

Linked List

The screenshot shows a web browser window with the URL `care.srmup.in/srmncretelab/#/srmncretelab/student/home`. The page displays a coding challenge titled "Linked List" under the "DS" (Data Structures) session. The challenge is at "Level 1" and is "Challenge 31". A green notification bar at the top states: "You have already solved this challenge! Though you can run the code with different logic!".

Question description

Dr.Siva jayaprakash is a faculty, who handling data structure course for IT department second year students. one day this faculty was handling very interesting topic in data structure such that Linked List, he has given the following explanation for Linked list concept.

"Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. Following are the important terms to understand the concept of Linked List.

Link - Each link of a linked list can store a data called an element.

Next - Each link of a linked list contains a link to the next link called Next.

LinkedList - A Linked List contains the connection link to the first link called First."

During this lecture time, principal surprisingly visited to the class and asking to conduct surprise test on Linked list concept. So the faculty decided to conduct test on the topic of Linked List. the question was given to last bench students that is,

The nodes are deleted D times from the end of the given linked list.

For example if the given Linked List is 5->10->15->20->25 and remove 2 nodes, then the Linked List becomes 5->10->15.

Constraint :

$1 < N < 1000$
 $1 < P < N-1$

INPUT Format

First line contains the number of datas- N.
Second line contains N integers the given linked list.

```
#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(){

```

```

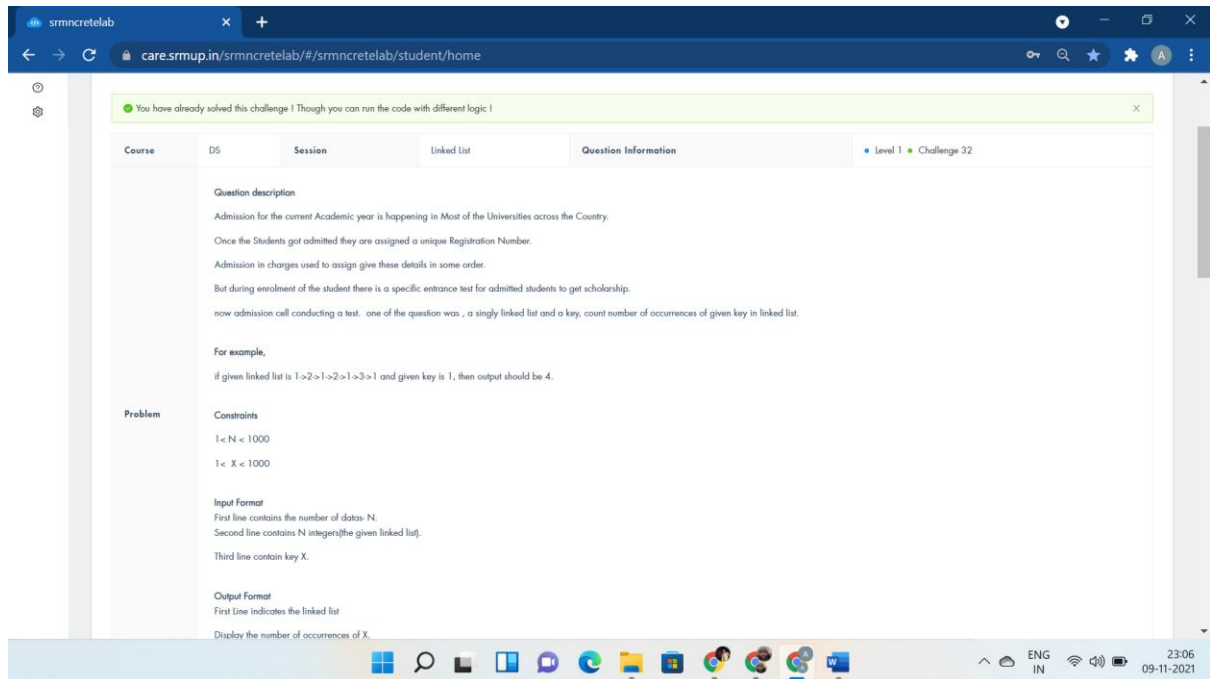
create();

return 0;

cout << "for(i=0;i<n;i++)";

}

```



```

#include <bits/stdc++.h>

using namespace std;

struct node
{
    int key;
    struct node *next;
};

void push(struct node** head_ref, int new_key)
{
    struct node* new_node = new node();
    new_node->key = new_key;
    new_node->next = (*head_ref);
    (*head_ref) = new_node;
}

```

```

void printList(node *node){
    while (node != NULL)
    {
        cout<<"--"<<node->key;
        node = node->next;
    }
}

int count(struct node* head,int search_for)
{
    node* current = head;
    int count=0;
    while (current != NULL)
    {
        if (current->key == search_for)
            count++;
        current = current->next;
    }
    return count;
}

int main()
{
    struct node* head = NULL;
    int x,n,t;
    cin>>n;
    while(n--){
        cin>>t;
        push(&head,t);
    }
    cin>>x;
    cout<<"Linked list:";
    printList(head);
    cout<<endl<<"Count of "<<x<<":"<<count(head, x);
    return 0;
}

```

}

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 33

Question description

Dr. Jagan is faculty, who handling data structure course for software engineering department second year students. one day this faculty was handling very interesting topic in data structure such that Linked List, he has given the following explanation for Linked list concept.

"Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. Following are the important terms to understand the concept of Linked List.

Link – Each link of a linked list can store a data called an element.

Next – Each link of a linked list contains a link to the next link called Next.

LinkedList – A LinkedList contains the connection link to the first link called First."

During this lecture time, last bench students was asking surprise test for Linked list concept. So the faculty decided to conduct test on the topic of Linked List.

the question was given to last bench students that is,

The nodes are deleted D times from the beginning of the given linked list.

For example if the given Linked List is 5->10->15->20->25 and remove 2 nodes, then the Linked List becomes 15->20->25.

Constraint :

$1 < N < 1000$
 $1 < P < N-1$

INPUT Format

First line contains the number of nodes- N.

```
#include<bits/stdc++.h>
```

```
using namespace std;
```

```
struct node {
```

```
    int data;
```

```
    node *next;
```

```
};
```

```
void insertAtEnd(node** head_ref, int new_data) {
```

```
    node* new_node = (node*)malloc(sizeof( node));
```

```
    node* last = *head_ref;
```

```
    new_node->data = new_data;
```

```
    new_node->next = NULL;
```

```
    if (*head_ref == NULL) {
```

```
        *head_ref = new_node;
```

```
        return;
```

```
    }
```

```
    while (last->next != NULL) last = last->next;
```

```
    last->next = new_node;
```

```
    return;
```

```
}
```

```
int main() {
```

```

node* head = NULL;

int n,c,z,i;

cin>>n;

for(i=0;i<n;i++){

    cin>>c;

    insertAtEnd(&head,c);

}

cin>>z;

for(int i=0;i<z;i++)

head=head->next;

cout<<"Linked List:";

node* node=head;

while(node!=NULL){

    cout<<"->"<<node->data;

    node=node->next;

}

return 0;

cout<<"void create()";

}

```

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 34

Question description

Once upon a time, in French Canada, there lived a fat old woman named Tante Adela.

She lived alone in her barn with her large grey cat and her cows.

She got up quite early one morning since it was baking day and she had a lot to accomplish.

She carried a pile of wood to her oven outdoors.

she ran across some old school classmates, with whom she reminisced about their school days and a mental exam competition.

One of the competition's requirements was to write a C function that searches a singly linked list for a given key "x." (Iterative).

If x is contained in the linked list, the function should return true; otherwise, it should return false.

For example,

if the key to be searched is 15 and linked list is 14>21>11>30>10,

then function should return false.

If key to be searched is 14, then the function should return true.

Constraints

1 < N < 1000

1 < X < 1000

Input Format

First line contains the number of datas- N.

Second line contains N integers[the given linked list].

Third line contains the key X to search.

```
#include <bits/stdc++.h>
```

```

using namespace std;

struct node
{
    int key;
    struct node* next;
};

void push(struct node** head_ref, int new_key)
{
    struct node* new_node = new node();
    new_node->key = new_key;
    new_node->next = (*head_ref);
    (*head_ref) = new_node;
}

bool search(struct node* head,int x)
{
    node* current = head;
    while (current != NULL)
    {
        if (current->key == x)
            return true;
        current = current->next;
    }
    return false;
}

int main()
{
    struct node* head = NULL;
    int x,n,t;
    cin>>n;
    while(n--){
        cin>>t;
        push(&head,t);
    }
}

```

```

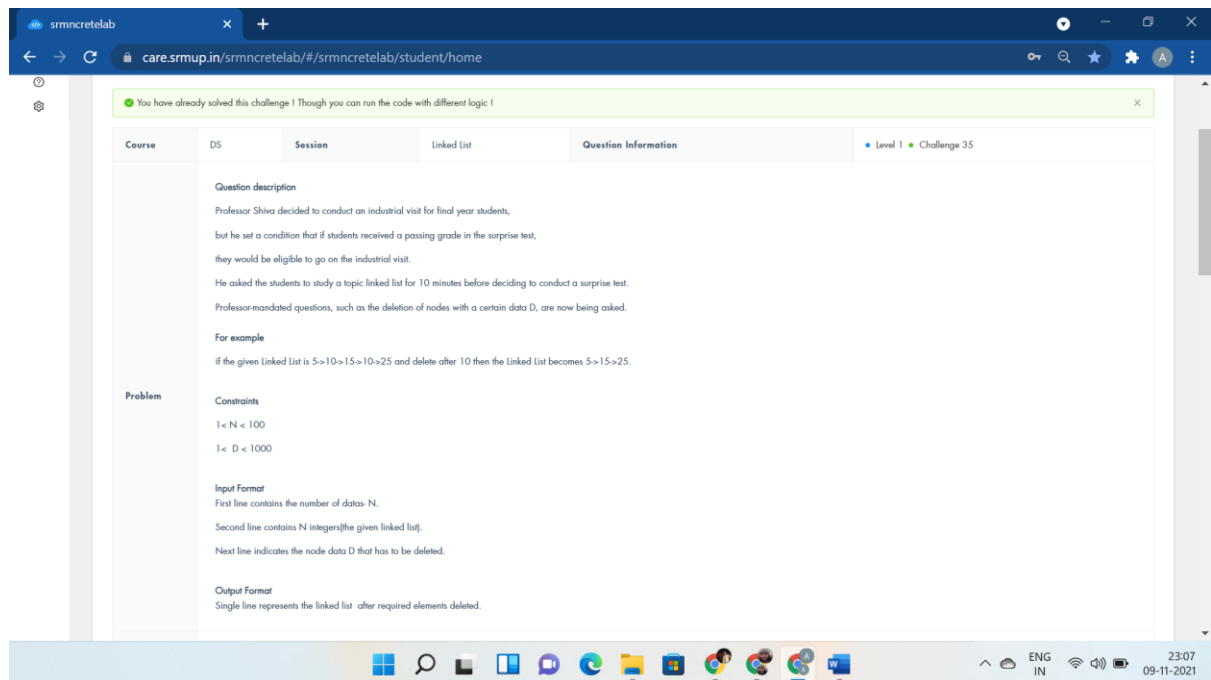
        cin>>x;

        search(head, x)? cout<<"Yes" : cout<<"No";

        return 0;

    }

```



```

#include<iostream>

using namespace std;

struct node{
    int data;
    struct node *next;
}*start;

void display();

void deleteNode(node*& head, int val)
{
    if (head == NULL) {
        return;
    }

    if (head->data == val) {
        node* t = head;
        head = head->next;
    }
}

```



```

        delete (t);

        return;
    }

    deleteNode(head->next, val);
}

int main() {
    int n;

    scanf("%d",&n);

    struct node *temp, *p2;

    start=NULL;

    for(int i=0;i<n;i++){

        temp=(struct node *)malloc(sizeof(struct node));

        scanf("%d", &temp -> data);

        temp->next = NULL;

        if(start == NULL){

            start= temp;

            p2 = temp;

        }

        else

        {

            p2->next=temp;

            p2=p2->next;

        }

    }

    int x;

    cin>>x;

    //display();

    for(int i=0;i<n;i++)

        deleteNode(start,x);

    display();

    return 0;

    cout<<"void del()void create() ";

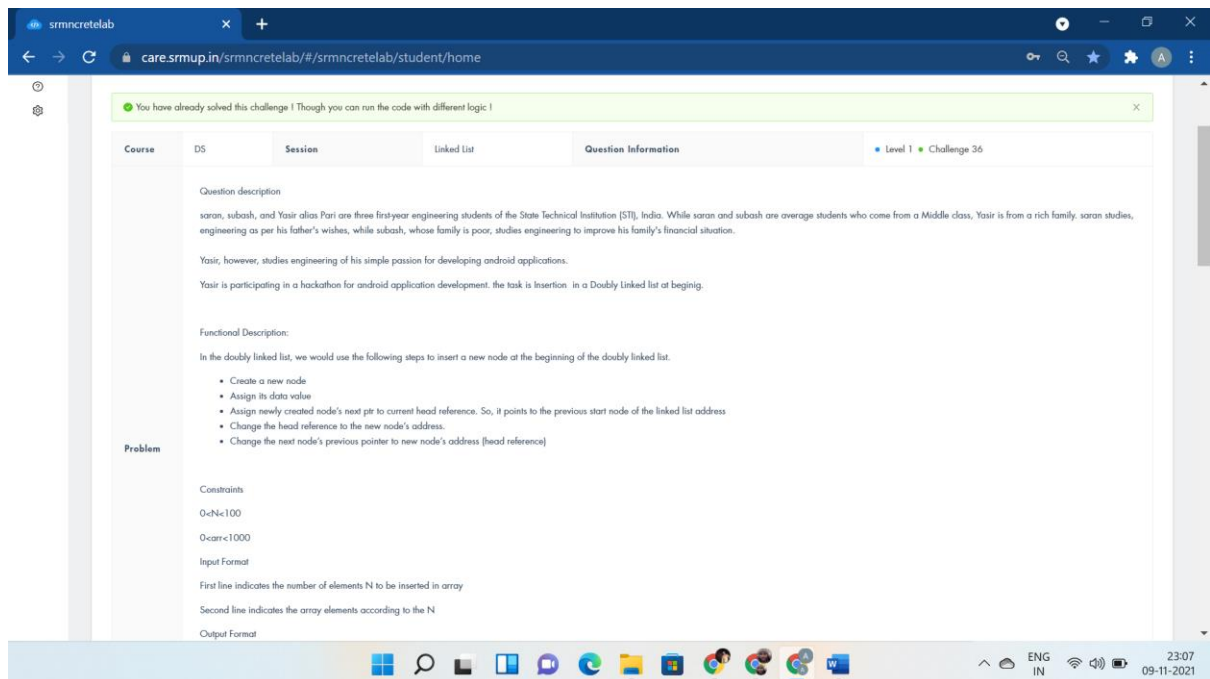
}

```

```

void display() {
    struct node *temp;
    temp = start;
    printf("Linked List:");
    while(temp != NULL)
    {
        printf("->%d",temp->data);
        temp = temp->next;
    }
}

```



```

#include <bits/stdc++.h>

using namespace std;

struct Node
{
    int data;
    struct Node *next;
    struct Node *prev;
};

```

```
void insertStart(struct Node** head,int data)
```

```
{
```

```
    struct Node* new_node = new Node();
```

```
    new_node->data = data;
```

```
    new_node->next = (*head);
```

```
    new_node->prev = NULL;
```

```
    if ((*head) != NULL)
```

```
        (*head)->prev = new_node;
```

```
    (*head) = new_node;
```

```
}
```

```
void printList(struct Node* node)
```

```
{
```

```
    Node* last;
```

```
    while (node != NULL)
```

```
    {
```

```
        cout<<node->data<<" ";
```

```
        last = node;
```

```
        node = node->next;
```

```
    }
```

```
    cout<<endl;
```

```
    while (last != NULL)
```

```
    {
```

```
        cout<<last->data<<" ";
```

```
        last = last->prev;
```

```
    }
```

```
}
```

```
int main()
```

```
{
```

```
    struct Node* head = NULL;
```

```
    int n;
```

```
    cin>>n;
```

```
    for(int i=0;i<n;i++){
```

```
        int t;
```

```

        cin>>t;

        insertStart(&head, t);

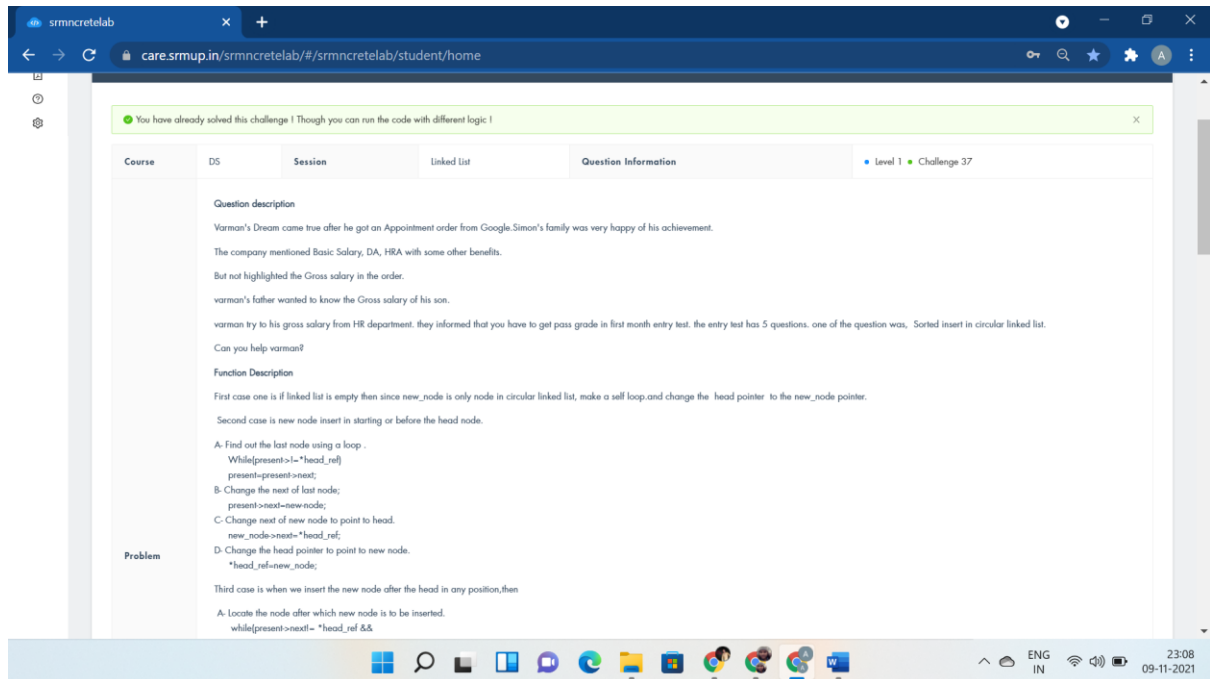
    }

    printList(head);

    return 0;

}

```



```

#include <stdio.h>

#include<stdlib.h>

struct Node{

    int data;

    struct Node *next;

};

void sortedInsert(struct Node** head_ref, struct Node* new_node)

{

    struct Node* current = *head_ref;

    if(current == NULL){

        new_node->next = new_node;
    }
}

```

```

        *head_ref=new_node;
    }

    else if(current->data >= new_node->data){
        while(current->next != *head_ref)
            current=current->next;

        current->next=new_node;
        new_node->next=*head_ref;
        *head_ref=new_node;
    }

    else{
        while(current->next != *head_ref && current->next->data < new_node->data)
            current = current->next;

        new_node->next = current->next;
        current->next=new_node;
    }
}

void printList(struct Node *start){
    struct Node *temp;
    temp=start;

    do{
        printf("%d ",temp->data);
        temp=temp->next;
    }while(temp->next != start);
    printf("%d",temp->data);
}

int main()
{

```

```

int n,i;

scanf("%d",&n);

struct Node *start=NULL;

struct Node *temp;

for(i=0; i<n; i++){

    temp=(struct Node*)malloc(sizeof(struct Node));

    scanf("%d",&temp->data);

    sortedInsert(&start, temp);

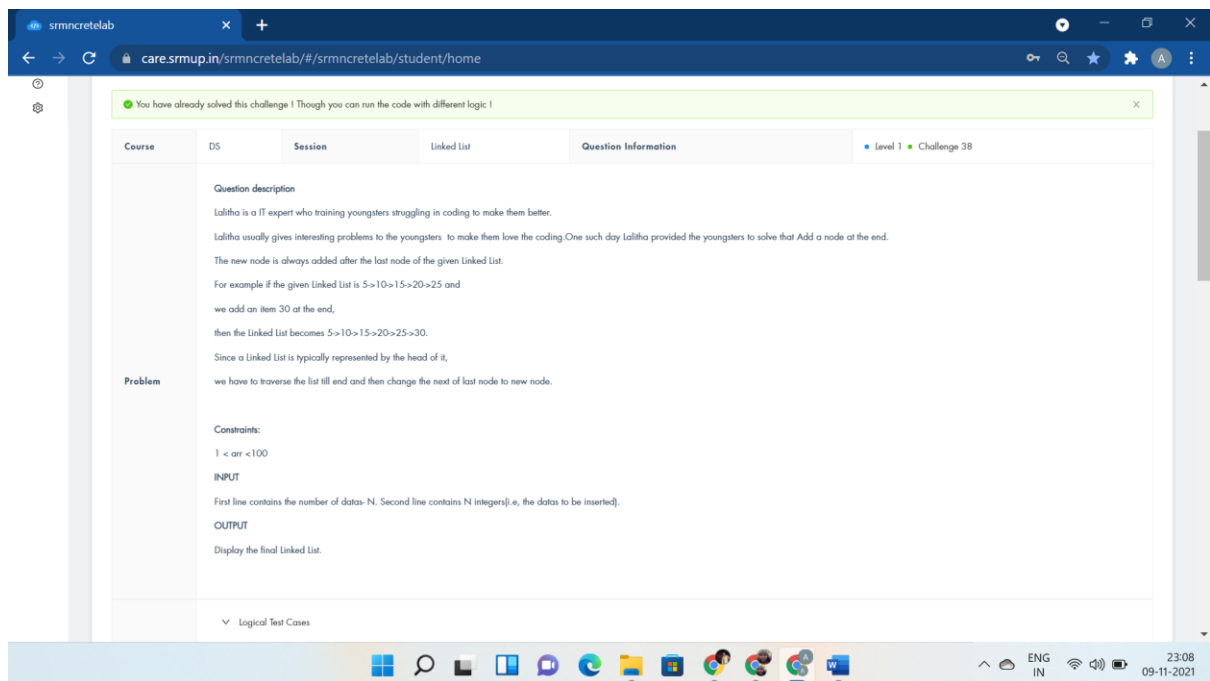
}

printList(start);

return 0;

}

```



```

#include <stdio.h>

#include<stdlib.h>

struct node{

    int data;

    struct node *next;

```

```

}*start;

void display();

int main() {

    int n;

    scanf("%d",&n);

    struct node *temp, *p2;

    start=NULL;

    while(n) {

        temp=(struct node *)malloc(sizeof(struct node));

        scanf("%d", &temp->data);

        temp->next = NULL;

        if(start == NULL){

            start= temp;

            p2 = temp;

        }

        else

        {

            p2->next=temp;

            // while(p2 != NULL && p2->next != NULL    p2=p2->next;

            p2=p2->next;

        }--n;

    }

    display();

    return 0;

}

void display() {

    struct node *temp;

    temp = start;

    printf("Linked List:");

    while(temp != NULL)

    {

        printf("->%d",temp->data);

        temp = temp->next;

    }

}

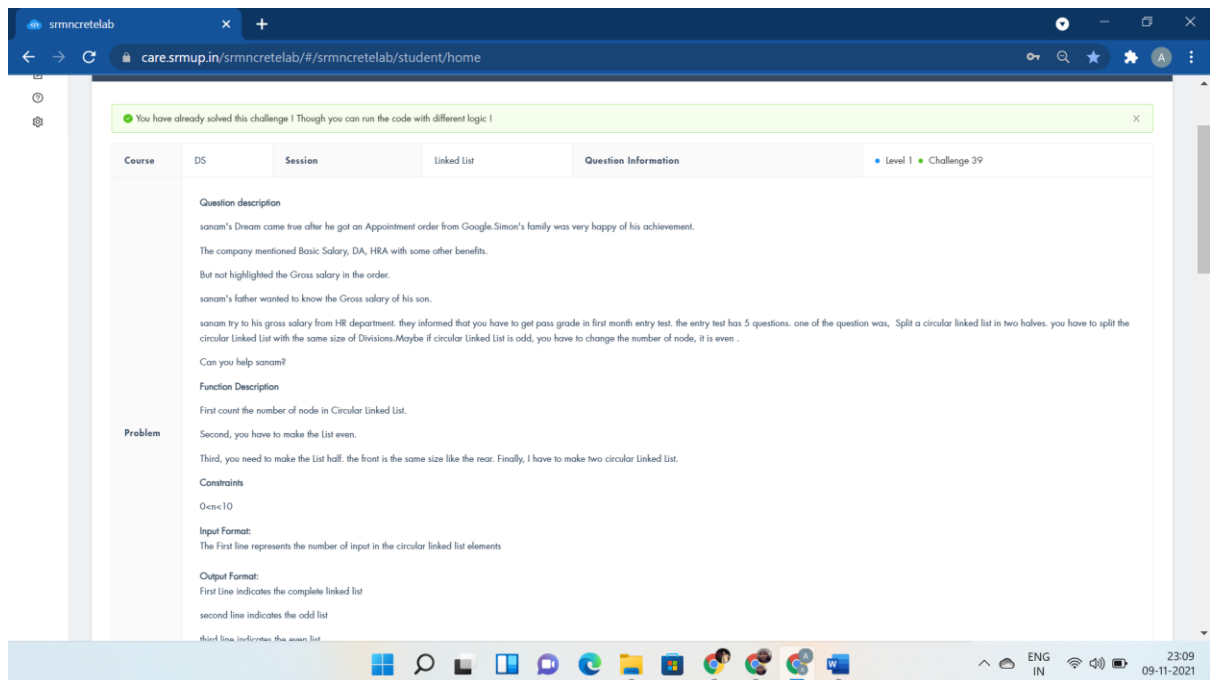
```

```

}

}

```



```

#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 even()
{
    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)

```

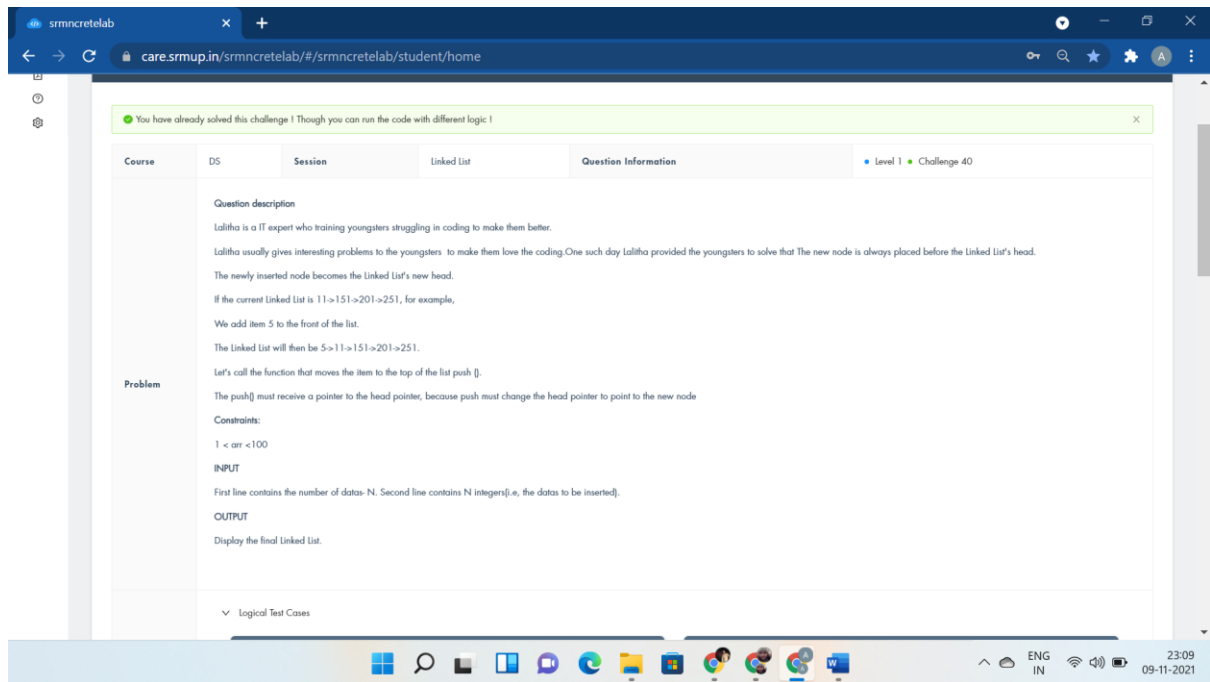
```

{
    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;
}

```



```
#include <bits/stdc++.h>

using namespace std;

struct node
{
    int data;
    node *next;
};

void push(node** start, int new_data){
    node* p1 = new node();
    p1->data = new_data;
    p1->next = *start;
    *start = p1;
}

void printList(node *node){
    while (node != NULL)
    {
        cout<<"->"<<node->data;
        node = node->next;
    }
}
```

```

int main(){

    node *start = NULL;

    int n,t;

    cin>>n;

    while(n--){

        cin>>t;

        push(&start,t);

    }

    cout<<"Linked List:";

    printList(start);

    return 0;

    cout<<"p1->next=start; void display()";

}

```

STACKS:-

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

Course	DS	Session	Stack	Question Information
				Level 1 Challenge 41

Question description

Hassan enjoys jumping from one building to the next. However, he merely jumps to the next higher building and stops when there are none accessible. The amount of stamina necessary for a voyage is equal to the xor of all the heights Hassan leaps till he comes to a halt.

If heights are [1 2 4], and he starts from 1, goes to 2 stamina required is $1 \oplus 2 = 3$, then from 2 to 3. Stamina for the entire journey is $1 \oplus 2 \oplus 4 = 7$. Find the maximum stamina required if can start his journey from any building.

Constraints

$1 \leq N \leq 10^5$
 $1 \leq \text{Height} \leq 10^9$

Input

First line: N , no of buildings.
 Second line: N integers, defining heights of buildings.

Output

Single Integer is the maximum stamina required for any journey.

Logical Test Cases

Test Case 1	Test Case 2
INPUT (STDIN) 5 1 2 3 8 6	INPUT (STDIN) 8 1 2 3 8 8 4 7 9

```
#include <stdio.h>
```

```
int main() {
```

```
    int i, j, arr[1000000], n, temp=0, st[1000000]= {0};
```

```
    scanf("%d",&n);
```