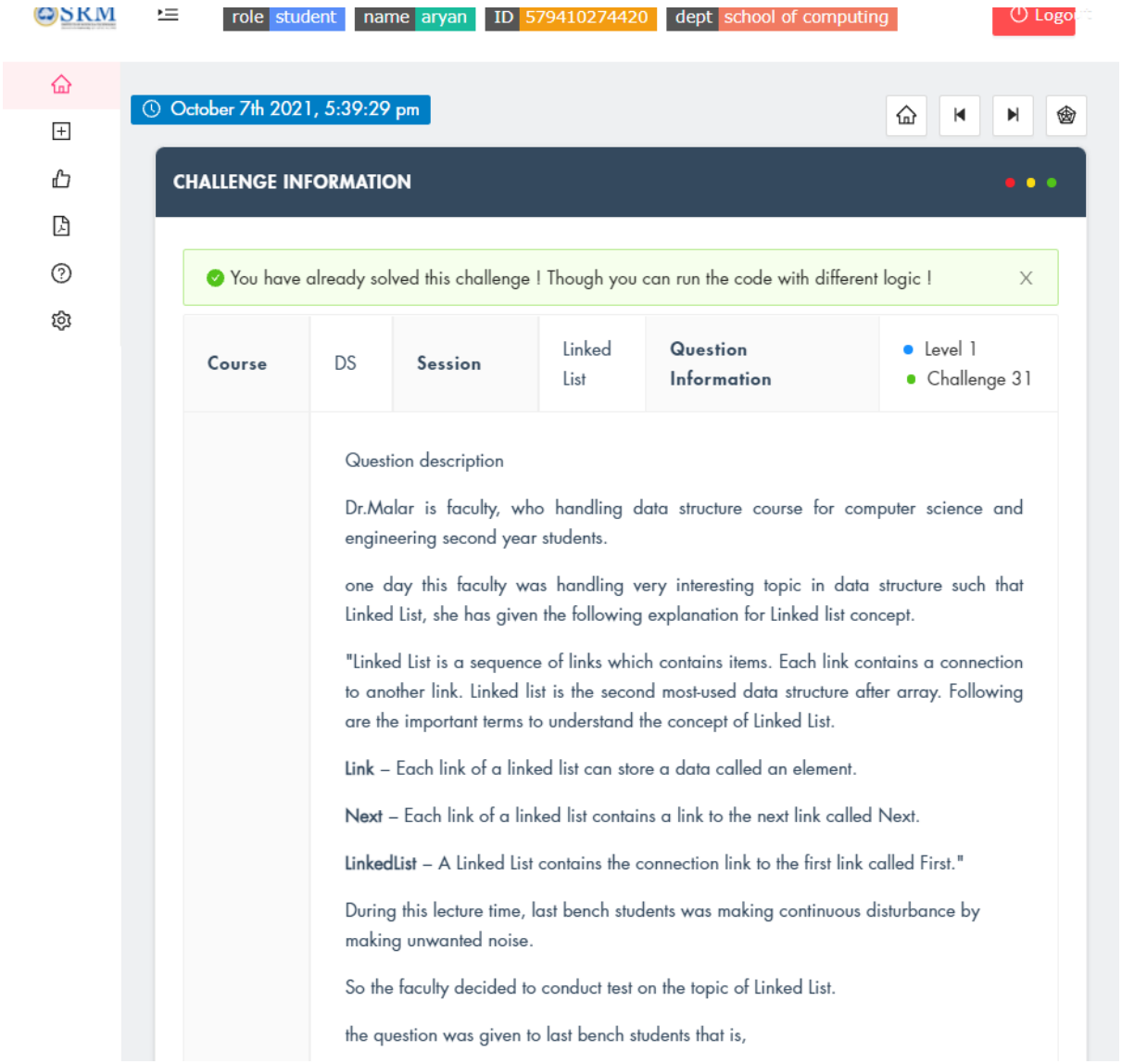
LINKED LIST:

1.

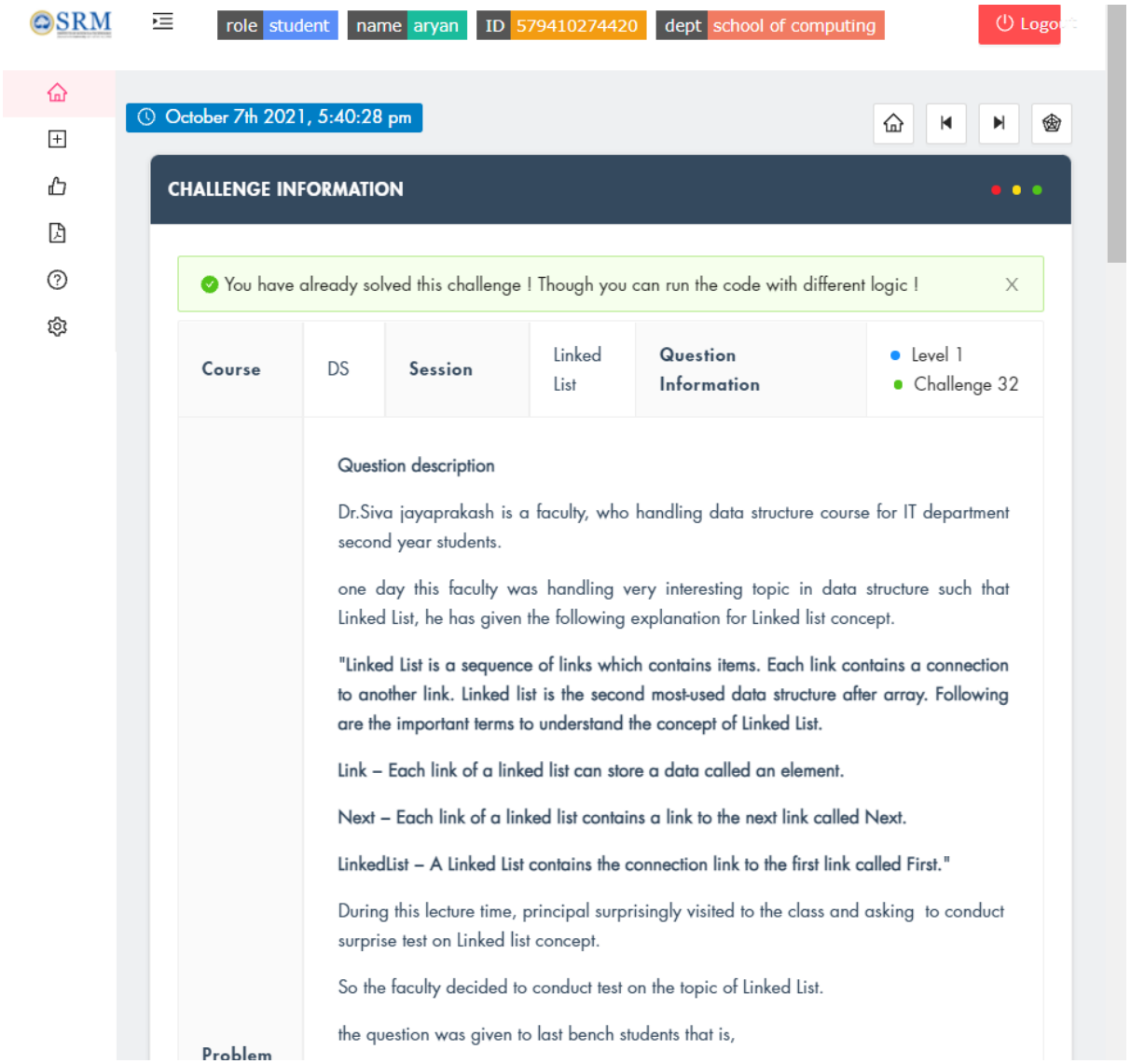



```
#include <bits/stdc++.h>
using namespace std;
struct node
{
    int data;
    struct node *next;
}*head = NULL;
  int n;
int in_pos(int n)
{
    int data1;
    cin>>data1;
    int i =1;
    struct node *r = head;

    while(i != n-1)
```

```
    {
          r =  r-> next;
          i++;
    }

    node *tt = new node;
    tt -> data = data1;
    tt -> next = r -> next;
    r -> next = tt;
     node *s = head;
cout<<"Linked List:";
      while(s != NULL)
      {
         cout<<"->";
         cout<<s-> data;
         s = s-> next;
      }
    return data1;
}

void create()
{

    int n;
     cin>>n;
     struct node *p = new node;
     int __n;
     cin>>__n;
     p -> data = __n;
     head = p;

    int i;
    for(i=0;i<n-1;i++)
    {
        int a;
        cin>>a;
         struct node *q = new node;
        q -> data = a;
        p -> next= q;
        p = p->next;
    }
    p -> next = NULL;
}
int main()
{
    create();
   int r;
   cin>>r;
```

```
    int s = in_pos(r);
        return 0;
        cout<<s<<"for(i=0;i<n;i++)";
}
```

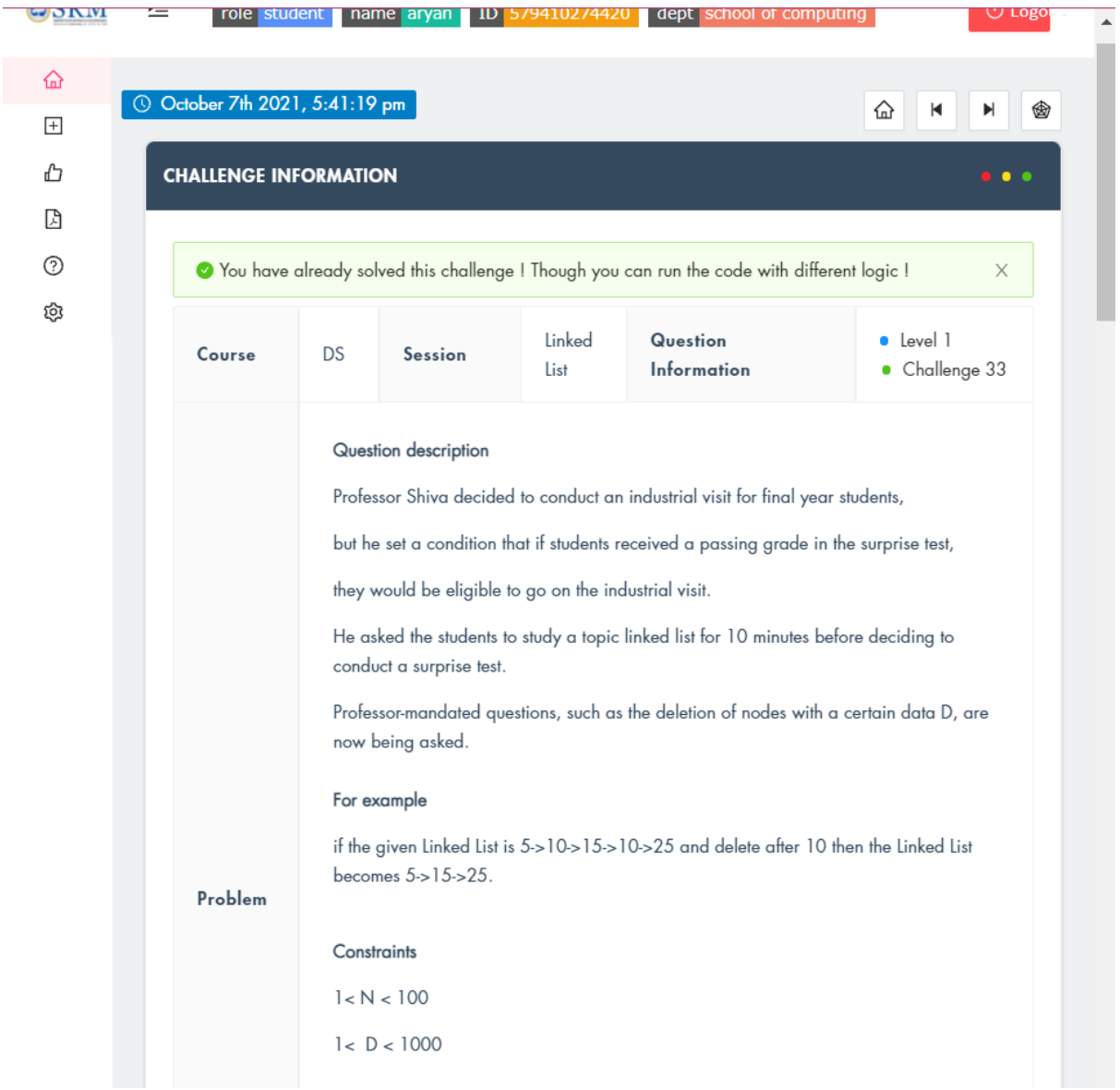
2.




```
#include <iostream>
using namespace std;
void tel()
{
   return;
}
struct node
{
   int data;
   node *next;
```

```
}*head = NULL;
void create()
{
   int n;
   cin>>n;
   struct node *p1 = new node;
   int m;
   cin>>m;
   p1 -> data = m;
   head = p1;
   int i;
   for(i=0;i<n-1;i++)
   {
      int a;
      cin>>a;
      node *tt = new node;
      tt -> data = a;
      p1 -> next = tt;
      p1=p1->next;

   }
   p1 -> next = NULL;
   int del;
   bool found = false;
   cin>>del;
   node *nn = head;
   while(nn != NULL)
   {
      nn = nn -> next;
      node *dd = nn;
      int m = del;    while(m-- > -1)
      {
         dd = dd -> next;    if(dd == NULL)
          {
             nn -> next = NULL;
             found = true; break;}}
      if(found) break;    }
   cout<<"Linked List:";
   while(head != NULL)
   {
      cout<<"->"<<head -> data;
      head = head -> next;  }}
int main()
{
   create();
         return 0;
         cout<<"for(i=0;i<n;i++)";
}
```

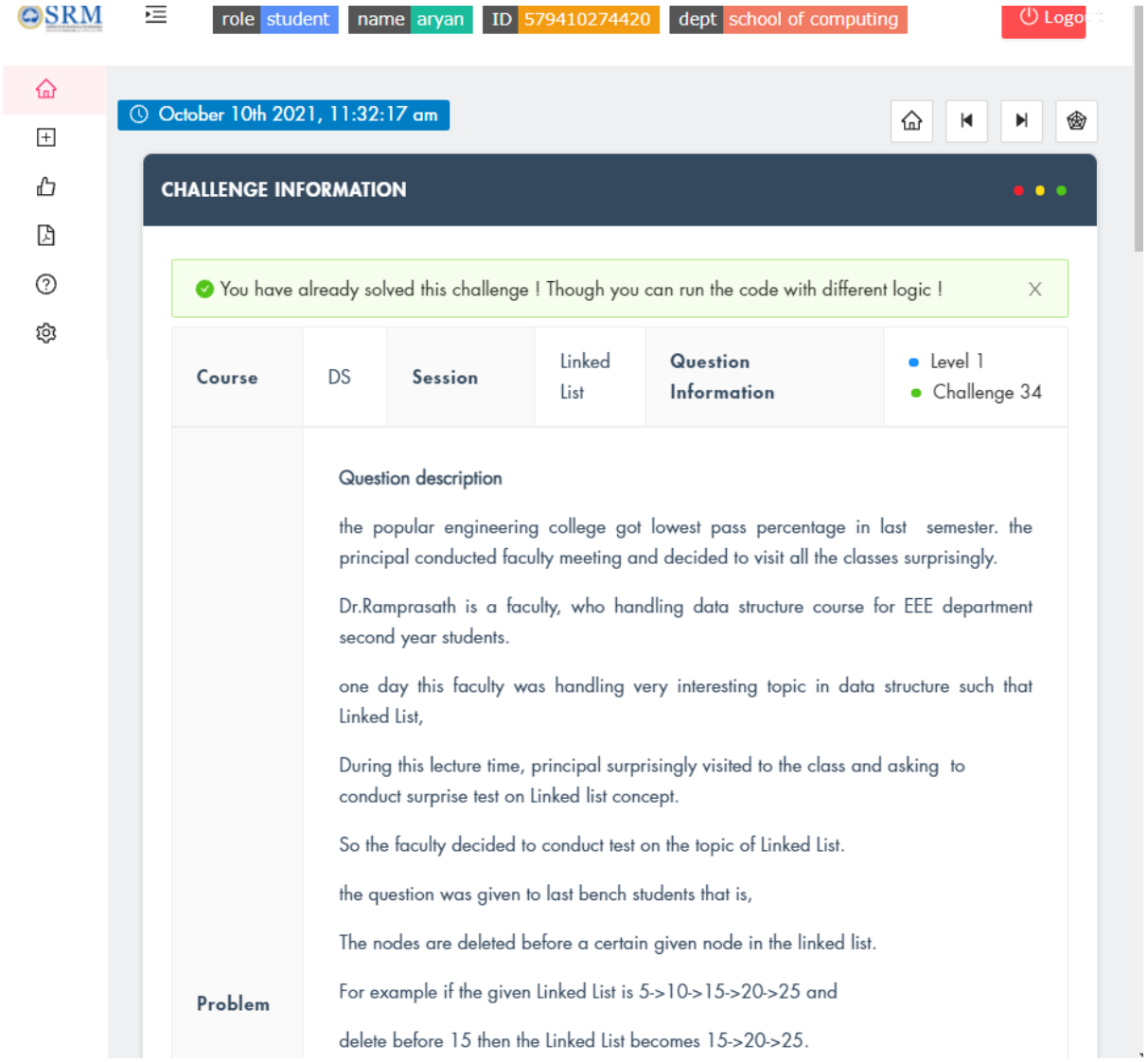
3.




```
#include <bits/stdc++.h>
using namespace std;
void ss()
{
   return;
}
int main()
{
   int n;
   cin>>n;
   int arr[n];
   for (int i = 0; i < n; ++i)
   {
      cin>>arr[i];
   }
  int m;
```

```
   cin>>m;
   cout<<"Linked List:";
    for(int p : arr)
       {
          if(p != m)
            cout<<"->"<<p;
       }
    return 0;
    cout<<"struct node node *next; void create() p2=p2->next; void del()";
}
```

4.

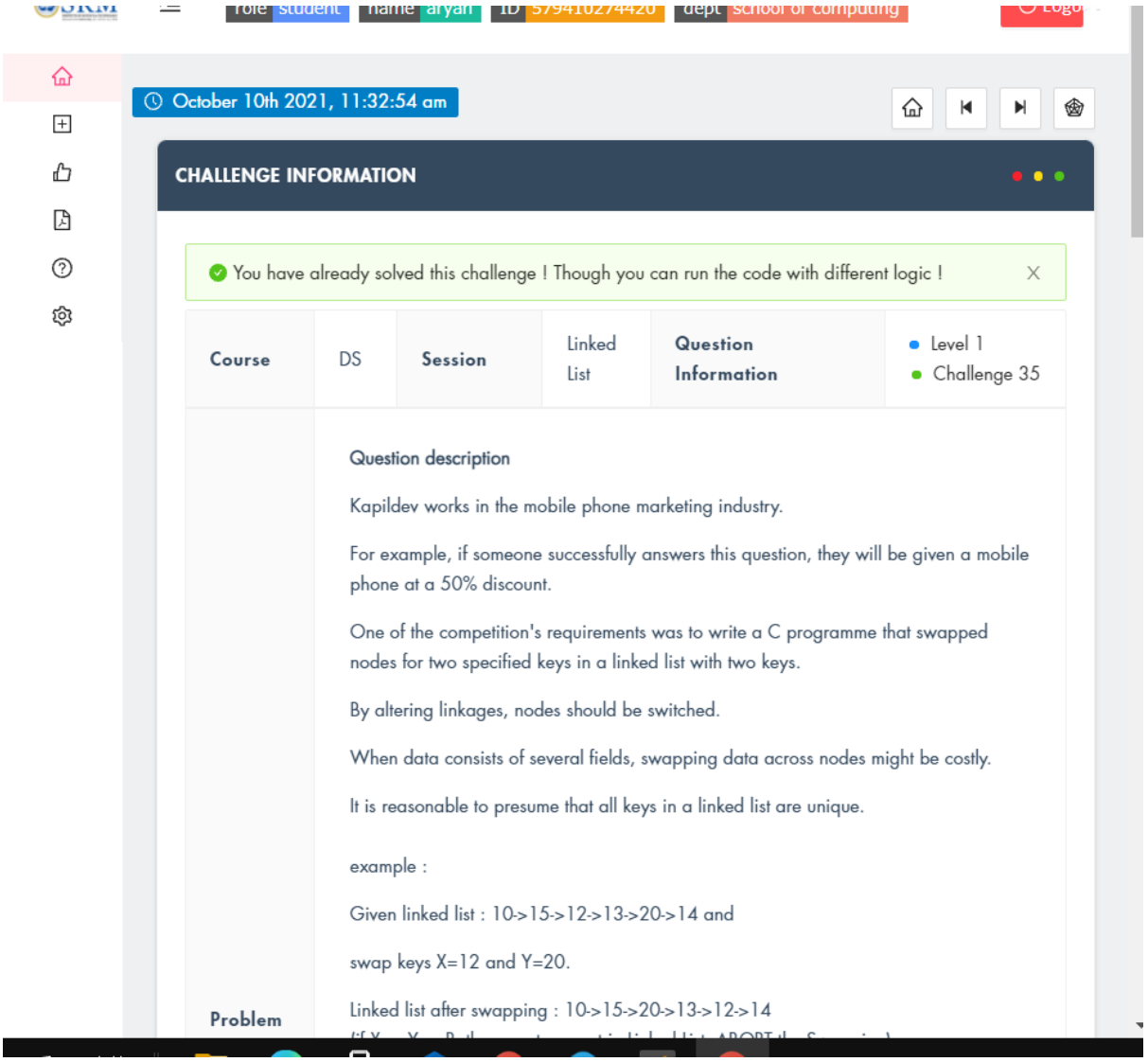



```
#include <iostream>
using namespace std;
void ss(){
   return;
}
struct node
{
```

```
    int data;
    node *next;
}*head  = NULL;
bool found = true;
 int n;
void del()
{
    int n,i=0;
    cin>>n;
    node *j = head;
    while (j != NULL)
    {   i++;
        if( j -> next -> data == n)
        {
            head = j -> next;
            break;
        }
        j = j -> next;
         if(i == n)
         {
             found = false;
             break;
         }
    }
    if(!found) cout<<"Invalid Node! ";
     cout<<"Linked List:";
    while(head != NULL)
        cout<<"->"<<head -> data,
        head = head -> next;}
void create()
{
    int n,i=0,first;cin>>n;node *p1 = new node;cin>>first;p1 -> data = first;
    head = p1;
    while(i!=n-1)
    {
        int a;
        cin>>a;
        node *n = new node;
        n -> data = a;
        n -> next = NULL;
        p1 -> next = n;
        p1 = n;
        i++;
    }
    p1 -> next = NULL;
}
int main()
{
```

```
    create();
    del();return 0;cout<<"p1=p1->next for(i=0;i<n;i++) p1=p1->next";
}
```

5.

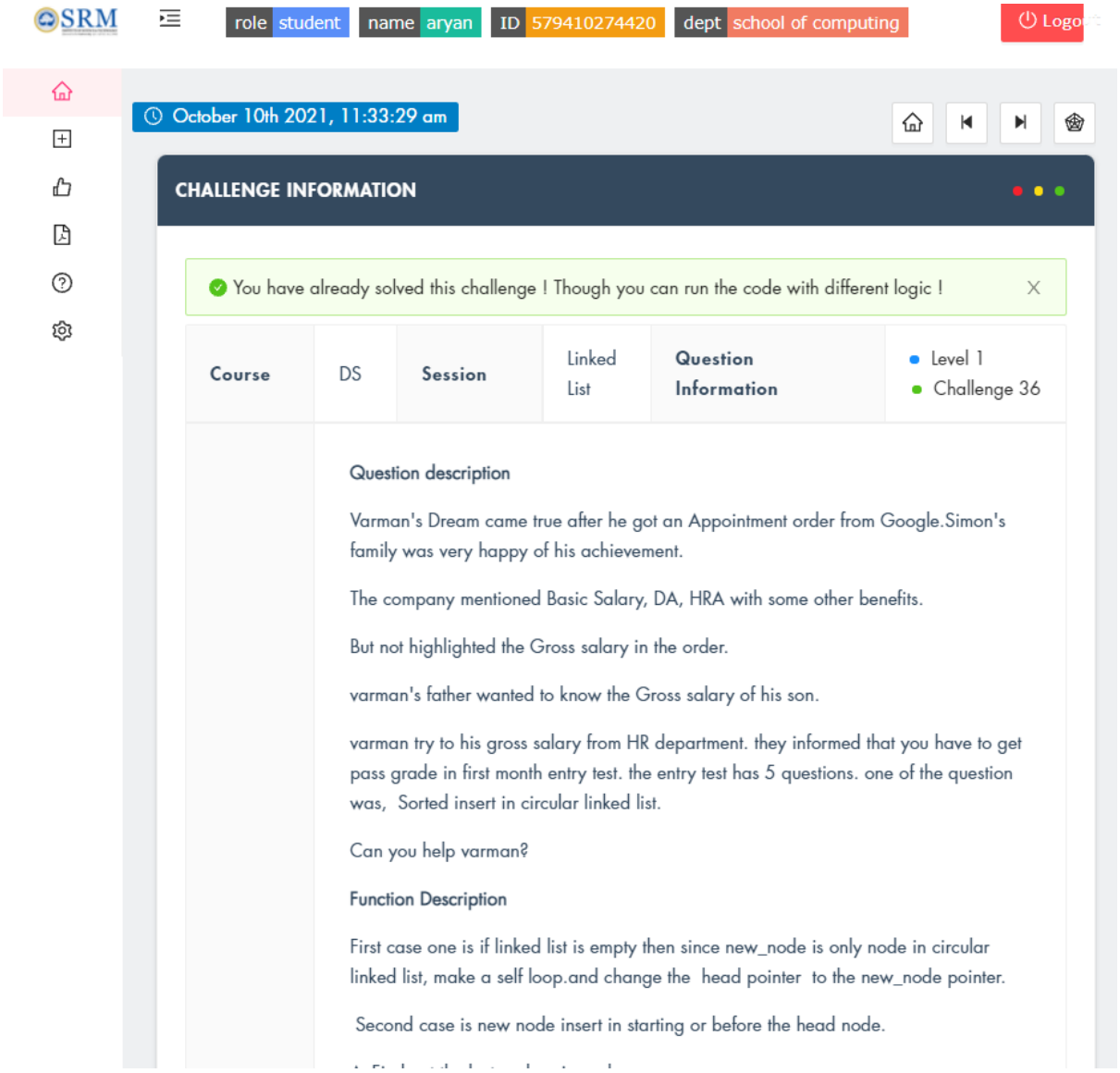



```
#include <iostream>
using namespace std;
struct node
{
    int data;
    struct node *next;
}*head = NULL;
void display(node *ss)
{
    if(ss == NULL) return;
    display(ss -> next);
    cout<<"-->"<<ss -> data;
}
void swapNodes(struct node **head_ref,int x,int y)
{
```

```
    if (x == y)
        return;
    node *prevX = NULL, *currX = *head_ref;
    while (currX && currX->data != x) {
        prevX = currX;
        currX = currX->next;
    }
    node *prevY = NULL, *currY = *head_ref;
    while (currY && currY->data != y) {
        prevY = currY;
        currY = currY->next;
    }
    if (currX == NULL || currY == NULL)
        return;
    if (prevX != NULL)
        prevX->next = currY;
    else
        *head_ref = currY;
    if (prevY != NULL)
        prevY->next = currX;
    else
        *head_ref = currX;
    node* temp = currY->next;
    currY->next = currX->next;
    currX->next = temp;
}
void create()
{
    int n;cin>>n;
    int rr;cin>>rr;
    node *tt = new node;tt -> data = rr;
    tt -> next = NULL;head = tt;
    int i;
    for(i=0;i<n-1;i++)
    {
        int a;
        cin>>a;
        node *q = new node;
        q -> data = a;
        q -> next = NULL;
        tt -> next = q;
        tt = q;
    }
}
int main()
{create();
 cout<<"before Swapping:";
  display(head);
```

```
 int x,y;
 cin>>x>>y;
 swapNodes(&head,x,y);
 cout<<"\nafter Swapping:";
 display(head);
        return 0;
}
```

6.

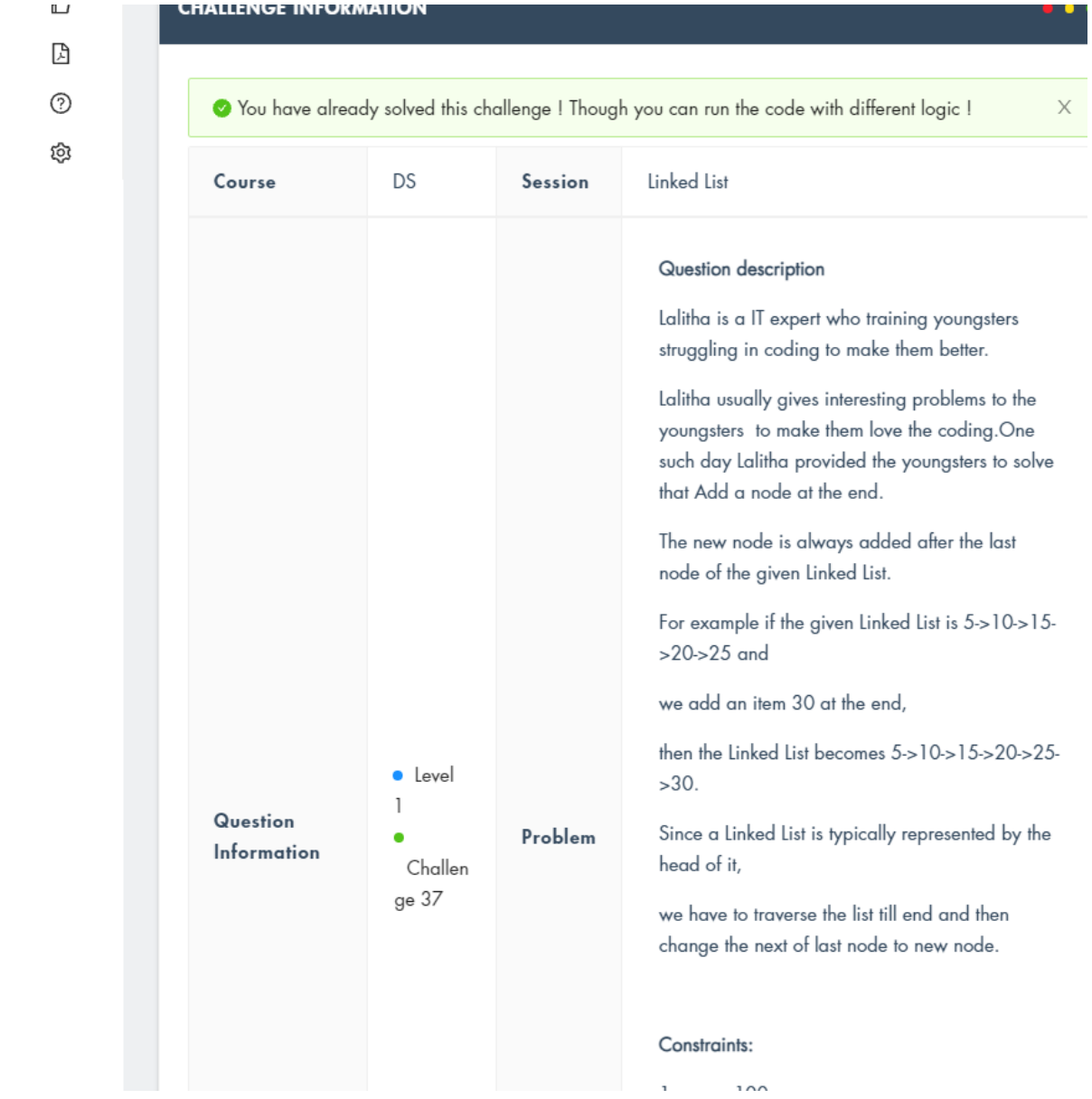



```
#include <bits/stdc++.h>
using namespace std;
struct Node {
    int data;
    struct Node *next;
}*head = NULL;
void sortedInsert(struct Node** head_ref, struct Node* new_node)
{
    Node* current;
```

```
    if (*head_ref == NULL || (*head_ref)->data
            >= new_node->data) {
        new_node->next = *head_ref;
        *head_ref = new_node;
    }
    else {
        current = *head_ref;
        while (current->next != NULL && current->next->data< new_node->data)
        {
            current = current->next;
        }
        new_node->next = current->next;
        current->next = new_node;
    }
}
Node* newNode(int new_data)
{

    Node* new_node = new Node();


    new_node->data = new_data;
    new_node->next = NULL;

    return new_node;
}

void display()
{
    Node* temp = head;
    while (temp != NULL) {
        if(temp-> next != NULL)
        cout<<temp->data<<" ";
        else
        cout<<temp -> data;
        temp = temp->next;
    }
}
int main()
{
    int j;
    cin>>j;
    int first;
    cin>>first;
    Node* new_node = newNode(first);
    sortedInsert(&head, new_node);
    for (int i = 0; i < j - 1 ; ++i)
    {
```

```
        int m;
        cin>>m;
        new_node = newNode(m);
    sortedInsert(&head, new_node);
    }
    display();

    return 0;
}
```

7.



You have already solved this challenge ! Though you can run the code with different logic !

| Course | DS | Session | Linked List |
|---|---|---|---|
| Question Information | Level 1 · Challenge 37 | Problem | Question description<br>Lalitha is a IT expert who training youngsters struggling in coding to make them better.<br>Lalitha usually gives interesting problems to the youngsters to make them love the coding.One such day Lalitha provided the youngsters to solve that Add a node at the end.<br>The new node is always added after the last node of the given Linked List.<br>For example if the given Linked List is 5->10->15->20->25 and<br>we add an item 30 at the end,<br>then the Linked List becomes 5->10->15->20->25->30.<br>Since a Linked List is typically represented by the head of it,<br>we have to traverse the list till end and then change the next of last node to new node.<br>Constraints: |

```
#include <iostream>
using namespace std;
struct node
{
    int data;
```

```
    node *next;
}*start = NULL;
void display()
{
    if(start == NULL) return;
    cout<<"->"<<start->data;
     start = start -> next;
    display();
}
void create()
{
    int n;
    cin>>n;  int first;cin>>first;
    node *p2 = new node;
    p2 -> data = first;
    p2 -> next = NULL;
    start = p2;
    for(int i =0; i<n-1; i++)
    {
        int a;
        cin>>a;
        node *yy = new node;
        yy -> data = a;
        yy -> next = NULL;
        p2 -> next = yy;
        p2=p2->next;
    }
}
int main()
{
    create();
    cout<<"Linked List:";
    display();

            return 0;
}
```

8.

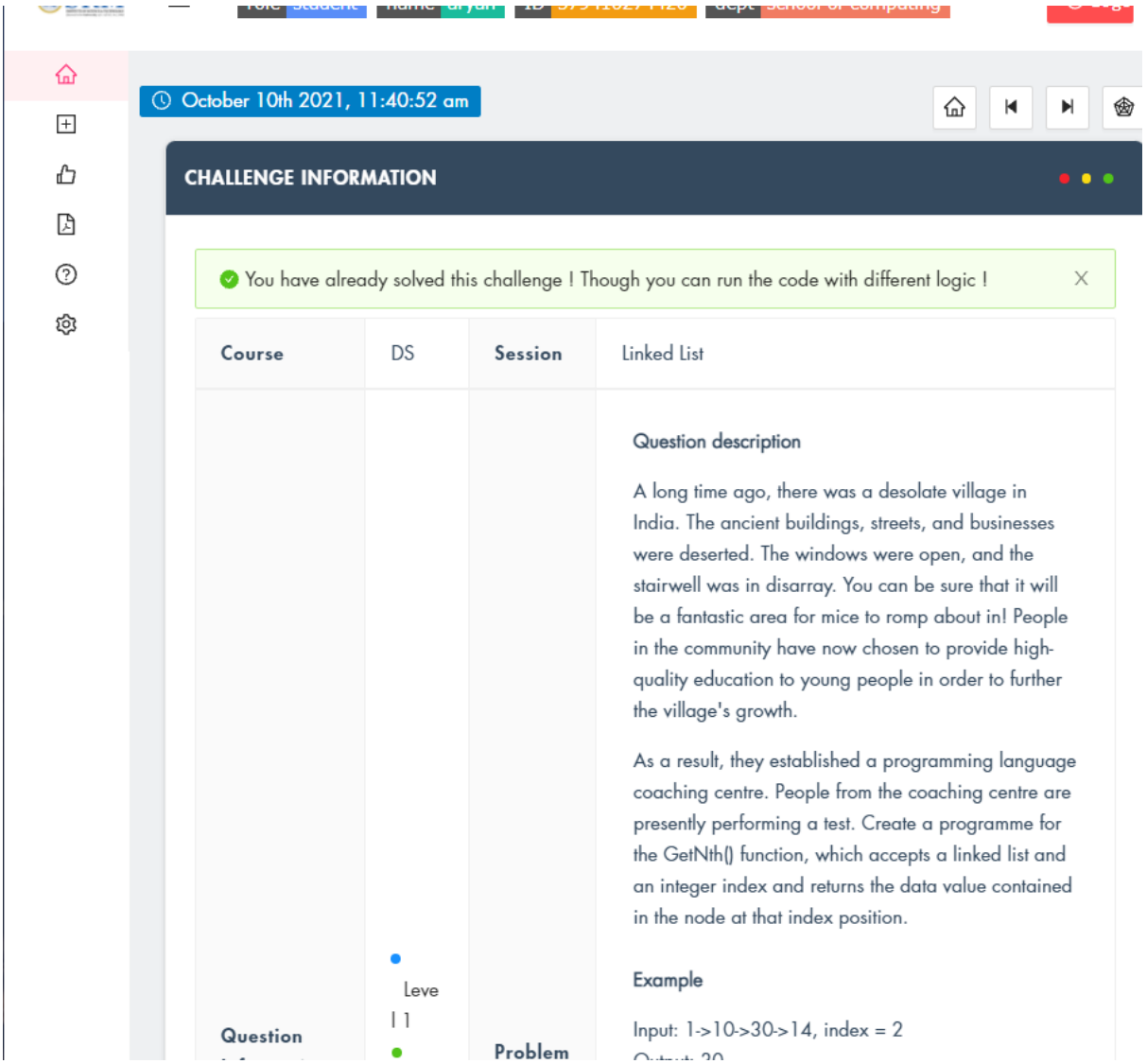
You have already solved this challenge ! Though you can run the code with different logic ! X

| Course | DS | Session | Linked List |
| --- | --- | --- | --- |
| Question Information | • Level 1<br>• Challenge 38 | Problem | Question description<br>Lalitha is a IT expert who training youngsters struggling in coding to make them better.<br>Lalitha usually gives interesting problems to the youngsters to make them love the coding.One such day Lalitha provided the youngsters to solve that The new node is always placed before the Linked List's head.<br>The newly inserted node becomes the Linked List's new head.<br>If the current Linked List is 11->151->201->251, for example,<br>We add item 5 to the front of the list.<br>The Linked List will then be 5->11->151->201->251.<br>Let's call the function that moves the item to the top of the list push ().<br>The push() must receive a pointer to the head pointer, because push must change the head pointer to point to the new node<br>Constraints:<br>1 < arr <100 |

```
#include <iostream>
using namespace std;
void ff()
{
   return;
}
struct node
{
   int data;
   node *next;
```

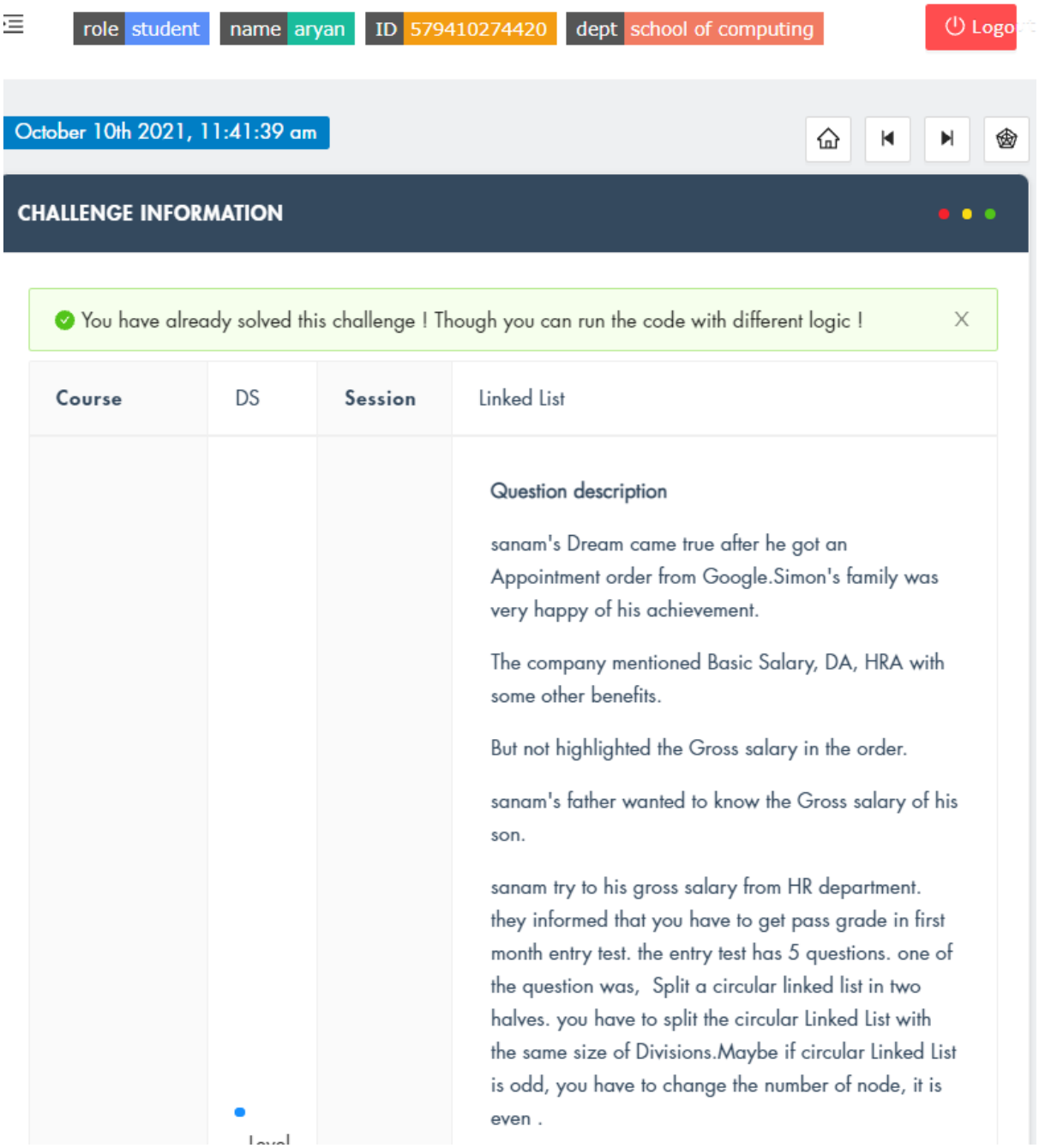
```
}*start = NULL;
int main()
{
    int n;
    cin>>n;
    int arr[n];
    for(int i =0; i<n;i++)
    {
        cin>>arr[i];
    }
    cout<<"Linked List:";
    for(int j = n-1;j >= 0 ; j--)
    cout<<"->"<<arr[j];
//    display(start);
          return 0;
          cout<<"struct node node *next; *start p1->next=start; void display()";
}
```

9.




```
#include <iostream>
using namespace std;
struct node
{
   int data;
   struct node *next;
}*head = NULL;
int i = 0,n;
int GetNth(struct node* head,int index)
{
   while(n-i != index)
   {
       head = head -> next;
       i++;
       if(i == index) break;
   }
   return head -> data;
```

```
}
void display(node *u)
{
    if(u == NULL) return;
    display(u -> next);
    cout<<"-->"<<u -> data;
}
int main()
{
     int first;
    cin>>n>>first;
    node *t = new node;
    t -> data = first;
    t -> next = NULL;
    head = t;
    for(int i = 0; i< n-1 ; i++)
    {
        cin>>first;
        node *u = new node;
        u -> data = first;
        u -> next = NULL;
        t -> next = u;
        t = u;
    }
    int index;
    cin>>index;
    node *hh = head;
    cout<<"Linked list:";
    display(head);
    cout<<"\nNode at index="<<index<<":"<<GetNth(hh,index);
          return 0;
}
```




```
#include <iostream>
using namespace std;
struct n
{
    int data;
    struct n *next;
}*odd,*even,*h = NULL,*tt;
void insert(int data)
{
     n *p = new n;
    p -> data = data;
```

```
    p -> next = NULL;
    tt -> next = p;
    tt = p;

}
void oodd()
{
    cout<<"Odd:\n";
    odd  = h;
    int i =1;
    cout<<"[h]";
    while(odd != NULL)
    {
        if((i%2))
        {
            cout<<"=>"<<odd -> data;
        }
        i++;
        odd = odd -> next;
    }
    cout<<"=>[h]";

}
void eeven()
{
    cout<<"Even:\n";
    even  = h;
    int i =1;
    cout<<"[h]";
    while(even != NULL)
    {
        if(!(i%2))
        {
            cout<<"=>"<<even -> data;
        }
        i++;
        even = even -> next;
    }
    cout<<"=>[h]";

}
void display(struct n *h)
{
    cout<<"Complete linked_list:\n[h]";
    while(h != NULL)
    {
        cout<<"=>"<<h -> data;
        h = h -> next;
```

```
    }
    cout<<"=>[h]";
}
int main()
{
int a;
cin>>a;
tt = new n;
tt -> data = 1;
tt -> next = NULL;
h = tt;
for(int i =2; i<= a; i++)
{
   insert(i);
}
n *y = h;
display(y);
cout<<"\n";
oodd();
cout<<"\n";
eeven();
            return 0;
}
```