CHANDIGARH UNIVERSITY
UNIVERSITY INSTITUTE OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Submitted By: Satyam
Submitted To: Navneet Chaudhry

Subject Name	Competitive Coding
Subject Code	21 CSP-314
Branch	BE-CSE
Semester	5th

LAB INDEX

NAME: Satyam
SUBJECT NAME: Competitive Coding Lab
UID: 20BCS9393
SUBJECT CODE: 21CSP-314
SECTION: 607-A

Sr. No	Program	Date	Evaluation				Sign
			LW (12)	VV (8)	FW (10)	Total (30)	
1.	**ARRAYS:** https://www.hackerrank.com/challenges/30-arrays/problem https://www.hackerrank.com/challenges/simple-arraysum/problem?isFullScreen=true https://www.hackerrank.com/challenges/compare-thetriplets/problem?isFullScreen=true https://www.hackerrank.com/challenges/diagonaldifference/problem?isFullScreen=true	04-Aug-2022					
2.	**STACK & QUEUES:** https://www.hackerrank.com/challenges/equalstacks/problem?isFullScreen=true https://www.hackerrank.com/challenges/game-of-twostacks/problem?isFullScreen=true https://www.hackerrank.com/challenges/balancedbrackets/problem?isFullScreen=true https://www.hackerrank.com/challenges/down-to-zeroii/problem?isFullScreen=true https://www.hackerrank.com/challenges/trucktour/problem?isFullScreen=true	18-Aug-2022					

3.	**Linked List:** https://www.hackerrank.com/challenges/compare-two-linkedlists/problem?isFullScreen=true https://www.hackerrank.com/challenges/insert-a-node-into-asorted-doubly-linked-list/problem?isFullScreen=true	25-Aug-2022					
4.	**Searching and Sorting:** https://www.hackerrank.com/challenges/missingnumbers/problem?isFullScreen=true https://www.hackerrank.com/challenges/closestnumbers/problem?isFullScreen=true	01-Sep-2022					
5.	**Graph:** https://www.hackerrank.com/challenges/bfsshortreach/problem?isFullScreen=true https://www.hackerrank.com/challenges/the-quickest-wayup/problem?isFullScreen=true https://www.hackerrank.com/challenges/eventree/problem?isFullScreen=true https://www.hackerrank.com/challenges/journey-to-themoon/problem?isFullScreen=true https://www.hackerrank.com/challenges/frog-inmaze/problem?isFullScreen=true	29-Sep-2022					
6.	**Trees:** https://www.hackerrank.com/challenges/tree-topview/problem?isFullScreen=true https://www.hackerrank.com/challenges/binary-search-treeinsertion/problem?isFullScreen=true https://www.hackerrank.com/challenges/swap-nodesalgo/problem?isFullScreen=true https://www.hackerrank.com/challenges/tree-huffmandecoding/problem?isFullScreen=true https://www.hackerrank.com/challenges/balancedforest/problem?isFullScreen=true	13-Oct-2022					

EXPERIMENT-2.2(a)

1. Aim/Overview of the practical:

To demonstrate the concept of Trees.

2. Task to be done/ Which logistics used:

https://www.hackerrank.com/challenges/tree-top-view/problem?isFullScreen=true

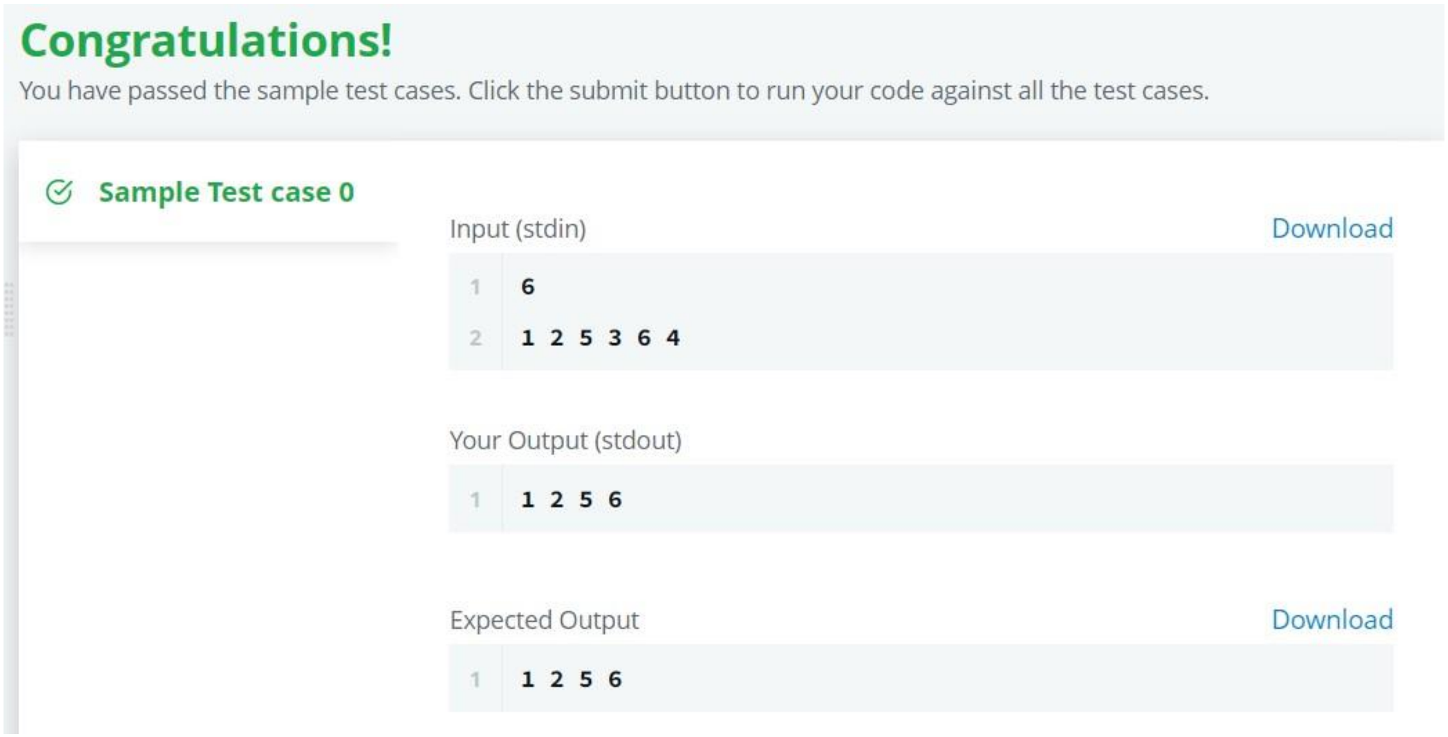
3. Steps for experiment/practical/Code:

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

using namespace std;
 class Node
{
public:
        int data;
Node *left;
        Node *right;
Node(int d) {
data = d;              left
= NULL;              right
= NULL;
        }
};  class
Solution {
public:
        Node* insert(Node* root, int data) {
if(root == NULL) {                 return
new Node(data);
```

```
            } else {                  Node* cur;
if(data <= root->data) {                      cur =
insert(root->left, data);                      root-
>left = cur;
                } else {
                    cur = insert(root->right, data);
root->right = cur;
               }

               return root;
           }
        } /*
class Node {
public:
        int data;
Node *left;
        Node *right;
Node(int d) {
data = d;              left
= NULL;              right
= NULL;
        }
};
*/

    void topView(Node * root) {
queue<pair<int,Node*>> q; q.push(make_pair(0,root));
map<int,Node*> ans;          for(auto
i=q.front();!q.empty();q.pop(),i=q.front()){
if(!i.second) continue;              ans.insert(i);
            q.push(make_pair(i.first+1,i.second->right));
            q.push(make_pair(i.first-1,i.second->left));
}
        for(auto i:ans) cout<<i.second->data<<" ";
    }



}; //End of Solution
 int main()
{
```

```
Solution
myTree;
    Node* root = NULL;
         int
t;      int
data;

    std::cin >> t;

    while(t-- > 0) {          std::cin >>
data;          root = myTree.insert(root,
data);
    }
    myTree.topView(root);
return 0;
}
```

4. Result/Output/Writing Summary:



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin)

```
6
1 2 5 3 6 4
```

Your Output (stdout)

```
1 2 5 6
```

Expected Output

```
1 2 5 6
```

EXPERIMENT-2.2(b)

1.Aim/Overview of the practical:

To demonstrate the concept of Trees.

2. Task to be done/ Which logistics used:

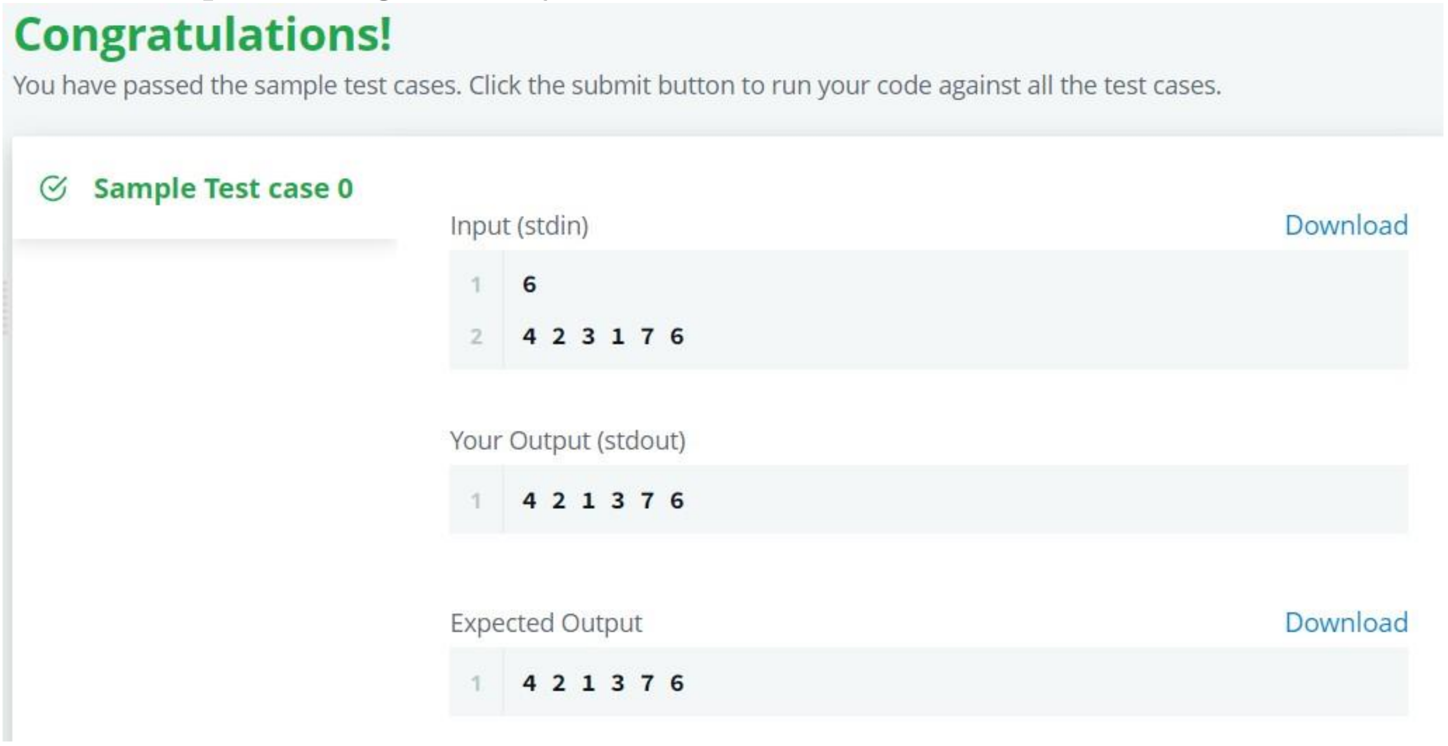
https://www.hackerrank.com/challenges/binary-search-tree-insertion/problem?isFullScreen=true

3. Steps for experiment/practical/Code:

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

using namespace std;
 class Node
{
public:
        int data;
Node *left;
        Node *right;
Node(int d) {
data = d;               left
= NULL;               right
= NULL;
        }
};
class Solution {
public:
    void preOrder(Node *root) {

        if( root == NULL )
return;

        std::cout << root->data << " ";
        preOrder(root->left);
preOrder(root->right);
    }

/*
Node is defined as
 class Node {
public:         int
data;         Node
*left;
        Node *right;
Node(int d) {
data = d;              left
= NULL;              right
= NULL;
        }
};

*/

Node * insert(Node * root, int value) {

if(root==NULL) {
Node* newNode;
    newNode = (Node*)malloc(sizeof(Node));
newNode->left = NULL;     newNode->right =
NULL;     newNode->data = value;
return newNode;
  }
  if(value <= root->data)     root->left =
insert(root->left, value);   else     root-
>right = insert(root->right, value);
     return
root;
}
};

int main() {

    Solution myTree;
    Node* root = NULL;
```

```
        int
t;     int
data;

    std::cin >> t;

    while(t-- > 0) {         std::cin >>
data;          root = myTree.insert(root,
data);
    }
    myTree.preOrder(root);

    return 0;
}
```

4. Result/Output/Writing Summary:


Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin)

```
6
4 2 3 1 7 6
```

Your Output (stdout)

```
4 2 1 3 7 6
```

Expected Output

```
4 2 1 3 7 6
```

EXPERIMENT-2.2(c)

1.Aim/Overview of the practical:

To demonstrate the concept of Trees.

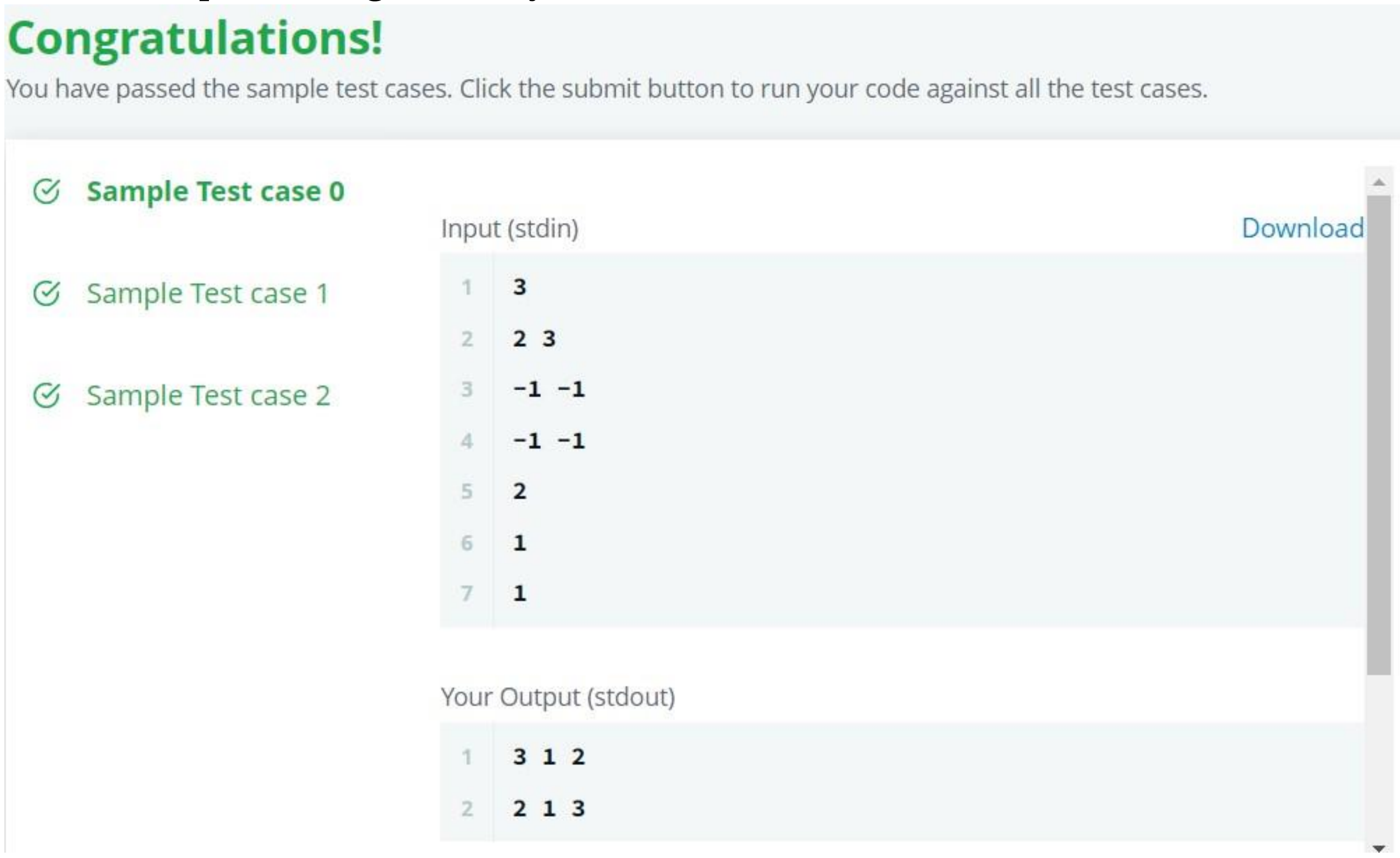
2. Task to be done/ Which logistics used:

https://www.hackerrank.com/challenges/swap-nodes-algo/problem?isFullScreen=true

3. Steps for experiment/practical/Code:

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream> #include
<algorithm> using namespace std;
vector<int> leftNode, rightNode;
int swapLevel;

void traverse(int node=1){
if (node == -1) return;
traverse(leftNode[node]);
cout << node << " ";
traverse(rightNode[node]);
if (node == 1) cout << endl;
}
void swap(int level=1, int node=1) {
if (node == -1) return;     if (level %
swapLevel == 0) {         int tmp =
leftNode[node];         leftNode[node] =
rightNode[node];         rightNode[node]
= tmp;
    }
    swap(level+1, leftNode[node]);
swap(level+1, rightNode[node]);
}  int
main() {
int count;
cin>>count;
leftNode.pus
h_back(0);
rightNode.pu
sh_back(0);
while(count-
-){
int L, R;
cin>>L>>R;
leftNode.pus
h_back(L);
rightNode.pu
sh_back(R);
    }     cin>>count;
while(count--){
cin >> swapLevel;
swap();
traverse();
    }
return 0;
}
```

4. Result/Output/Writing Summary:



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Sample Test case 1

Sample Test case 2

Input (stdin)

```
3
2 3
-1 -1
-1 -1
2
1
1
```

Your Output (stdout)

```
3 1 2
2 1 3
```

EXPERIMENT-2.2(d)

1.Aim/Overview of the practical:

To demonstrate the concept of Trees.

2. Task to be done/ Which logistics used:

https://www.hackerrank.com/challenges/tree-huffman-decoding/problem?isFullScreen=true

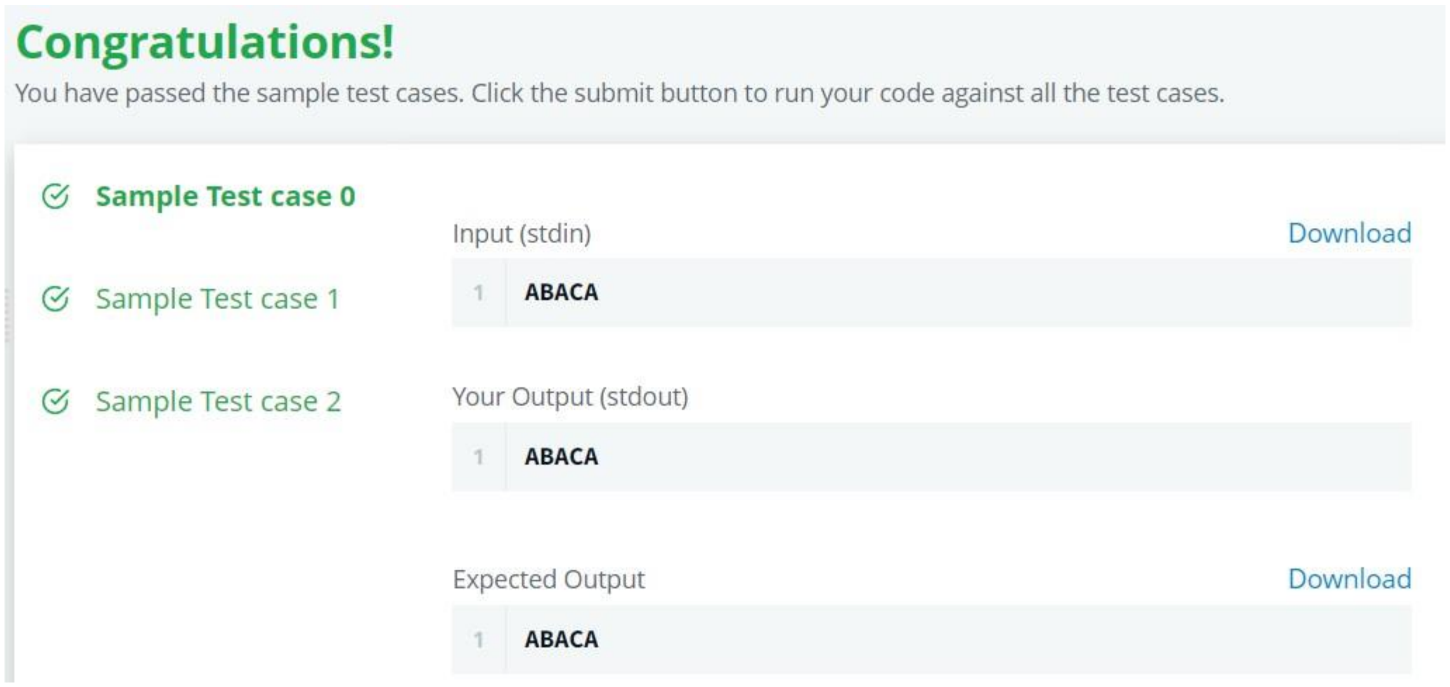
3. Steps for experiment/practical/Code:

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

typedef struct node {
int freq;     char
data;     node *
left;     node *
right;
} node;
struct deref:public binary_function<node*, node*, bool> {
bool operator()(const node * a, const node * b)const {
return a->freq > b->freq;
    }
};
typedef priority_queue<node *, vector<node*>, deref> spq;

node * huffman_hidden(string s) {
     spq pq;
vector<int>count(256,0);

    for(int i = 0; i < s.length(); i++ ) {
count[s[i]]++;
    }

    for(int i=0; i < 256; i++) {

        node * n_node = new node;
n_node->left = NULL;         n_node-
>right = NULL;         n_node->data =
(char)i;         n_node->freq = count[i];
```

```
        if( count[i] != 0 )
pq.push(n_node);

}
    while( pq.size() != 1 ) {

        node * left = pq.top();
pq.pop();         node * right = pq.top();
pq.pop();         node * comb = new node;
comb->freq = left->freq + right->freq;
comb->data = '\0';         comb->left = left;
comb->right = right;         pq.push(comb);
    }
    return pq.top();
}
void print_codes_hidden(node * root, string code, map<char, string>&mp) {

    if(root == NULL)
return;

    if(root->data != '\0') {
mp[root->data] = code;
    }
    print_codes_hidden(root->left, code+'0', mp);     print_codes_hidden(root-
>right, code+'1', mp);

}

/*
The structure of the node is

typedef struct node {
     int freq;
char data;
node * left;
node * right;

} node;

*/
```

```
void decode_huff(node * root,string s)
{     string ans = "";     node* n = root;
for(auto itr = s.begin(); itr != s.end();itr++){
node* next;          if(*itr == '0'){
next = n -> left;
        }
else{
            next = n -> right;
        }
        if(next -> data == '\0'){
n = next;         }         else{
ans += next -> data;
n = root;
        }
}
    cout << ans << endl;
}

int main() {
string s;
std::cin >>
s;

    node * tree = huffman_hidden(s);
string code = "";     map<char,
string>mp;

    print_codes_hidden(tree, code, mp);

    string coded;

    for( int i = 0; i < s.length(); i++ ) {
coded += mp[s[i]];
    }
    decode_huff(tree,coded);

    return 0;
}
```

4. Result/Output/Writing Summary:



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Sample Test case 1

Sample Test case 2

Input (stdin) Download

```
ABACA
```

Your Output (stdout)

```
ABACA
```

Expected Output Download

```
ABACA
```

EXPERIMENT-2.2(e)

1.Aim/Overview of the practical:

To demonstrate the concept of Trees.

2. Task to be done/ Which logistics used:

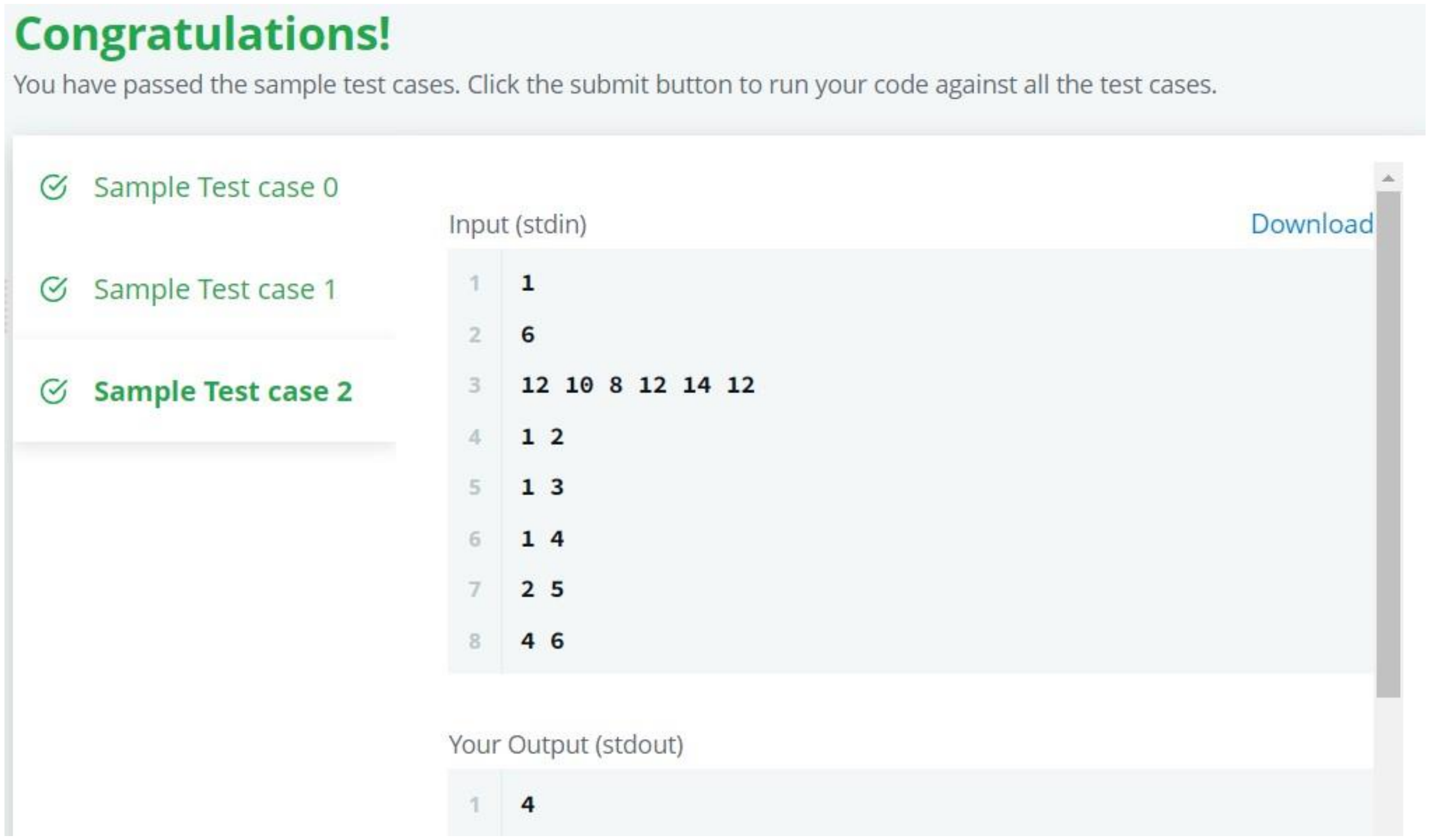
https://www.hackerrank.com/challenges/balanced-forest/problem?isFullScreen=true

3. Steps for experiment/practical/Code:

```
#include <iostream>
#include <cstdio>
#include <vector>
#include <algorithm>
#include <string>
#include <set>
```

```
#include <map>
#include <queue>
#include <stack>
#include <deque>
#include <cassert>
#include <stdlib.h>

using namespace std; typedef
long long ll; const ll INF =
(ll) 1e18; const int N =
(int) 5e4 + 10;

vector<int> g[N];
ll c[N]; ll f[N];
ll res = INF; ll
tot = 0; bool
was[N];
void upd(ll a, ll b, ll c) {
if (a == b && c <= a)
res = min(res, a - c);     if
(a == c && b <= a)          res
= min(res, a - b);     if (b ==
c && a <= b)          res =
min(res, b - a);
}
set<ll>* unite(set<ll>* a, set<ll>* b) {     if (a->size()
> b->size())         swap(a, b);     for (ll x : *a) {
if (b->count(tot - 2 * x))             upd(tot - 2 * x, x,
x);         if (b->count(x))             upd(x, x, tot - 2
* x);         if ((tot - x) % 2 == 0 && b->count((tot - x)
/ 2))             upd((tot - x) / 2, x, (tot - x) / 2);
    }     for (ll x :
*a) {         b-
>insert(x);
    }
delete a;
return b;
}
set<ll>* dfs(int v) {     was[v] =
true;     f[v] = c[v];     set<ll>*
sv = new set<ll>();     for (int to
: g[v])         if (!was[to]) {
set<ll>* sto = dfs(to);
f[v] += f[to];             sv =
unite(sv, sto);
        }
    if (f[v] % 2 == 0 && sv->count(f[v] / 2))
upd(f[v] / 2, f[v] / 2, tot - f[v]);     if (sv-
>count(tot - f[v]))         upd(tot - f[v], 2 * f[v]
- tot, tot - f[v]);     if (sv->count(2 * f[v] -
tot))         upd(2 * f[v] - tot, tot - f[v], tot -
f[v]);     sv->insert(f[v]);     return sv;
}  void solve() {     int n;
cin >> n;     for (int i = 0; i <
N; i++) {         was[i] = false;
g[i].clear();         c[i] = 0;
}     tot = 0;     res = INF;
for (int i = 0; i < n; i++) {
cin >> c[i];         tot += c[i];
}
    for (int i = 0; i < n - 1; i++) {
int x, y;         cin >> x >> y;
        --x;         --y;
g[x].push_back(y);
g[y].push_back(x);
    }     set<ll>* s = dfs(0);
//for (int i = 0; i < n; i++)
    //    cerr << f[i] << " ";
//cerr << endl;     delete s;
if (res == INF)         res =
-1;     cout << res << endl;
    // cerr << "----------" << endl;
}  int main() {
ios_base::sync_with_stdio(0);
int p;     cin >> p;
    while (p--) {
solve();
    }
return 0;
}
```

4. Result/Output/Writing Summary:



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
1
6
12 10 8 12 14 12
1 2
1 3
1 4
2 5
4 6
```

Your Output (stdout)

```
4
```

Learning outcomes (What I have learnt):

1. Through this experiment I learn concepts of Trees.

2. Different operations on Trees.

3.Learned about different algorithms of Trees.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			