Name: Harsh Chauhan **UID: 21BCS10053**

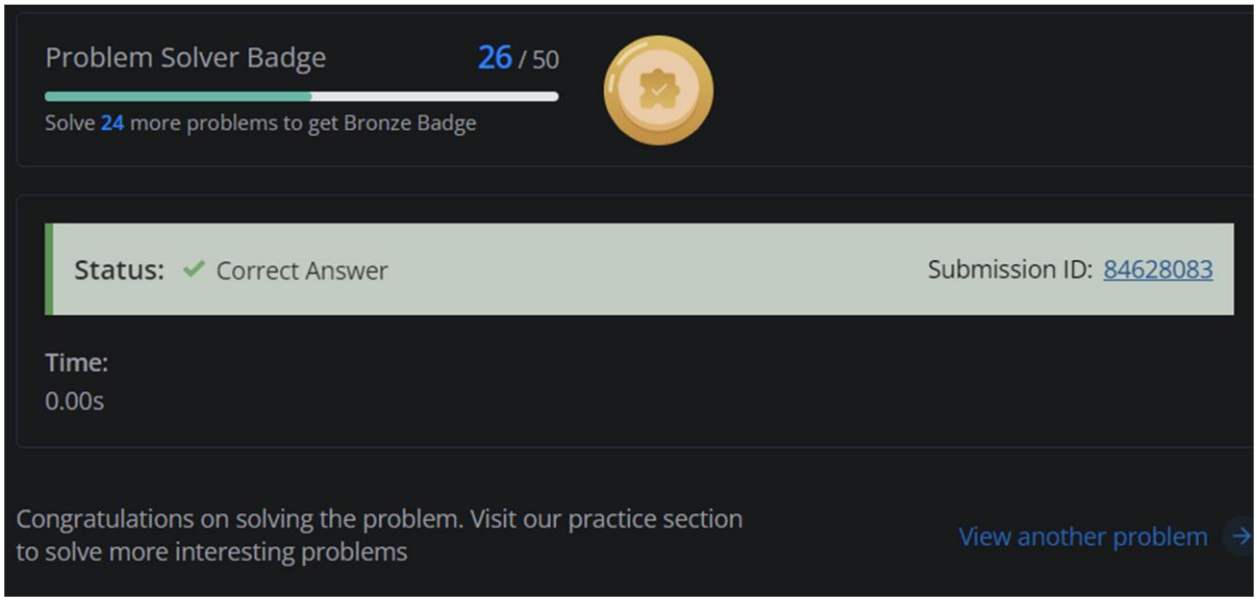
Q1. Temple land

Code:

```
#include <bits/stdc++.h>
using namespace std;
// INT HERE MEANS LONG LONG
#define int long long
#define endl '\n'

bool solve(int* arr, int n)
{
   if(arr[0]!=1 || arr[n-1]!=1)
return false;     if(n%2 == 0)
   return false;

   for(int i = 0; i<n/2; i++)
   {
      if((arr[i]-arr[i+1])!=-1)
return false;
   }
   for(int i = n/2; i<n-1; i++)
   {
      if((arr[i]-arr[i+1])!=1)
return false;
   }
   return true;
}
int32_t main(){
ios_base::sync_with_stdio(false);
cin.tie(NULL);
    int t;     cin >> t;
while (t--){         int n;
cin>>n;         int* arr =
new int[n];

       for(int i = 0; i<n; i++)
       {
          cin>>arr[i];
       }
```

```
        if(solve(arr,n))
        cout<< "yes"<<endl;

        else
        cout<<"no"<<endl;
    }
return 0; }
```

Output:

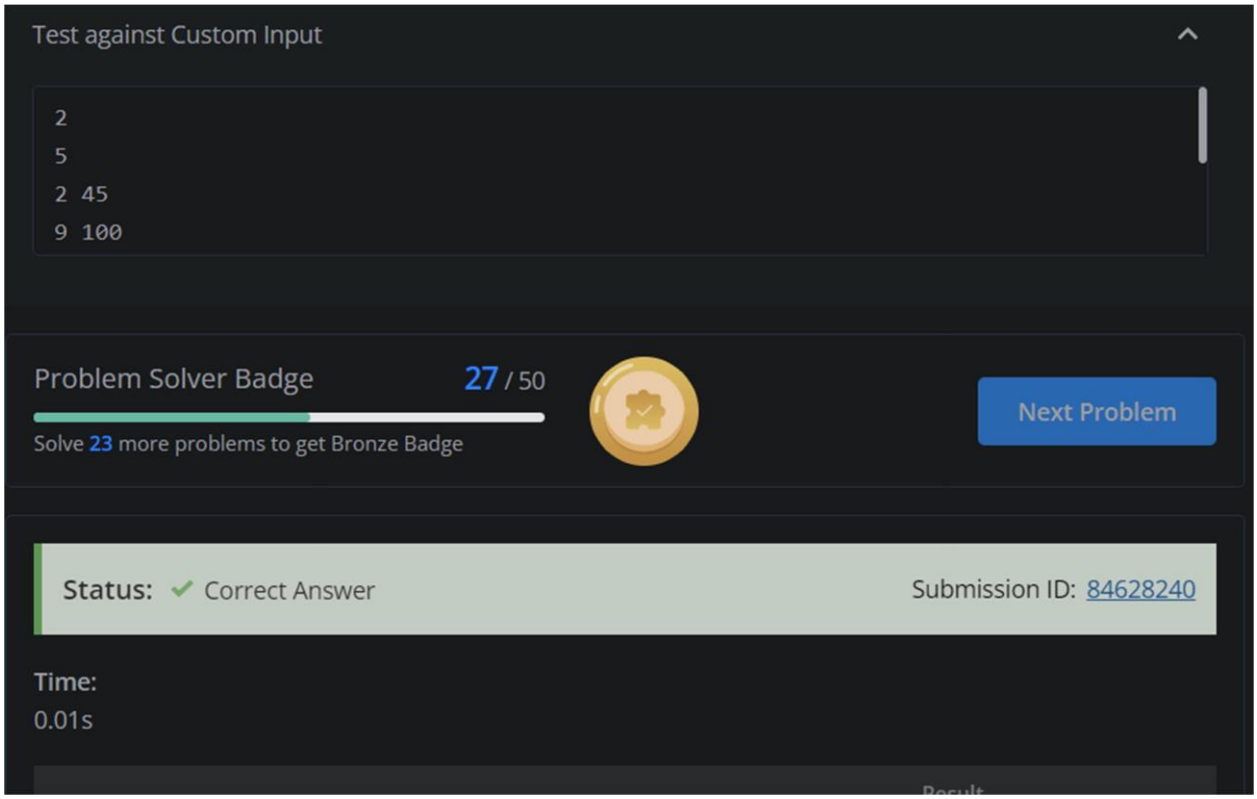


Q2. That Is my Score!

Code:

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

namespace std;


int main() {

map<int,int> m;

int n, x,y,sum=0,t;

cin>>t;


 while(t--)
```

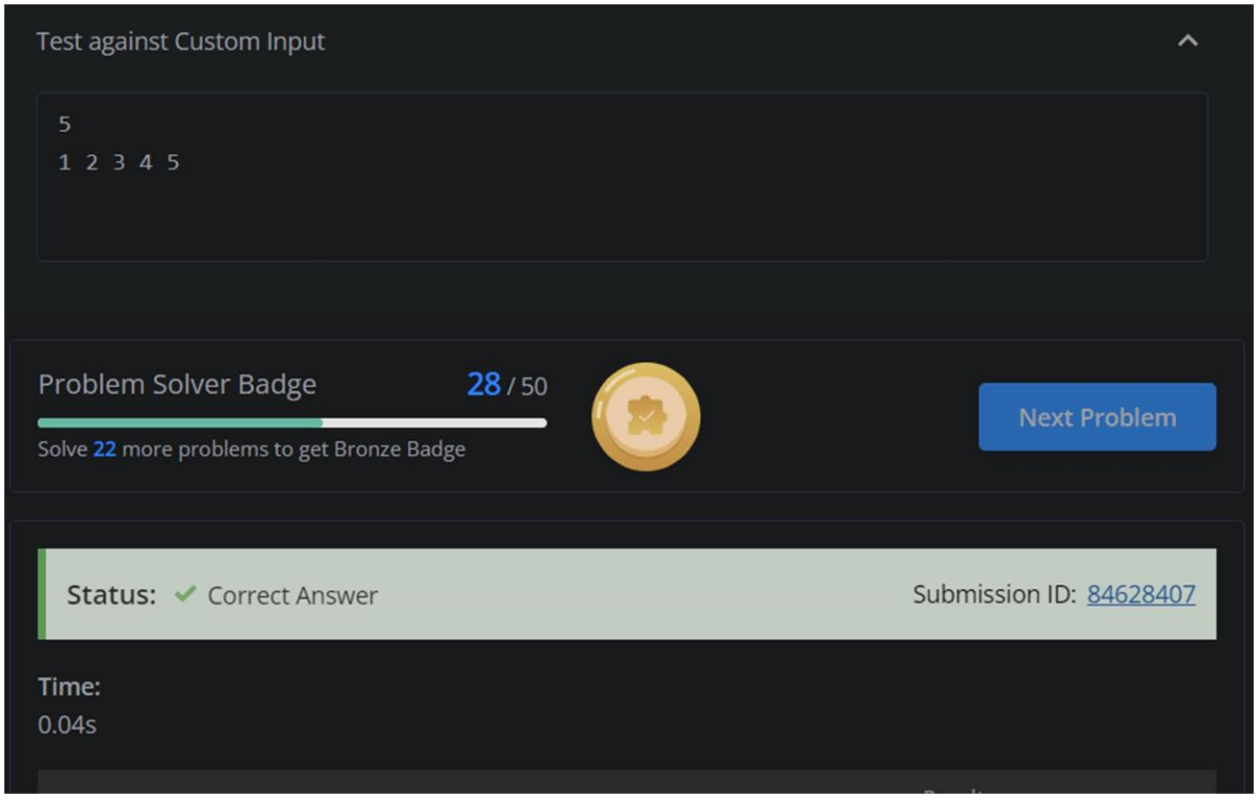
```
 {
cin>>n;

 for(int i=0; i<n; i++)
  {
    cin>>x>>y;

    if(x<=8)
    {

   m[x]=max(m[x],y);

    }   }   map<int,
int>::iterator it;

  for(it=m.begin();it!=m.end();it++)
  {
   sum=sum+(*it).second ;
  }

cout<<sum<<endl; sum=0;
m.clear();
} return 0
;
}
```



Q3. Pairwise AND Sum Code:

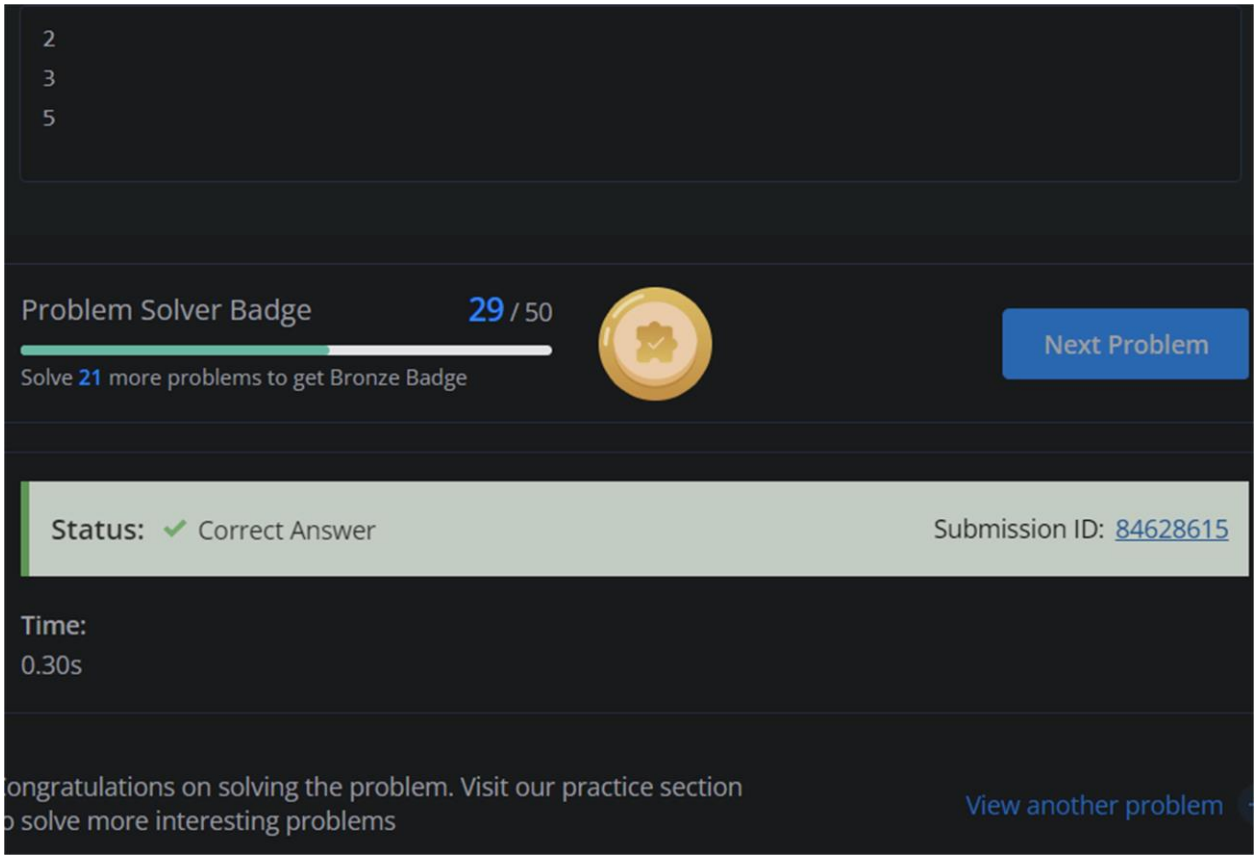
```
#include <bits/stdc++.h>
using namespace std;
#define ll long long int int
main() {
        // your code goes here
        ll n;
cin>>n;
vector<ll> v(n);
vector<ll> arr(31);
for(ll i=0;i<n;i++){
   cin>>v[i];
        }
```

```
        ll sum=0;
        for(int i=0;i<n;i++){
for(int j=0;j<31;j++){
if((v[i])&(1<<j)) arr[j]++;
            }
        }
        for(int j=0;j<31;j++){
   ll cnt=(arr[j]*(arr[j]-1))/2;
sum+=(cnt*(1<<j));
        }
        cout<<sum<<endl;
return 0;
}
```



Q4. Roads in Chef land Code:

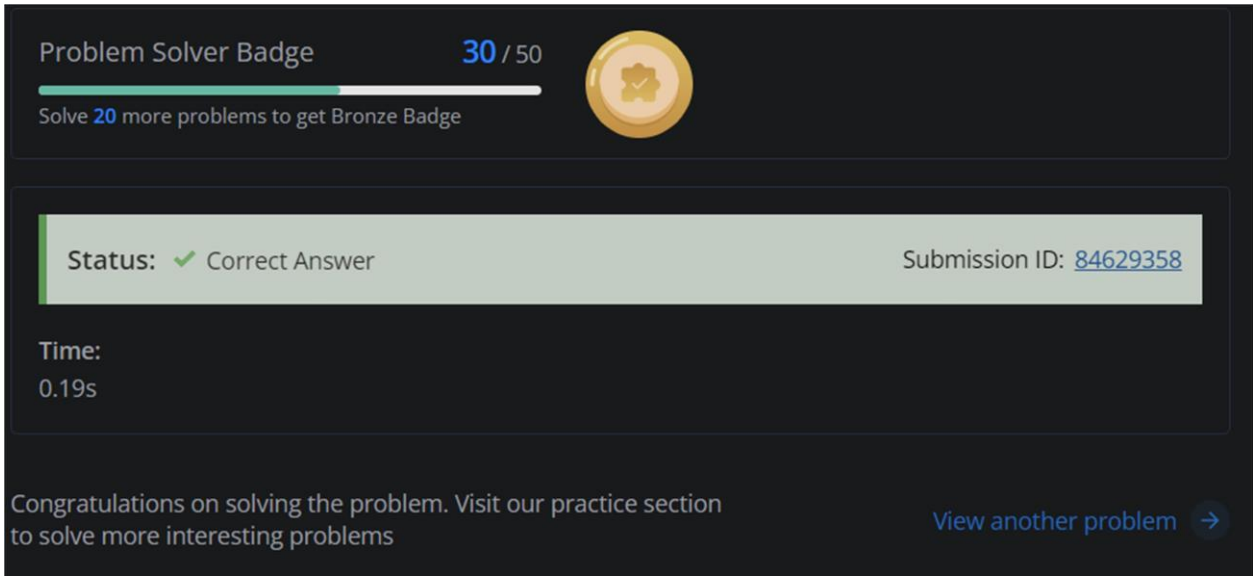
```
#include <bits/stdc++.h>
using namespace std; void
solve()
{     int n;     cin >> n;
if ((n & (n - 1)) == 0)
   {
          cout << -1 << endl;
       return;

   }
   long long ans = 0;     for (int i =
1; i <= n; i <<= 1)         ans += ((n -
i) / (i << 1)) * i;     for (int i = 2; i <
n; i <<= 1)         ans += i;     cout
<< ans << endl; } int main() {
ios_base::sync_with_stdio(false);
cin.tie(NULL);
   int t;
cin >> t;
while (t--)
{
solve();     }
return 0; }
```



Q5. Gold collection Code:

```
#include <iostream>
using namespace std;

int main() {     int t;
scanf("%d",&t);
        while(t--)
        {
                int n;
        scanf("%d",&n);

                        int arr[n];
                        for(int i=0;i<n;i++)
                        {
                                scanf("%d",&arr[i]);
                        }
                        for(int i=1;i<n;i++)
                        {
                                arr[i]+=arr[i-1];
                        }
```

```
                int q;
                scanf("%d",&q);
        for(int i=0;i<q;i++)
                {
                        int q1,q2;
                        scanf("%d%d",&q1,&q2);
                        if(q1==1)
                        {
                                printf ("%d\n"  ,arr[q2-1]);
                        }
                        else
                        {
                                printf("%d\n" ,  arr[q2-1]-arr[q1-2]);
                        }
                }
}
        return 0;
}
```

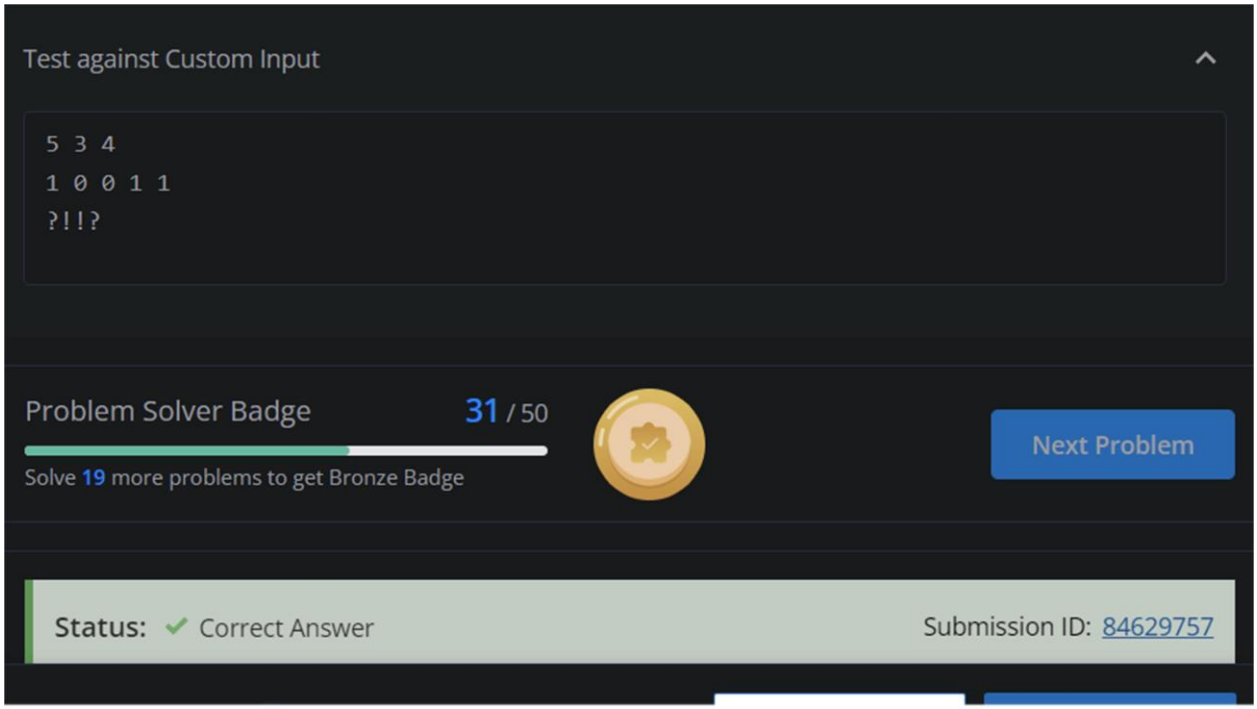


Q6. Chef and Sub Array

Code:

```
#include <bits/stdc++.h>
using namespace std;

int n, k, p, a[200001], sum[200001], maxi[524288];
```

```
string s;

void build(int node, int l, int r) {
if (l == r) {         maxi[node] =
sum[l];         return;
   }
   int p1 = 2 * node + 1, p2 = 2 * node + 2, c = (l + r) / 2;
build(p1, l, c);
   build(p2, c + 1, r);     maxi[node] =
max(maxi[p1], maxi[p2]);
}

int query(int node, int l, int r, int i, int j) {     if (l == i
&& r == j)         return maxi[node];     int p1 = 2 * node
+ 1, p2 = 2 * node + 2, c = (l + r) / 2;
   if (j <= c) return query(p1, l, c, i, j);     if (i > c) return
query(p2, c + 1, r, i, j);     return max(query(p1, l, c, i, c),
query(p2, c + 1, r, c + 1, j));
}

int main() {     ios::sync_with_stdio(0);
cin.tie(0); cout.tie(0);
   //freopen("test01.in", "r", stdin);
//freopen("test01.out", "w", stdout);
cin >> n >> k >> p;     k = min(k, n);
for (int i = 1; i <= n; i++) {
       cin >> a[i];
a[n + i] = a[i];
   }     cin >> s;     sum[0] = 0;     for (int i
= 1; i <= k; i++)         sum[i] = sum[i - 1]
+ a[i];     for (int i = k + 1; i <= 2 * n; i++)
sum[i] = sum[i - 1] + a[i] - a[i - k];
build(0, 1, 2 * n);     for (int i = 0, pos = 1;
i < s.size(); i++) {
       if (s[i] == '?')             cout << query(0, 1, 2 * n, pos + k - 1,
pos + n - 1) << "\n";         else {             pos--;             if (pos < 1)
pos = n;
       }     }
return 0; }
```



Q7. Multiple Linked Lists

```
#include <iostream> using
namespace std;
struct Node{
int data;
   Node* link;     Node*
head = NULL;     bool
circular = false;     int *table
= new int[1001];     Node*
NodeTable[1001];     int N =
0; public:     int* getTable()
          {
      return table;
   }
   Node()
   {
      for(int i = 0; i < 1001; i++)
                  table[i] = 0;
      for(int i = 0; i < 1001; i++)
```

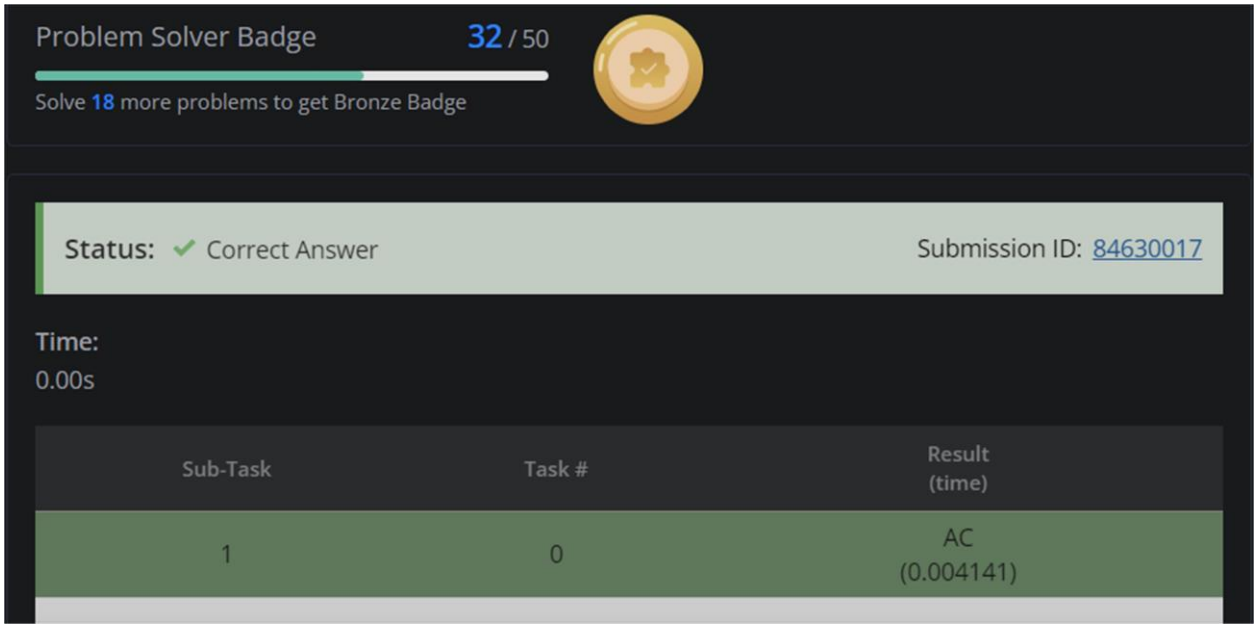
```
                    NodeTable[i] = NULL;
   }    void
first(int x){
      Node* temp1 = new Node;
temp1->data = x;       temp1-
>link = NULL;
NodeTable[x] = temp1;
N++;       if(head == NULL)
      {
         head = temp1;
table[head->data] = 0;
         return;
      }
      Node* temp = head;
while(temp->link != NULL)
temp = temp->link;
      temp->link = temp1;
   }

   void second(int y, int x)
   {
      N++;
      find(y) ? insertAfter(y, x) : insertBefore(y, x);
   }
   void insertBefore(int y, int x)
   {
      Node* temp = new Node;
temp->data = y;       temp-
>link = NULL;
NodeTable[y] = temp;
         if(head->data == x)
         {
         temp->link = head;
head = temp;
return;
      }
      Node* temp1 = head;
while(temp1->link->data != x)
temp1 = temp1->link;       temp->link =
temp1->link;
      temp1->link = temp;
   }
```

```
   void insertAfter(int y, int x){
Node* temp = new Node;
temp->data = x;         temp-
>link = NULL;
NodeTable[x] = temp;
Node* temp1 = head;
while(temp1->data != y)
temp1 = temp1->link;
temp->link = temp1->link;
       temp1->link = temp;
   }

   void third(int x, int y, int z)
   {
      Node* temp1 = new Node;
temp1->data = z;         temp1-
>link = NULL;
NodeTable[z] = temp1;         int
distance = 0;         Node* temp
= head;         while(temp-
>data!=x)            temp=temp-
>link;         Node* Initial =
temp;
       while(temp->data != y)
       {
          temp = temp->link;
          distance++;
       }         distance /=
2;         while(distance !=
0)
       {
          Initial = Initial->link;
distance--;
       }
       temp1->link = Initial->link;
       Initial->link = temp1;
       N++;
   }

   bool find(int x)
   {
      Node* temp = head;         while(temp != NULL)
          {
```

```
            if(temp->data == x)
return true;
            temp = temp->link;
        }
        if(temp == NULL)
return false;
    }

    void display()
    {
        Node* temp = head;
        while(temp != NULL)
            {
            cout << temp->data << " ";
            temp = temp->link;
        }
}
    void fourth(int x, int p)
    {         if(find(x) == false)
    return;         Node* temp =
NodeTable[x];
            while(p != 0)               {
if(temp->link == NULL)
                    {
                        temp->link = head;
                                temp = head;
                    circular = true;
                                p--
;                   }
else
                    {
                        temp = temp->link;
                        p--;
                    }
        }
                NodeTable[x]->link = temp;
    }     void
createtable()
            {
        for(int i = 1; i < 1001; i++)
if(NodeTable[i] != NULL)
if(NodeTable[i]->link != NULL)
table[NodeTable[i]->link->data]++;
```

```
   }
   void displaytable()
         {
      for(int i = 1; i < 10; i++)
         cout << i << " " << table[i] << endl;
   }    bool
Circular()
         {
      return circular;
   }    int
length()
         {
      return N;
   } }; int
main() {
Node A;
int N;    cin
>> N;
char c;
int b;
   for(int i = 0; i < N; i++)
         {
      cin>>c;       cin>>b;
if(c == 'I' && b == 0)
                  {
         int x;
cin >> x;
A.first(x);
      }
      else if(c == 'I' && b == 1)
                  {
         int x,y;
cin >> x >> y;
         A.second(x, y);
      }

      else if(c == 'I' && b == 2)
                  {
         int x, y, z;
cin >> x >> y >> z;
         A.third(x, y, z);
      }
      else if(c == 'U')
```

```
                    {
            int p;
cin >> p;
            A.fourth(b, p);
        }
    }
    A.createtable();     int*
ansTable = A.getTable();     int
countMultiple = 0;
    A.Circular() ? cout << 1 << endl : cout << 0 << endl;
    for(int i = 1; i <= 1000; i++)
if(ansTable[i] >= 2)
                        countMultiple++;
    if(countMultiple == 0)
            {
                cout << 0 << endl;
                A.display();
            }
    else
    {
        cout << countMultiple << endl;
for(int i = 1; i <= 1000; i++)
if(ansTable[i] >= 2)                cout
<< i << " ";         cout<<endl;
for(int i = 1; i <= 1000; i++)
if(ansTable[i] >= 2)                cout
<< ansTable[i] << " ";
    }
    cout << endl;
}
```


Problem Solver Badge
32 / 50
Solve 18 more problems to get Bronze Badge
Status: Correct Answer
Submission ID: 84630017
Time:
0.00s
Sub-Task
Task #
Result (time)
1
0
AC (0.004141)