)</pre>
                head3=dummy=new ListNode(head1.val);
                head1=head1.next;
            } else{ head3=dummy=new
            ListNode(head2.val); head2=head2.next;
            }
        while (head1 != null && head2 != null) { if
        (head1.val < head2.val) { head3.next = new</pre>
        ListNode(head1.val); head1 = head1.next;
            } else {
               head3.next = new ListNode(head2.val);
                head2 = head2.next;
```







**SORT LIST** 



head3=head3.next;



```
while(head1!=null)
            head3.next=new ListNode(head1.val);
            head1=head1.next;
          head3=head3.next;
    while(head2!=null)
    { head3.next=new ListNode(head2.val);
        head2=head2.next;
        head3=head3.next;
   return dummy;
ListNode getMid(ListNode head) {ListNode midPrev = null;
    while (head != null && head.next != null) { midPrev =
    (midPrev == null) ? head : midPrev.next; head =
    head.next.next;
    ListNode mid = midPrev.next;
    midPrev.next = null; return
    mid;
```







