Laborator 6
Arbori oarecare
Determinarea arborelui parțial de cost minim
Algoritmul lui Kruskal
Algoritmul lui Prim
Arbori Binari

Probleme rezolvate

1. Să se creeze un arbore binar cu informaţii numere întregi, apoi să se tipărească suma elementelor pare din arbore.

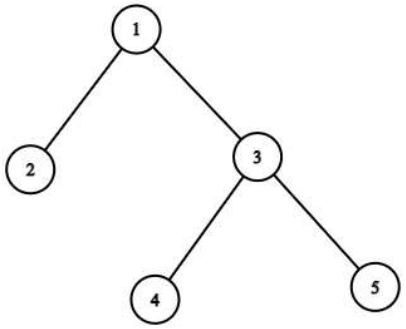
Rezolvare:

Codul sursa este urmatorul:

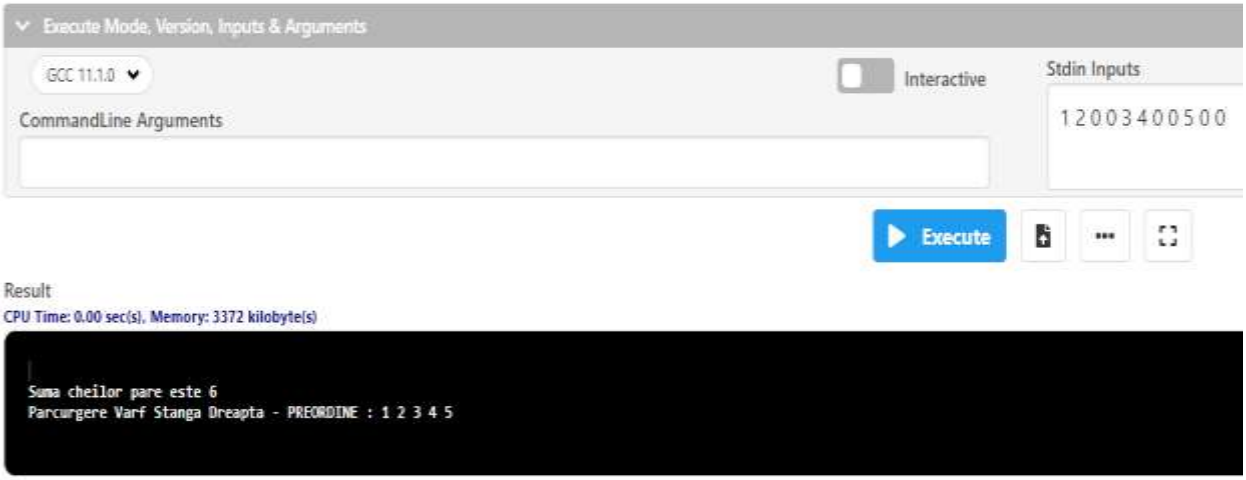
```
#include <iostream>
using namespace std;
typedef struct arbore
{
int inf;
struct arbore *st,*dr;
}arbore;
arbore *rad;
arbore *creare(void)
{
int n;
arbore *c;
cin>>n;
if(n != 0){
        c = new arbore;
        c -> inf = n;
        c -> st = creare();
        c -> dr = creare();
        return c;
}
else return(NULL);
}
void PREORDINE(arbore *c)
{
if(c != NULL){
        cout<<c -> inf<<" ";
        PREORDINE(c -> st);
        PREORDINE(c -> dr);
```

```
}
   return;
}

int suma(arbore *p)
{
if(p == NULL) return 0;
else     if(p -> inf % 2 == 0) return (p -> inf + suma(p -> st) + suma(p -> dr));
          else return (suma(p -> st) + suma(p -> dr));
}
int main(void)
{
rad = creare();
cout<<"\nSuma cheilor pare este "<<suma(rad);
cout<<"\nParcurgere Varf Stanga Dreapta - PREORDINE : ";
PREORDINE(rad);
          return 0;
}
```

Se considera urmatorul arbore cu radacina 1, iar valorile care se introduc sunt:
1 2 0 0 3 4 0 0 5 0 0



Executia programului folosind compilatorul online https://www.jdoodle.com/online-compiler-c++/:



2. Sa se construiasca un arbore binar si apoi sa se caute o valoare data, specificandu-se daca este sau in arbore.

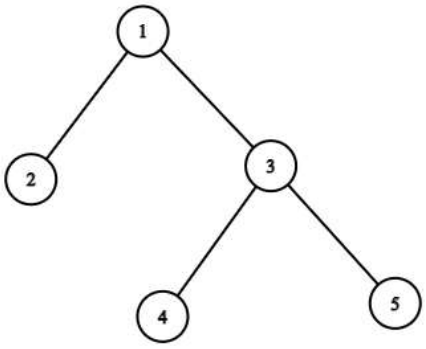
Rezolvare:

Codul sursa este urmatorul:

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

typedef struct arbore
        {
                int inf;
                struct arbore *st,*dr;
        }arbore;
arbore *rad,*q;
int val;

arbore *creare(void)
{
int n;
arbore *c;
//cout<<" Dati n = ";
cin>>n;
if(n!=0)
{
        c=new arbore;
        c->inf=n;
        c->st=creare();
        c->dr=creare();
        return c;
}
else return(NULL);
}
void INORDINE(arbore *c)
{
if(c!=NULL)
{
        INORDINE(c->st);
        cout<<c->inf<<" ";
        INORDINE(c->dr);
}
return;
}
arbore *cautare_recursiva(arbore *p,int val)
{
if( (p==NULL) || (p->inf==val) ) return p;
else
  if(val < p->inf)  return (cautare_recursiva(p->st,val));
```

```
  else return (cautare_recursiva(p->dr,val));
}
int main(void)
{
rad = creare();
cout<<"\nParcurgere INORDINE (stanga-varf-dreapta) ";
INORDINE(rad);
cout<<"\nDati informatia pe care o cautati ";
   cin>>val;cout<<val;
q = cautare_recursiva(rad,val);
if(q == NULL) cout<<"\nValoarea "<<val<<" NU exista in arbore";
              cout<<"\nEXISTA!!";
   return 0;
}
```

Se considera urmatorul arbore cu radacina 1, iar valorile care se introduc sunt:
1 2 0 0 3 4 0 0 5 0 0



Executia programului folosind compilatorul online https://www.jdoodle.com/online-compiler-c++/:

3. Se citește un arbore oarecare prin vectorul de tati. Sa se determine si sa se afișeze cel mai lung lanț din arbore.

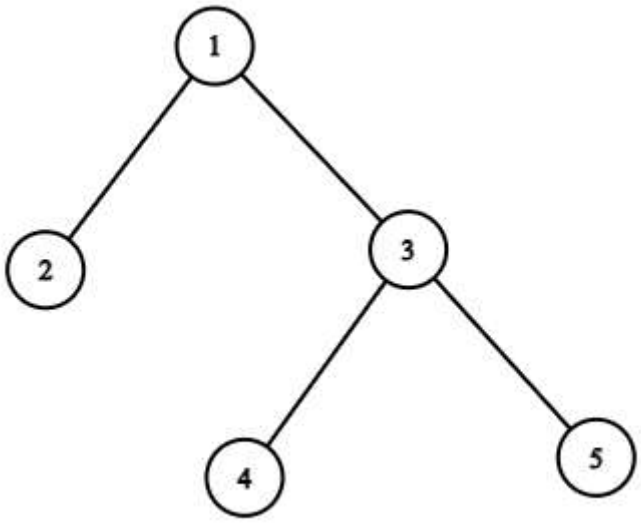
Rezolvare:

Codul sursa este urmatorul:

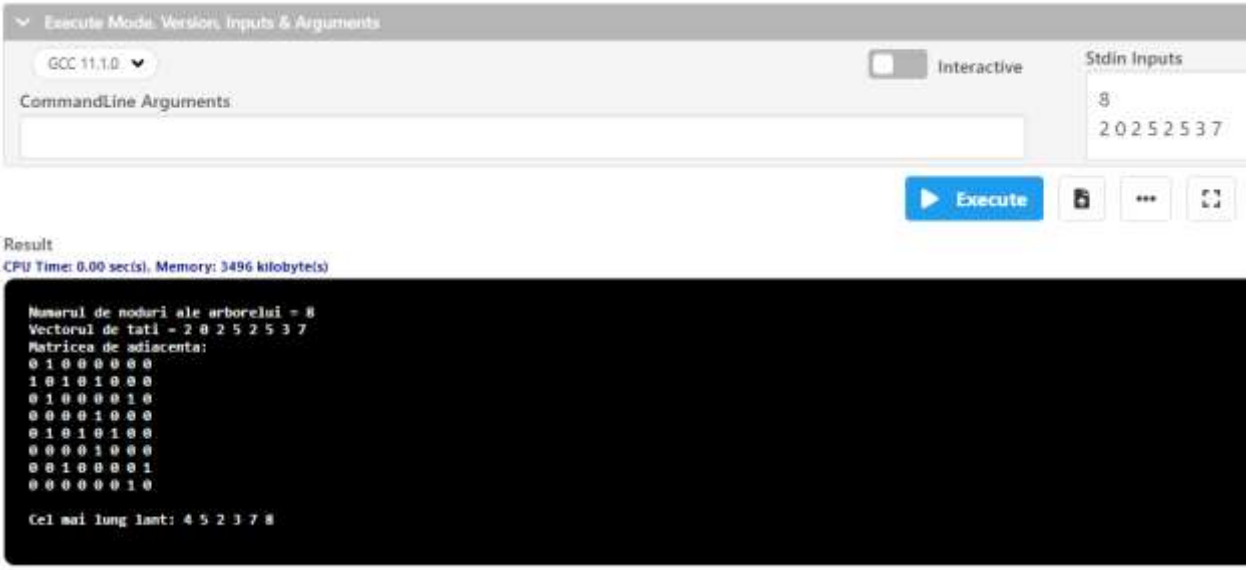
```
#include <iostream>
using namespace std;

int a[20][20],t[100],n,maxim,mx,my,p[100],f[100];
void citire()
{
   int i;
   cout<<"Numarul de noduri ale arborelui = ";
   cin>>n;cout<<n;
   for(i=1;i<=n;i++)
   {
      cin>>t[i];
      f[t[i]]=1;
      a[i][t[i]]=1;
      a[t[i]][i]=1;
   }
}

void df( int r, int niv,int k)
{
   p[k]=1;
   if( niv > maxim)
   {
      maxim = niv;
   mx = r;
   my = k;
}
   for(int i = 1; i <= n; i++)
    if( !p[i] && a[k][i])  df(r, niv + 1, i);
}

void ff(int r)
{
   if(t[r]) ff(t[r]);
   t[t[r]]=r;
}

void lant(int r)
{
```

```
    if(t[r]) lant(t[r]);
    cout<<r<<" ";
}

int main()
{
    citire();
    cout<<"\nVectorul de tati = ";
    for(int i = 1; i <= n; i++) cout<<t[i]<<" ";
    cout<<"\nMatricea de adiacenta:\n";
    for(int i = 1; i <= n; i++)
    {
        for(int j = 1; j <= n; j++) cout<<a[i][j]<<" ";
        cout<<'\n';
    }
    for(int i=1;i<=n;i++)
        if(f[i]==0)
    {
        for(int j=1;j<=n;j++)   p[j] = 0;
        df(i,0,i);
              }
    ff(mx);
    t[mx]=0;
    cout<<"\nCel mai lung lant: ";
    lant(my);
    return 0;
}
```

Exemplu:
Pentru vectorul de tati: 2 0 2 5 2 5 3 7
se afișează lanțul: 4 5 2 3 7 8



Executia programului folosind compilatorul online https://www.jdoodle.com/online-compiler-c++/:



4. Se citește un număr natural n. Construiți un arbore cu proprietatea ca fiecare varf are numărul de descendenti directi cu 1 mai mare decat nivelul pe care se afla. Excepție fac frunzele și nodul pentru care se termina cele n varfuri.
Astfel, rădăcina (aflată pe nivelul 0) are un singur descendent direct, varful de pe nivelul 1 are 2, cele de pe nivelul 2 au cate trei, etc.
Arborele va fi reprezentat prin vectorul legăturilor de tip tata.

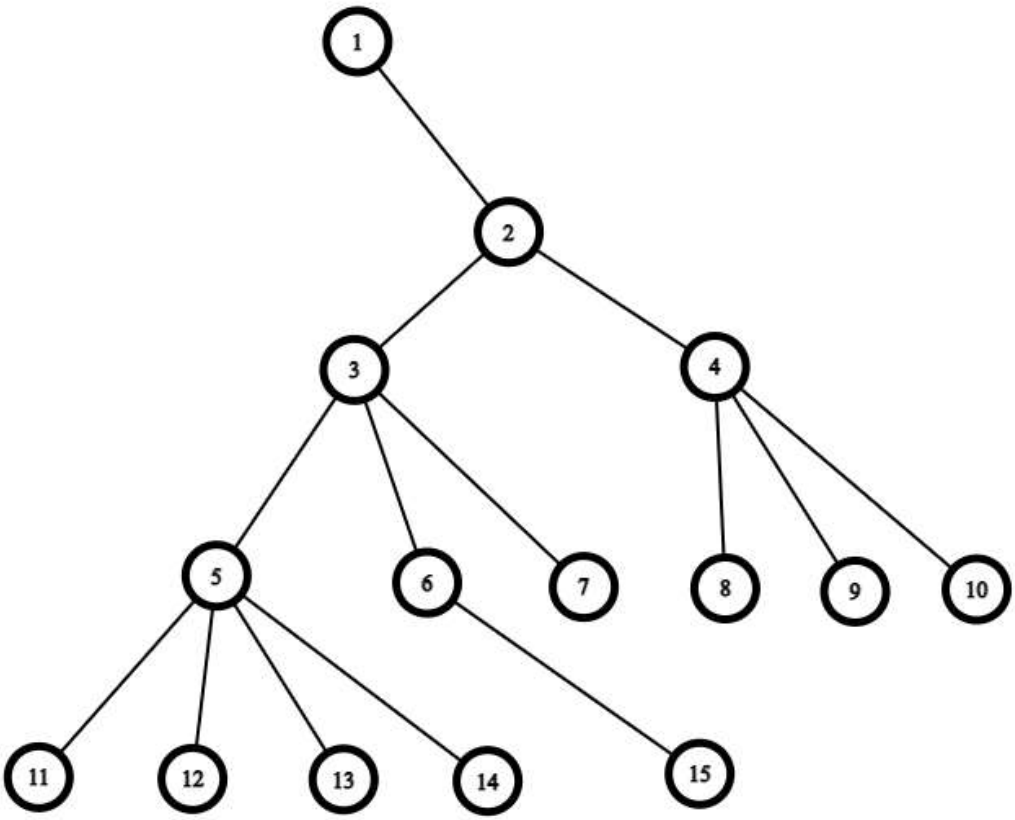
Rezolvare:

Codul sursa este următorul:

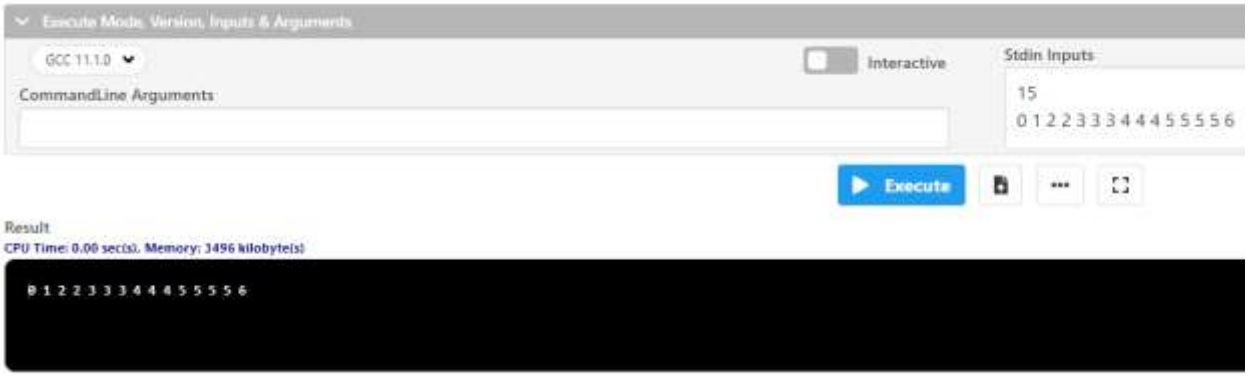
```
#include<fstream.h>
const int inf=15000;
int t[100],n;
ifstream f("date.in");
ofstream g("date.out");

int main()
 {
  int v,k,niv,p,i;
  f>>n;
  t[1]=0;
  p=1;
  niv=1;
  k=2;
```

```
  v=1;
  while(k<=n)
  {
      p=p*niv;
      for(i=1;i<=p && k<=n;i++)
      { t[k++]=v;
        if(i%niv==0) v++;
      }
      niv++;
  }
  for(i=1;i<=n;i++) g<<t[i]<<" ";
  f.close();
  g.close();
}
```

Exemplu:
n=15
Vectorul tata:
0 1 2 2 3 3 3 4 4 4 5 5 5 5 6



Executia programului folosind compilatorul online https://www.jdoodle.com/online-compiler-c++/:



5. Se dau doi arbori cu rădăcină prin doi vectori de tip tata. Determinati daca cei doi arbori cu radacina reprezinta acelasi arbore (difera doar in urma alegerii rădăcinilor diferite)

Rezolvare:
Codul sursa este următorul:

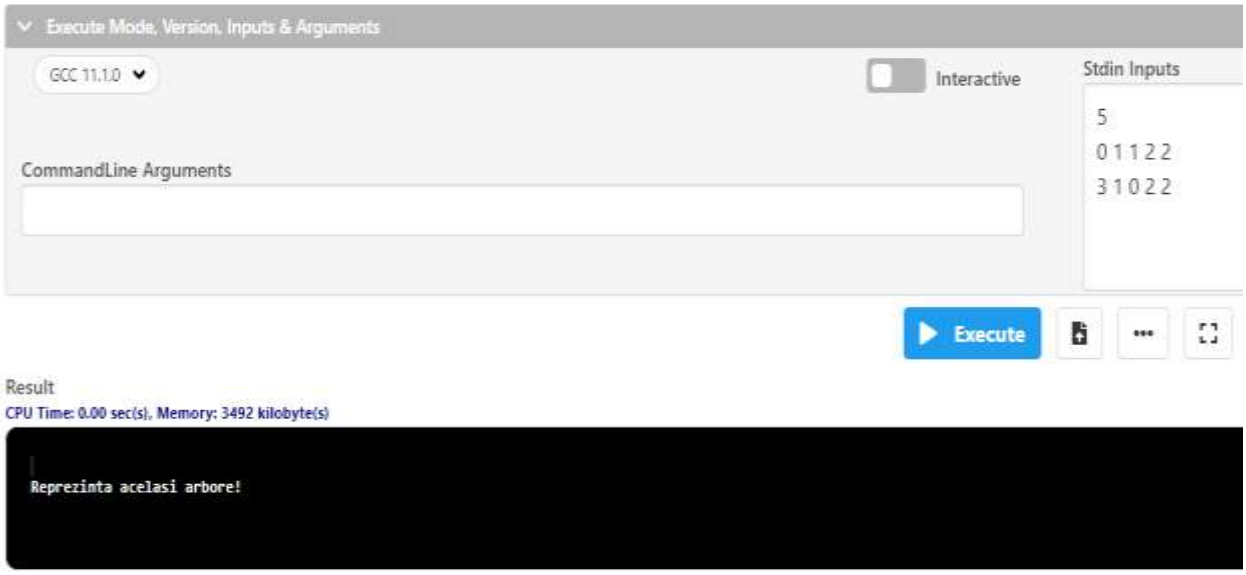
```
#include <iostream>
using namespace std;

int n,t[100],s[100];
void citire()
{
   cin>>n;
   for(int i=1;i<=n;i++)   cin>>t[i];
   for(int i=1;i<=n;i++)   cin>>s[i];
}

int main()
{
   citire();
   int ok=1,gasit;
   for(int i=1;i<=n;i++)
      if(t[i])
      {
         gasit=0;
         for(int j=1;j<=n;j++)
            if(i==j && t[i]==s[j] || i==s[j] && t[i]==j)
               gasit=1;
         if(!gasit) ok=0;
```

```
    }
  if(ok) cout<<"\nReprezinta acelasi arbore!";
  else cout<<"\nNU reprezinta acelasi arbore!";
  return 0;
}
```

Exemplu:
5
0 1 1 2 2
3 1 0 2 2

Reprezinta acelasi arbore!

Executia programului folosind compilatorul online https://www.jdoodle.com/online-compiler-c++/:



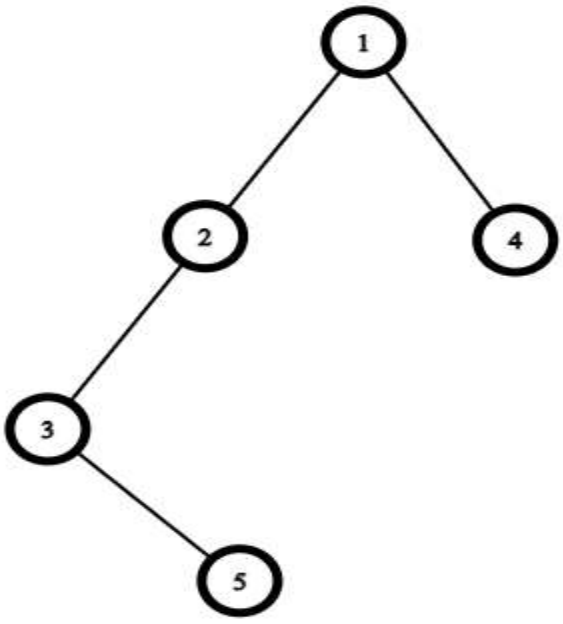
6. Se citește un arbore cu n varfuri dat prin vectorul muchiilor și apoi se citește varful radacina. Sa se calculeze și sa se afișeze numărul de niveluri ale arborelui.

Rezolvare:
Codul sursa este urmatorul:

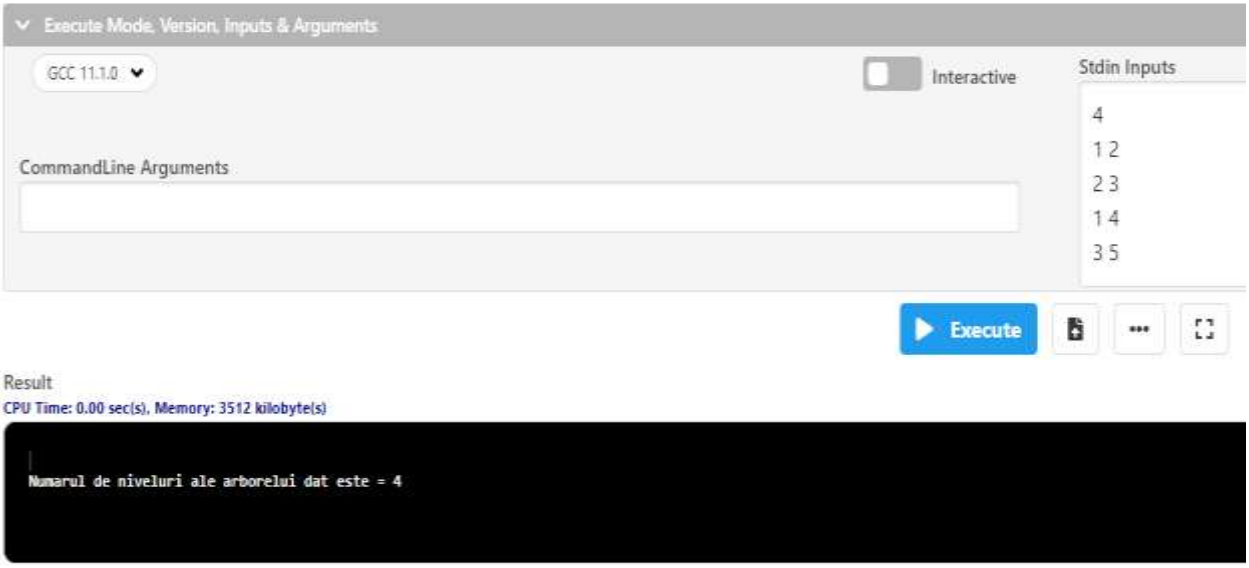
```
#include <iostream>
using namespace std;

int n, maxim, r, a[100][100], p[100];

void citire()
```

```
{
   int i,x,y;
   cin>>n;     // n = numarul de muchii ale arborelui
   for(i=1;i<=n-1;i++)
   {
      cin>>x>>y;
      a[x][y] = a[y][x] = 1;
   }
   cin>>r;
}

void DF(int r, int niv)
{
   p[r] = 1;
   if(niv > maxim) maxim = niv;
   for(int i=1;i<=n;i++)
      if(a[r][i] && !p[i])
         DF(i, niv + 1);
}

int main()
{
   citire();
         DF(r, 1);   // consideram ca radacina este pe nivelul 1
         cout<<"\nNumarul de niveluri ale arborelui dat este = "<<maxim;
         return 0;
}
```

Exemplu:
Pentru un arbore cu 5 noduri si muchiile [1,2] [2,3] [1,4] [3,5] numărul de niveluri este 4.



Executia programului folosind compilatorul online https://www.jdoodle.com/online-compiler-c++/:



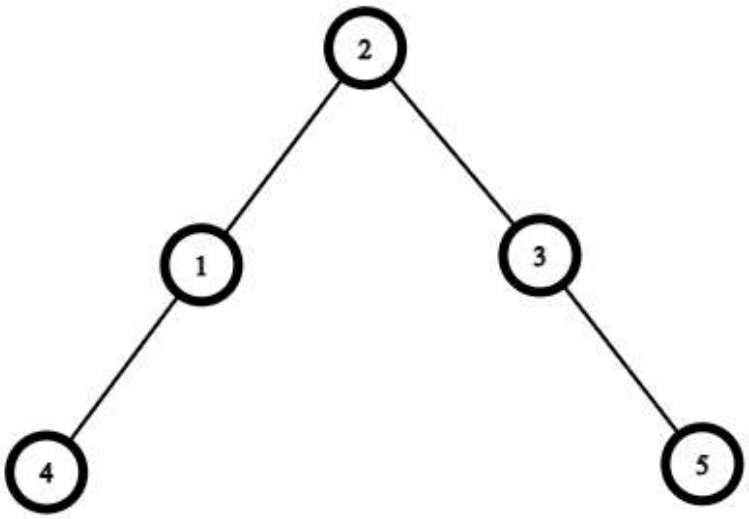
7. Se citește un arbore cu n varfuri dat prin vectorul TATA. Sa se afiseze frunzele arborelui.

Rezolvare:

Codul sursa este urmatorul:

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

int n, T[100], p[100], i;

int main()
{
        cout<<"Dati numarul de noduri ale arborelui = ";
        cin>>n;cout<<n;
        for(i=1;i<=n;i++)
  {
     cin>>T[i];
        p[T[i]]=1;
  }
  cout<<"\nFrunzele arborelui dat sunt: ";
        for(i=1;i<=n;i++)
          if(!p[i]) cout<<i<<" ";
        return 0;
}
```

Exemplu: Pentru vectorul de tati 2 0 2 1 3 se vor afișa frunzele 4 și 5.



Executia programului folosind compilatorul online https://www.jdoodle.com/online-compiler-c++/:

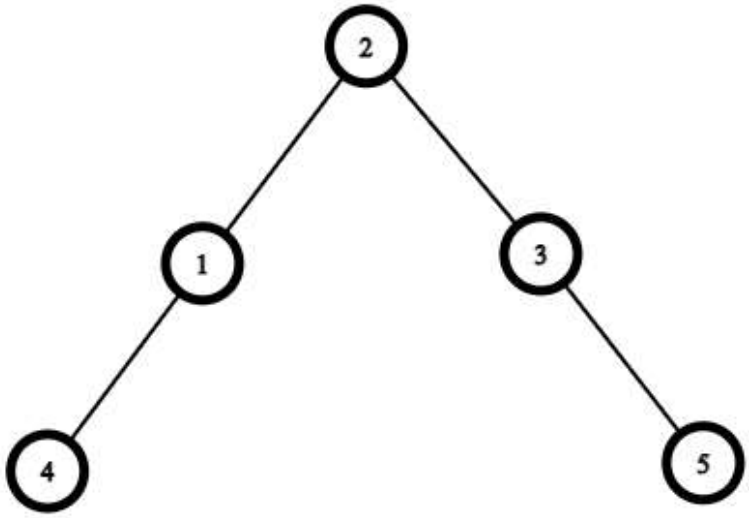
8. Se citeste un arbore cu n varfuri dat prin vectorul TATA.
a) Sa se afiseze gradele varfurilor.
b) Sa se afiseze pentru fiecare varf nivelul pe care se afla (numerotarea nivelelor incepe de la 0-radacina).

Rezolvare:

Codul sursa este urmatorul:

```
#include <iostream>
using namespace std;
int n, r, i, T[100], D[100], p[100], niv[100];

void afis()
{
   cout<<"\nGradele fiecarui nod al aborelui sunt: ";
   for(int i=1;i<=n;i++) cout<<D[i]<<" ";
  cout<<"\nNivelele pe care se afla fiecare nod sunt (se considera ca radacina este pe nivelul
0): ";
   for(i=1;i<=n;i++) cout<<niv[i]<<" ";
}

void df(int r)
{
  for(int i=1;i<=n;i++)
   if(T[i]==r && !p[i])
     { p[i]=1;
       niv[i]=niv[r]+1;
       df(i);
     }
}

int main()
{
        cout<<"Dati numarul de noduri ale arborelui: ";
        cin>>n;cout<<n;
        for(i=1;i<=n;i++)
        {
                cin>>T[i];
                if(T[i]!=0)
                {
                D[i]++;
                D[T[i]]++;
                }
        }
        for(i=1;i<=n;i++)
                if(T[i]==0) r = i;
        niv[r] = 0;
        df(r);
        afis();
        return 0;
}
```

Exemplu: Pentru vectorul de tati 2 0 2 1 3 se vor afișa următorii vectori:
Gradele: 2 2 2 1 1
Nivelele: 1 0 1 2 2



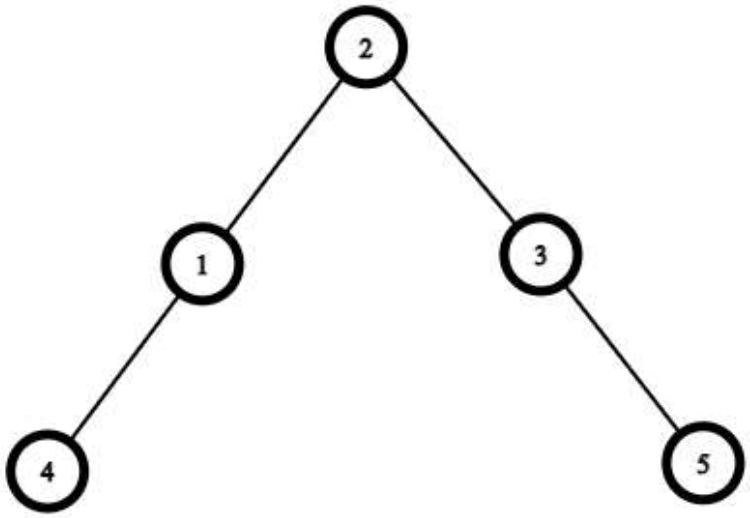
Executia programului folosind compilatorul online https://www.jdoodle.com/online-compiler-c++/:

9. Se citește un arbore cu n varfuri dat prin vectorul TATA și apoi un varf k. Sa se afiseze vectorul TATA obtinut prin mutarea radacinii arborelui în varful k.

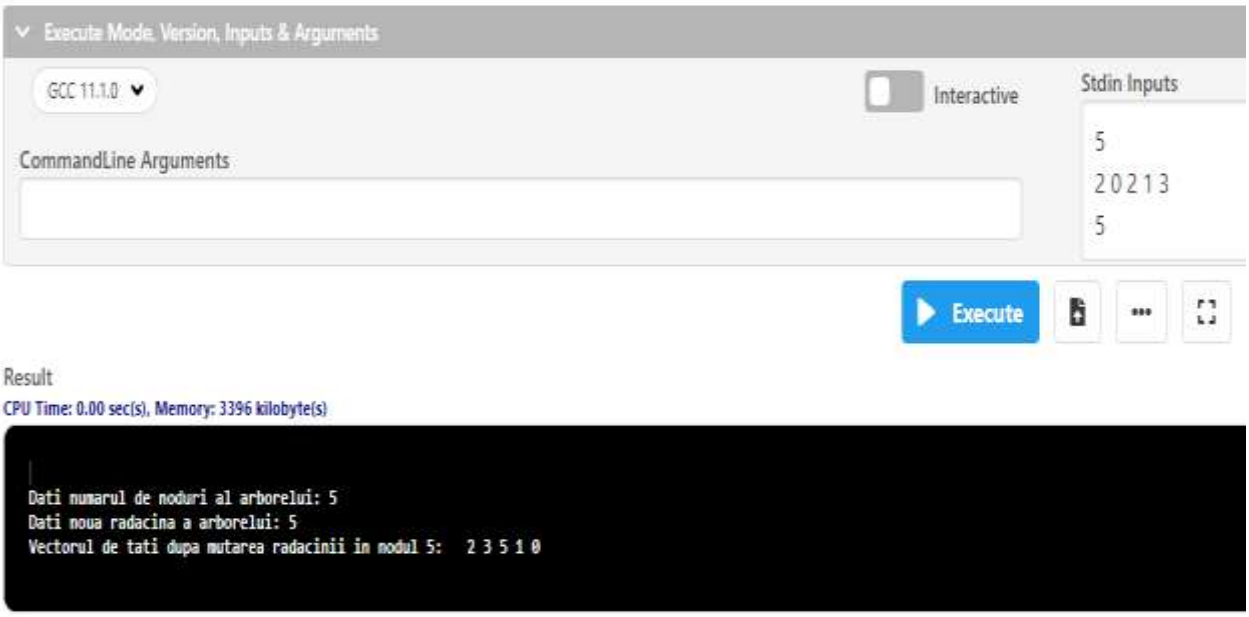
Rezolvare:
Codul sursa este următorul:

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

int n, t[100], k;

void citire()
{
   cout<<"\nDati numarul de noduri al arborelui: ";
   cin>>n;cout<<n;
         for(int i=1;i<=n;i++) cin>>t[i];
         cout<<"\nDati noua radacina a arborelui: ";
         cin>>k;cout<<k;
}
void f(int k)
{
   if(t[k]) f(t[k]);
         t[t[k]]=k;
}

void afisare()
{
   for(int i=1;i<=n;i++) cout<<t[i]<<" ";
}

int main()
{
   citire();
         f(k);
         t[k] = 0;
         cout<<"\nVectorul de tati dupa mutarea radacinii in nodul "<<k<<":   ";
         afisare();
         return 0;
}
```

Exemplu: Pentru vectorul de tati 2 0 2 1 3 și nodul k = 5 se va afișa vectorul 2 3 5 1 0.



Executia programului folosind compilatorul online https://www.jdoodle.com/online-compiler-c++/:



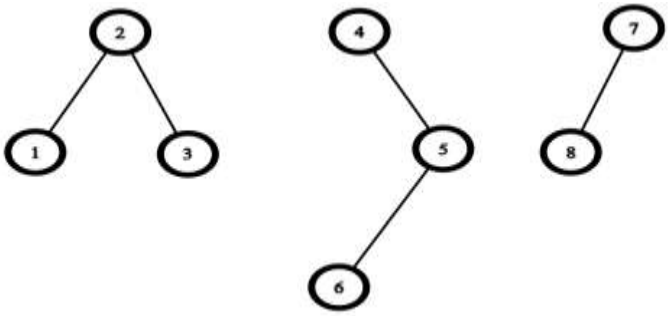
10. Se citește o pădure cu n varfuri prin vectorul de tati. Sa se determine din cati arbori este formată pădurea.

Rezolvare:

Codul sursa este următorul:

```
#include <iostream>
using namespace std;
int t[100], n, k;
void citire()
{
   cout<<"\nDati numarul de noduri al arborelui: ";
   cin>>n;cout<<n;
   for(int i=1;i<=n;i++)   cin>>t[i];
}
int main()
{
   citire();
   cout<<"\nNumarul de arbori din padure este: ";
         for(int i=1; i <= n; i++)
         if(t[i] == 0) k++;
            cout<<k<<" ";
   return 0;
}
```

Exemplu: Pentru vectorul de tati 2 0 2 0 4 5 0 7, pădurea este formată 3 arbori.



Executia programului folosind compilatorul online https://www.jdoodle.com/online-compiler-c++/:

Probleme propuse spre rezolvare

1. Să se ruleze programele prezentate mai sus, urmărind apelurile şi valorile parametrilor de apel.

2. Să se scrie o funcţie care returnează numărul de nivele ale unui arbore binar (înălţimea arborelui). Funcţia va primi ca parametru un pointer p către rădăcina arborelui.

3. Să se scrie o procedură care tipărește informaţiile din nodurile aflate pe un anumit nivel dat. Procedura va primi ca parametri numărul nivelului, precum şi un pointer către rădăcina arborelui.

4. Să se afle înălțimea unui arbore cu rădăcină, adică distanța maximă de la oricare din frunze la rădăcină

5. Să se afle diametrul unui arbore fără rădăcină, adică distanța maximă între oricare două frunze

6. Să se afle centrul sau bicentrele unui arbore, adică nodul sau nodurile care minimizează distanța maximă până la oricare dintre frunze

7. Din două parcurgeri ale unui arbore, să se reconstituie muchiile lui:
   - Arbore general: preordine + postordine
   - Arbore binar: oricare dintre preordine, inordine, postordine

8. Un arbore binar retine numere întregi.
a) să se afișeze numerele utilizând una dintre metode.
b) să se afișeze numerele pare din arbore
c) sa se determine cel mai mare numar din arbore
d) sa se determine suma cifrelor tuturor numerelor din arbore
e) afisati frunzele
f) sa se determine daca exista o anumita valoare in arbore
g) sa se determine daca arborele contine numere prime
h) sa se genereze oglinditul arborelui
i) sa se afiseze subordonatii stangi
j) sa se inlocuiasca o cheie cu o alta
k) sa se inverseze doua chei
l) sa se afiseze fratele lui x
m) sa se afiseze tatal lui x
n) sa se afiseze fii (fiul) lui x
o) sa se determine minimul din arbore
p) sa se afiseze nodurile cu un singur subordonat

9. Fie un arbore binar memorat prin vectorii stang si drept. Sa se parcurga arborele prin cele trei metode.

Laborator 6

10. Fie un arbore binar.
a) Sa se afiseze cate niveluri are arborele
b) Sa se afiseze nodurile de pe nivelul x
c) sa se afiseze nodurile pe niveluri
d) Calculati si afisati suma nodurilor de pe un nivel dat
e) sa se afisese frunzele care nu se gasesc pe ultimul nivel

11. Un arbore binar retine caractere.
a) sa se determine cate vocale retine arborele
b) se citeste un sir de caractere de la tastatura. Sa se determine daca sirul citit este egal cu sirul determinat de parcurgerea arborelui (svd, vsd sau sdv).

12. Fie un graf orientat memorat prin matricea de adiacenta. Sa se determine daca graful poate fi arbore binar. In caz afirmativ, pentru o solutie oarecare, sa se parcurga svd.

13. Fie un arbore binar. Sa se completeze arborele astfel incat fiecare nod sa aiba 2 subordonati. Valoarea cu care se face completarea se citeste de la tastatura.

14. Fie un arbore binar cu n noduri numerotate de la 1 la n cu radacina 1, in care cheia fiecarui nod este un numar intreg. Numarul de noduri se citeste de la tastatura. Reprezentarea in memorie se face inlantuit cu referinte descendente astfel: pe fiecare dintre cele n randuri ale fisierului text **ARBORE.IN** se afla cate trei numere intregi, separate prin cate un spatiu reprezentand fiul stang, fiul drept si valoarea cheii fiecarui nod al arborelui. Sa se scrie cate un program C++ pentru fiecare dintre cerintele de mai jos:
a) sa se afiseze nodurile care retin informatiile numere pare
b) sa se afiseze nodurile care au doar descendent stang
c) sa se afiseze cheile nodurilor care au doar descendent drept
d) sa se afiseze cheile din nodurile terminale
e) sa se afiseze fiii nodului x, furnizat de utilizator
f) sa se afiseze nodul tata al unui nod x, furnizat de utilizator
g) sa se scrie in fisierul noduri.out, nodurile ale caror chei sunt egale cu o valoare val, citita de la tastatura
h) sa se calculeze inaltimea arborelui binar dat
i) sa se determine nivelul maxim
j) sa se afiseze cheia maxima