LOVELY PROFESSIONAL UNIVERSITY

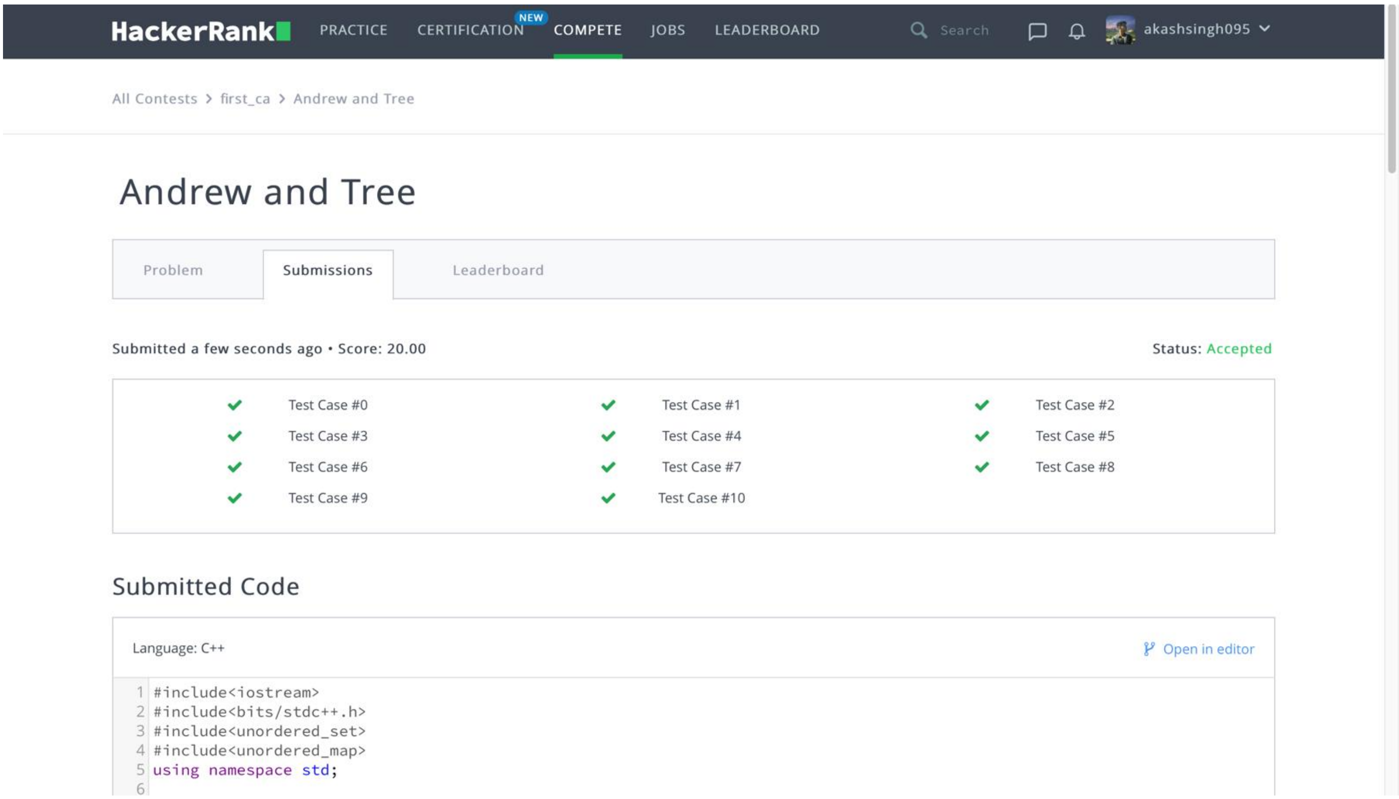
CSE331
Assignment 1

Name : Akash
Reg no:11805342

Problem 1:

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

#define ll                long long int
#define llmin               -1e18
#define ff                 first
#define ss                  second
#define pb                  push_back
#define vi                 vector<int>
#define take(a,n)           vector <int> a; for(int i=0;i<n;i++){int aa; cin>>aa; a.pb(aa);}
#define vll                vector<ll>
#define full(a)             (a.begin(),a.end())
#define vc                  vector<char>
#define iz(n)               int n; cin>>n;
#define iz2(n,m)              int n,m; cin>>n>>m;
#define mii                 map<int,int>
#define setbits(x)           __builtin_popcountll(x)
#define zerobits(x)           __builtin_ctzll(x)    //zeros before first 1
#define com                  int t; cin>>t; while(t--)
#define forn(n)              for(int i=0;i<n;i++)
#define fo(x,y)              for(int i=x;i<y;i++)
#define pq                  priority_queue <int, vector<int>, greater<int> >

struct Node{
   int data;
   Node *left;
   Node *right;
   Node(int x){
      this->data=x;
      this->left=NULL;
      this->right=NULL;
   }
};
```

```
Node* insertLevelOrder(int arr[], Node* root,
                       int i, int n)
{
    if (i < n)
    {
        Node* temp = new Node(arr[i]);
        root = temp;

        root->left = insertLevelOrder(arr,
                   root->left, 2 * i + 1, n);

        root->right = insertLevelOrder(arr,
                  root->right, 2 * i + 2, n);
    }
    return root;
}

int height(struct Node *root)
{
    if(root==NULL)
        return 0;

    int l = 1+height(root->left);

    int r=1+height(root->right);
    if(l>r)
        return l;
    else
        return r;

}

void spiral(vector<int> &vec, struct Node *root,int level ,int flag)
{
    if(root==NULL)
        return;
    if(level==1)
        vec.push_back(root->data);

    else if(level>1)
    {
        if(flag)
        {
            spiral(vec,root->left,level-1,flag);
            spiral(vec,root->right,level-1,flag);
        }
        else{
            spiral(vec,root->right,level-1,flag);
            spiral(vec, root->left,level-1,flag);
        }
    }
}

vector<int> findSpiral(Node *root)
{
    if(root==NULL){
        vector<int> vec;
        return vec;
    }

    int h = height(root);
    bool flag = true;

    vector<int> vec;
    for(int i=1;i<=h;i++)
    {
        spiral(vec,root,i,flag);
        flag=!flag;
    }
    return vec;
}


int main(){
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);


    // #ifndef ONLINE_JUDGE
    // freopen("input.txt", "r", stdin);
    // freopen("output.txt", "w", stdout);
    // #endif

    int n;
    cin>>n;

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

    }




    Node *root=insertLevelOrder(arr,root,0,n);

    vector <int> res=findSpiral(root);

    for (int i = 0; i < res.size(); i++)
    {
        cout<<res[i]<< " ";
    }





    return 0;
}
```



Andrew and Tree

Submitted a few seconds ago • Score: 20.00

Status: Accepted

✓	Test Case #0	✓	Test Case #1	✓	Test Case #2
✓	Test Case #3	✓	Test Case #4	✓	Test Case #5
✓	Test Case #6	✓	Test Case #7	✓	Test Case #8
✓	Test Case #9	✓	Test Case #10		

Submitted Code

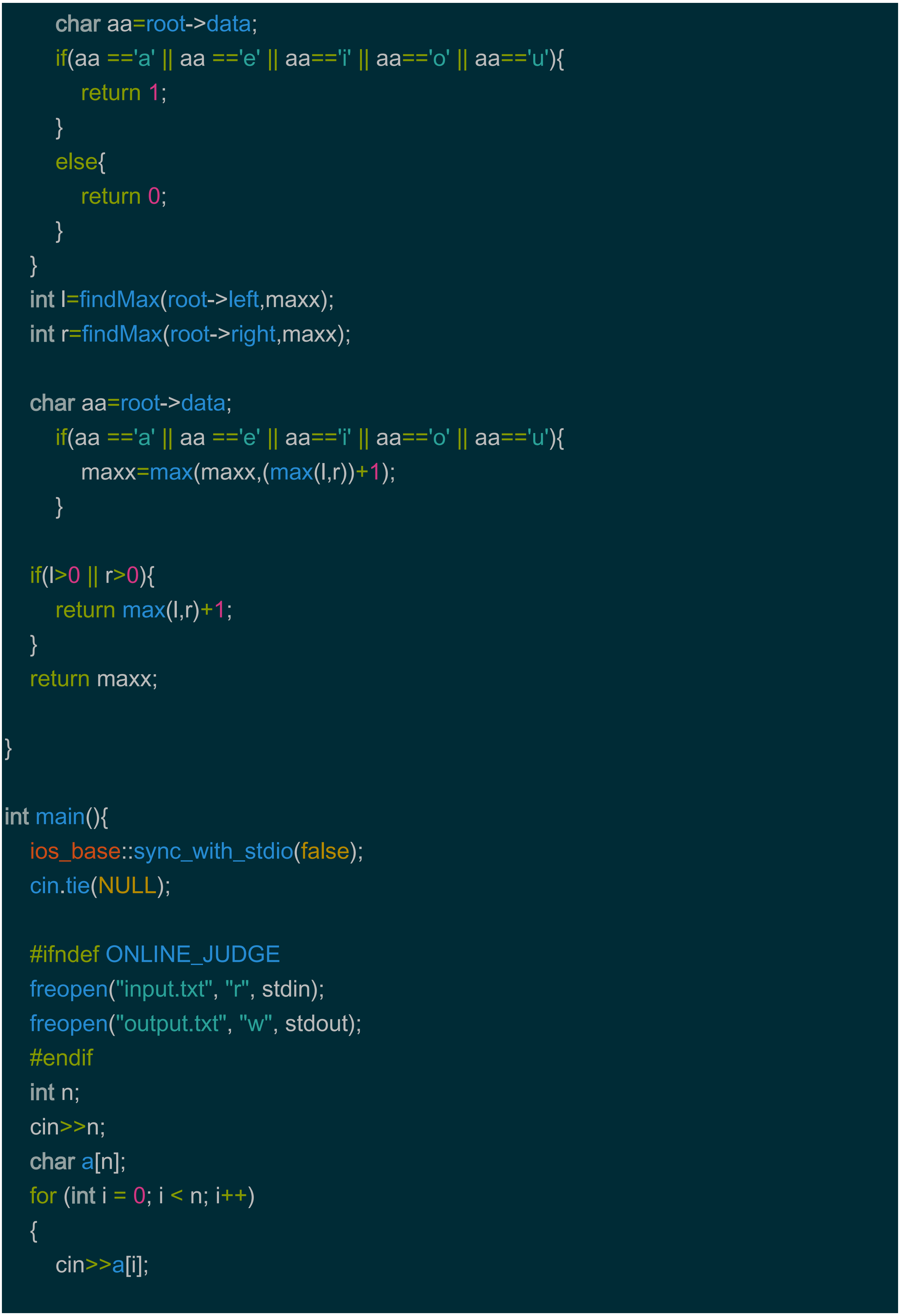
Language: C++

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

```

Problem 2:

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

#define ll                  long long int
#define llmin                 -1e18
#define ff                  first
#define ss                    second
#define pb                    push_back
#define vi                   vector<int>
#define take(a,n)                vector <int> a; for(int i=0;i<n;i++){int aa; cin>>aa; a.pb(aa);}
#define vll                  vector<ll>
#define full(a)               (a.begin(),a.end())
#define vc                    vector<char>
#define iz(n)                 int n; cin>>n;
#define iz2(n,m)                int n,m; cin>>n>>m;
#define mii                   map<int,int>
#define setbits(x)               __builtin_popcountll(x)
#define zerobits(x)               __builtin_ctzll(x)     //zeros before first 1
#define com                     int t; cin>>t; while(t--)
#define forn(n)                 for(int i=0;i<n;i++)
#define fo(x,y)                for(int i=x;i<y;i++)
```

```
#define pq                  priority_queue <int, vector<int>, greater<int> >

struct Node{
    char data;
    Node *left;
    Node *right;
    Node(int x){
        this->data=x;
        this->left=NULL;
        this->right=NULL;
    }
};

Node* insertLevelOrder(char arr[], Node* root,
                       int i, int n)
{
    if (i < n)
    {
        if(arr[i]=='#'){
            return NULL;
        }
        Node* temp = new Node(arr[i]);
        root = temp;

        root->left = insertLevelOrder(arr,
                   root->left, 2 * i + 1, n);

        root->right = insertLevelOrder(arr,
                  root->right, 2 * i + 2, n);
    }
    return root;
}

int findMax(Node *root,int &maxx){
    if(!root){
        return 0;
    }
    if(root->left==NULL && root->right==NULL){
```



```
        char aa=root->data;
        if(aa =='a' || aa =='e' || aa=='i' || aa=='o' || aa=='u'){
            return 1;
        }
        else{
            return 0;
        }
    }
    int l=findMax(root->left,maxx);
    int r=findMax(root->right,maxx);

    char aa=root->data;
        if(aa =='a' || aa =='e' || aa=='i' || aa=='o' || aa=='u'){
            maxx=max(maxx,(max(l,r))+1);
        }

    if(l>0 || r>0){
        return max(l,r)+1;
    }
    return maxx;

}

int main(){
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);

    #ifndef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    #endif
    int n;
    cin>>n;
    char a[n];
    for (int i = 0; i < n; i++)
    {
        cin>>a[i];
    }

    Node *root=insertLevelOrder(a,root,0,n);

    int maxx;
    cout<<findMax(root,maxx);

    return 0;
}
```