

Laborator 5
Grafuri neorientate. Parcurgeri in grafuri neorientate:
1. Parcurgerea in latime (*Breadth First*)
2. Parcurgerea in adancime (*Depth First*)

Probleme rezolvate

1. Algoritmul de parcurgere in latime (Breadth First):

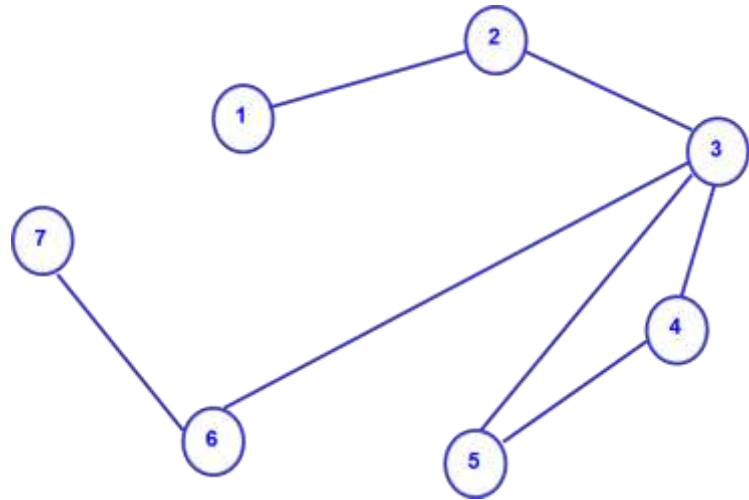
```
#include<iostream>
using namespace std;

int viz[30],n,i,j,k,u,v,p,a[20][20],c[30];
int main(void)
{
    cout<<"Dati numarul de varfuri n = ";
    cin>>n;cout<<n<<'\n';
    for(i=1; i<=n-1; i++)
        for(j=i+1; j<=n; j++){
            cin>>a[i][j];
            a[j][i] = a[i][j];
        }
        cout<<"Dati varful de plecare ";    cin>>i;cout<<i<<'\n';
    for(j=1; j<=n; j++) viz[j]=0;
    c[1]=i;
    p=1;
    u=1;
    viz[i]=1;
    while(p<=u){
        v=c[p];
        for(k=1; k<=n; k++){
            if( (a[v][k]==1) && (viz[k]==0) ){
                u++;
                c[u]=k;
                viz[k]=1;
            }
        }
        p++;
    }
    cout<<"Lista varfurilor in parcugerea in latime: \n";
    cout<<i<<" ";
    for(j=2; j<=u; j++) cout<<c[j]<<" ";
    return 0;
}
```

Sa se aplice acest algoritm pentru graful $G=\{X,U\}$, unde $X=\{1,2,3,4,5,6,7\}$ si $U=\{(1,2), (2,3), (3,4), (3,5), (3,6), (6,7)\}$

Varful de plecare este 1

Parcurea in latime permite afişarea următoarelor varfuri: 1, 2, 3,4, 5, 6, 7



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

The screenshot shows an online C++ compiler interface. At the top, it says "Execute Mode, Version, Inputs & Arguments". The compiler version is "GCC 11.1.0". There is an "Interactive" toggle switch. The "Stdin Inputs" section contains the following text:
7
100000
10000
1110
000
00
1
1
The "CommandLine Arguments" field is empty. At the bottom right, there is an "Execute" button and some utility icons. Below the compiler interface, the "Result" section shows:
CPU Time: 0.00 sec(s), Memory: 3368 kilobyte(s)
Dati numarul de varfuri n = 7
Dati varful de plecare 1
Lista varfurilor in parcurea in latime:
1 2 3 4 5 6 7

2. Parcurgerea in latime – varianta recursiva:

Se construiește o funcție numită BF_recursiva cu un parametru formal - i, care reprezintă poziția curentă la care s-a ajuns în coadă

Algoritmul este următorul:

se parcurg nodurile grafului, cu j:

- dacă j este adiacent cu nodul curent din coadă și j este nevizitat
- atunci se adaugă la coadă;
- și apoi se marchează ca fiind vizitat;
- dacă mai sunt elemente în coadă se trece la următorul și se reapelează funcția

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

int a[20][20];
int coada[20], viz[20];
int i, n, j, u, nod_plecure, m, x, y;

void BF_recursiva(int i)
{
    int j,v;
    for (j=1;j<=n;j++)
    {
        v=coada[i];
        if ((a[v][j]==1) && (viz[j]==0))
        {
            u=u+1;
            coada[u]=j;
            viz[j]=1;
        }
    }
    if (i<=u) BF_recursiva(i+1);
}

int main()
{
    cout<<"numarul de varfuri n = ";
    cin>>n;cout<<n;
    cout<<"numarul de muchii m = ";
    cin>>m;cout<<m;
    for (i=1;i<=m;i++)
    {
        cout<<"\ndati vf. x si y: "; cin>>x>>y;cout<<x<<" "<<y;
        a[x][y]=1; a[y][x]=1;
    }
    for (i=1;i<=n;i++) viz[i]=0;
    cout<<"\ndati nodul de plecare : ";
    cin>>nod_plecure;cout<<nod_plecure<<"\n";
}
```

```
viz[nod_plecare]=1;
coada[1]= nod_plecare;
u = 1;
BF_recurсивa(1);
cout<<"Parcugerea in latime - varianta recursiva este: ";
for(i=1; i<=u; i++) cout<<coada[i]<<" ";
    return 0;
}
```

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

Execute Mode, Version, Inputs & Arguments

GCC 11.1.0

Interactive

Stdin Inputs

7
6
1 2
2 3
3 4
3 5
3 6
6 7
1

CommandLine Arguments

Execute

Result

CPU Time: 0.00 sec(s), Memory: 3504 kilobyte(s)

```
numarul de varfuri n = 7numarul de muchii m = 6
dati vf. x si y: 1 2
dati vf. x si y: 2 3
dati vf. x si y: 3 4
dati vf. x si y: 3 5
dati vf. x si y: 3 6
dati vf. x si y: 6 7
dati nodul de plecare : 1
Parcugerea in latime - varianta recursiva este: 1 2 3 4 5 6 7
```

3. Algoritm de parcurgere in latime (Depth First):

Codul sursa al algoritmului de parcurgere in adancime:

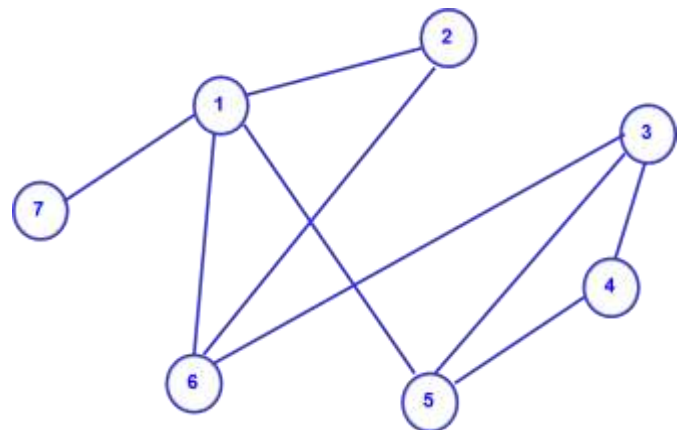
```
#include<iostream>
using namespace std;

int n, m, i, j, p, a[20][20], viz[30];
void df(int k)
{
    int j;
    cout<<k<<" ";
    viz[k]=1;
    for(j=1; j<=n; j++)
        if( (a[k][j]==1) && (viz[j]==0) ) df(j);
    return;
}
int main(void)
{
    cout<<"\nDati numarul de varfuri n = ";
    cin>>n;cout<<n;
    for(i=1; i<=n-1; i++)
        for(j=i+1; j<=n; j++)
        {
            cin>>a[i][j];
            a[j][i]=a[i][j];
        }
    cout<<"\nDati varful de plecare ";
    cin>>p;cout<<p;
    cout<<"\nParcurgerea in adancime (Depth First): ";
    df(p);
    return 0;
}
```

Să considerăm următorul graf neorientat:

Metoda *DF*, aplicată acestui graf, pornind de la vârful inițial 1, conduce la vizitarea varfurilor în următoarea ordine:

1,2,6,3,4,5,7



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

Execute Mode, Version, Inputs & Arguments

GCC 11.1.0

Interactive

Stdin Inputs

7
1 0 0 1 1 1
0 0 0 1 0
1 1 1 0
1 0 0
0 0
1
1

CommandLine Arguments

Execute

Result

CPU Time: 0.00 sec(s), Memory: 3500 kilobyte(s)

```
Dati numarul de varfuri n = 7
Dati varful de plecare 1
Parcuretea in adancime (Depth First): 1 2 6 3 4 5 7
```

4. Se da un graf $G=\{X,U\}$ cu n varfuri si m muchii. Sa de determine componentele conexe ale grafului G .

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

int a[20][20],cc[30];
int n,ncc,i,j,v;      // ncc=nr. de componente conexe
void df(int v)
{
    int i;
    cc[v]=ncc;
    for(i=1;i<=n;i++)
        if( (a[v][i]==1) && (cc[i]==0) )
            df(i);
}

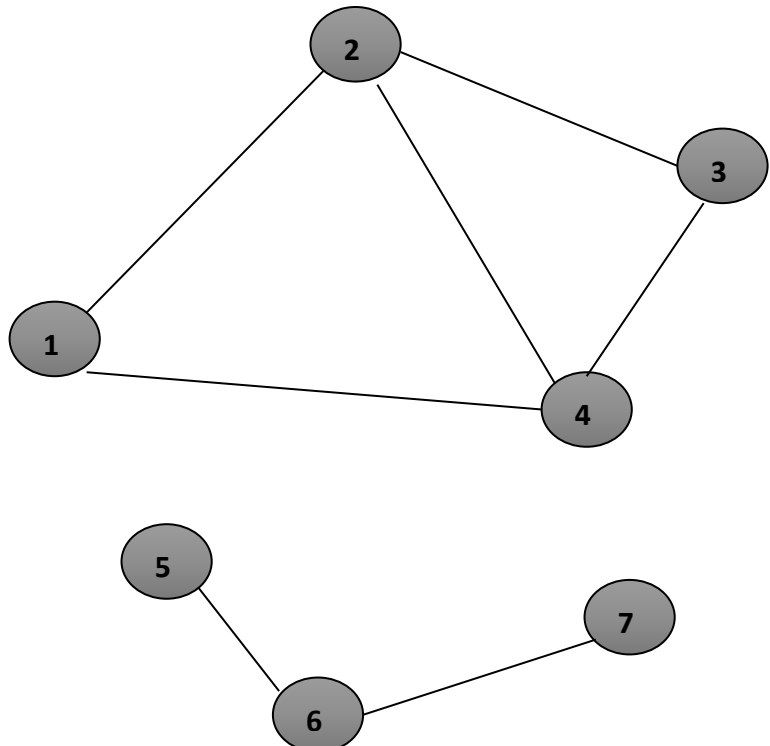
int main(void)
{
    cout<<"Dati numarul de varfuri n = ";
```

```

cin>>n; cout<<n;
cout<<"\nMatricea de adiacenta "<<endl;
for(i=1;i<=n-1;i++)
  for(j=i+1;j<=n;j++)
  {
    cin>>a[i][j];
    a[j][i]=a[i][j];
  }
for(i=1;i<=n;i++){
  for(j=1;j<=n;j++)  cout<<a[i][j]<<" ";
  cout<<'\n';
}
ncc = 0;
for(i=1;i<=n;i++) cc[i]=0;
for(i=1;i<=n;i++)
  if (cc[i]==0) {
    ncc=ncc+1;
    df(i);
  }
for(i=1;i<=ncc;i++){          // ncc = nr. de comp. conexe
  cout<<"componenta "<<i<<": ";
  for(j=1;j<=n;j++)
    if(cc[j]==i)  cout<<j<<" ";
  cout<<endl;
}
return 0;
}

```

Să considerăm următorul graf neorientat:



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

Execute Mode, Version, Inputs & Arguments

GCC 11.1.0 Interactive Stdin Inputs

7
101000
10000
1000
000
10
1

CommandLine Arguments

Execute

Result

CPU Time: 0.00 sec(s), Memory: 3492 kilobyte(s)

```
Dati numarul de varfuri n = 7
Matricea de adiacenta
0 1 0 1 0 0 0
1 0 1 0 0 0 0
0 1 0 1 0 0 0
1 0 1 0 0 0 0
0 0 0 0 0 1 0
0 0 0 0 1 0 1
0 0 0 0 0 1 0
componenta 1: 1 2 3 4
componenta 2: 5 6 7
```

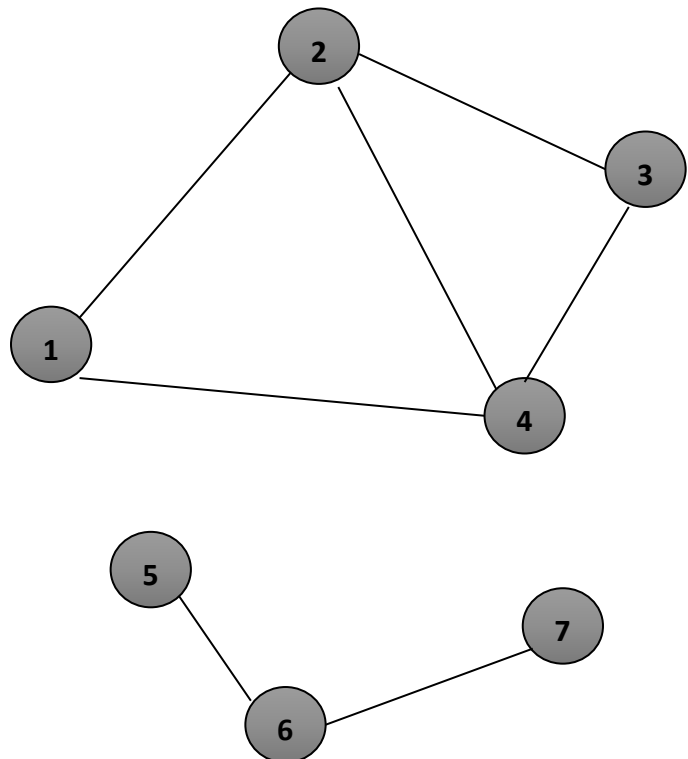
5. Să se verifice dacă un graf dat prin matricea de adiacență este **graf conex** si să se afișeze un mesaj corespunzător.

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

int a[10][10], n, viz[10];
void citire()
{
    int i,j;
    for(i=1;i<n;i++)
        for(j=i+1;j<=n;j++)
        {
            cin>>a[i][j];
            a[j][i]=a[i][j];
        }
}
void parcurg(int x)
{
    int i;
```

```
viz[x]=1;
for(i=1;i<=n;i++)
    if(a[x][i]&&viz[i]==0) parcurg(i);
}
int conex()
{
int i;
parcurg(1);
for(i=1; i<=n; i++)
    if(viz[i]==0) return 0;
return 1;
}
int main()
{
cout<<"Numarul de varfuri n=";
cin>>n; cout<<n;
citire();
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";
}
if(conex()==1)        cout<<"Graful dat este conex";
else        cout<<"Graful dat NU este conex";
return 0;
}
```

Să considerăm următorul graf neorientat:



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

The screenshot shows an online C++ compiler interface. At the top, it displays 'Execute Mode, Version, Inputs & Arguments'. The compiler version is set to 'GCC 11.1.0'. There is an 'Interactive' toggle switch. A 'Stdin Inputs' panel on the right contains the following input: 7, 101000, 10000, 1000, 000, 10, 1. Below the compiler settings is a 'CommandLine Arguments' text input field. At the bottom right, there is a blue 'Execute' button and three utility icons (copy, refresh, fullscreen). Below the compiler interface, the 'Result' section shows 'CPU Time: 0.00 sec(s), Memory: 3448 kilobyte(s)'. The output is displayed in a black terminal window with white text:

```
Numarul de varfuri n=7
Matricea de adiacenta:
0 1 0 1 0 0 0
1 0 1 0 0 0 0
0 1 0 1 0 0 0
1 0 1 0 0 0 0
0 0 0 0 1 0
0 0 0 0 1 0 1
0 0 0 0 1 0
Graful dat NU este conex
```

Probleme propuse spre rezolvare

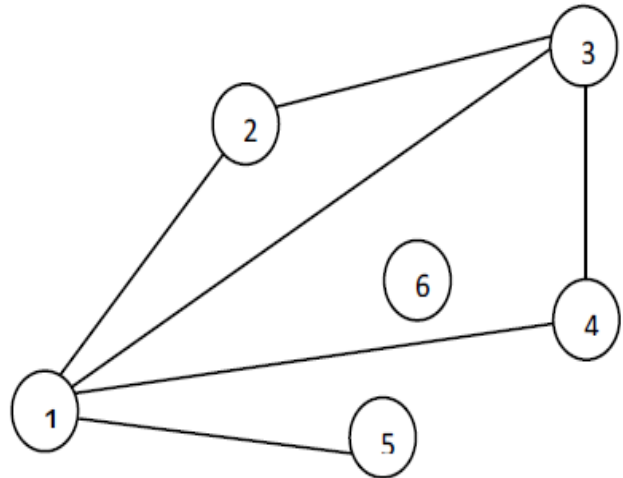
1. Să se ruleze programele prezentate mai sus, urmărind apelurile și valorile parametrilor de apel.
2. Fie graful neorientat $G=(X,U)$, unde numărul de noduri este 7 și $U=\{(1,2), (2,4), (2,7), (3,6), (3,7), (4,6), (4,7), (6,7)\}$. Sirul de noduri 7, 3, 6, 7, 4, 2, 7 este:
 - a) lant neelementar
 - b) lant compus
 - c) ciclu
 - d) ciclu elementar
3. Fie graful neorientat $G=(X,U)$, unde numărul de noduri este 7 și $U=\{(1,2), (2,4), (2,7), (3,6), (3,7), (4,6), (4,7), (6,7)\}$. Câte noduri cu grad impar sunt în graful G:
 - a) 3
 - b) 6
 - c) 5
 - d) 4
4. Scrieti răspunsul pentru fiecare dintre cerintele următoare.
 - a) Ce reprezintă un subgraf?
 - b) Desenati un graf bipartit care să aibă gradele nodurilor: 1, 1, 2, 2, 2, 3, 3
 - c) Scrieti o functie prin care să se verifice dacă un graf conține noduri izolate.
 - d) Care este numărul total de grafuri neorientate cu n noduri?
5. Fie graful neorientat $G=(X,U)$, unde numărul de noduri este 8 și $U=\{(2,5), (3,5), (3,6), (5,8), (6,7), (7,8)\}$. Care sunt nodurile care nu aparțin nici unui ciclu?
 - a) 2, 3, 5
 - b) 1, 2, 4
 - c) 3, 7, 8
 - d) 1, 4, 7
6. Care dintre secvențele următoare reprezintă sirul gradelor nodurilor unui graf complet?
 - a) 4 4 4 4
 - b) 2 2 2 2
 - c) 5 5 5 5 5 5
 - d) 3 3 3 3 3

PROIECTAREA ALGORITMILOR

Laborator 5

7. Scrieti răspunsul pentru fiecare dintre cerințele următoare.

- Ce reprezintă un graf partial?
- Pentru graful $G=(X, U)$ din figură scrieti: matricea de adiacență și nodul de grad maxim
- Scrieti funcția de parcurgere în lățime a unui graf.
- Care este numărul de muchii al unui graf complet?
- Scrieti o funcție care afișează toate nodurile de grad maxim dintr-un graf neorientat.



8. Determinați vecinii unui varf al unui graf orientat.

9. Determinați gradele exterioare și interioare ale varfurilor unui graf, gradul exterior minim, gradul exterior maxim, gradul interior minim și gradul interior maxim.

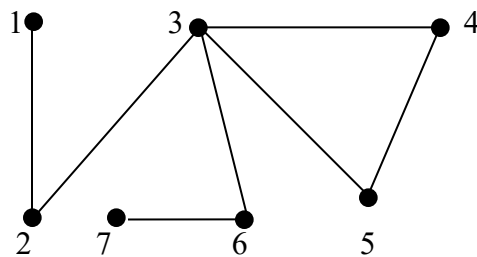
10. Verificați dacă un graf este simetric/antisimetric.

11. Se citește matricea de adiacență a unui graf orientat. Să se afișeze toate nodurile pentru care $d_+(x)=d_-(x)$ (gradul exterior este egal cu gradul interior). Pentru un nod x citit de la tastatură să se afișeze toate nodurile adiacente cu acestea.

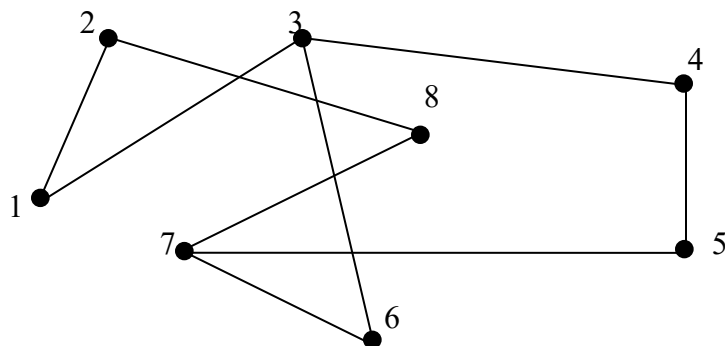
12. Să se verifice dacă o secvență de noduri dată reprezintă un drum elementar sau ne-elementar într-un graf orientat. Numărul de noduri, matricea de adiacență, și secvența de noduri se citesc de la tastatură.

13. Descrieți în pseudocod sau în C++ algoritmii de căutare în adâncime și în lățime în grafuri. Parcurgeți în adâncime, respectiv în lățime următoarele grafuri:

a) Graf neorientat



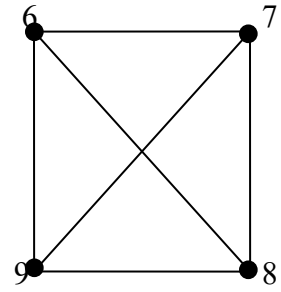
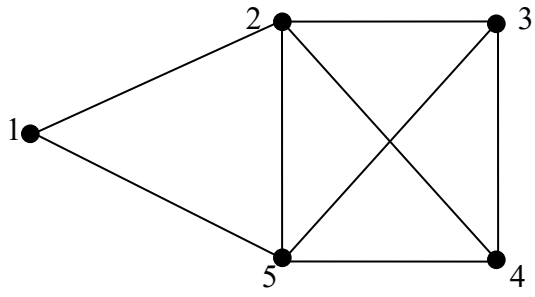
b) Graf neorientat



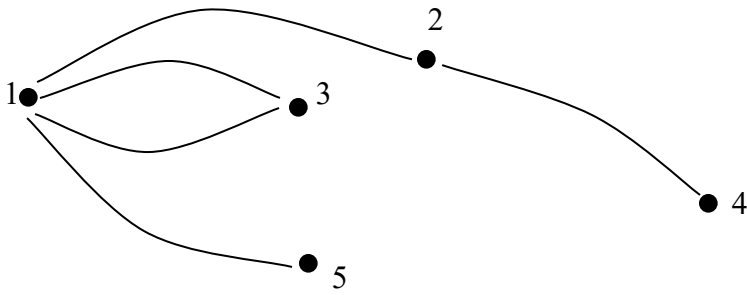
PROIECTAREA ALGORITMILOR

Laborator 5

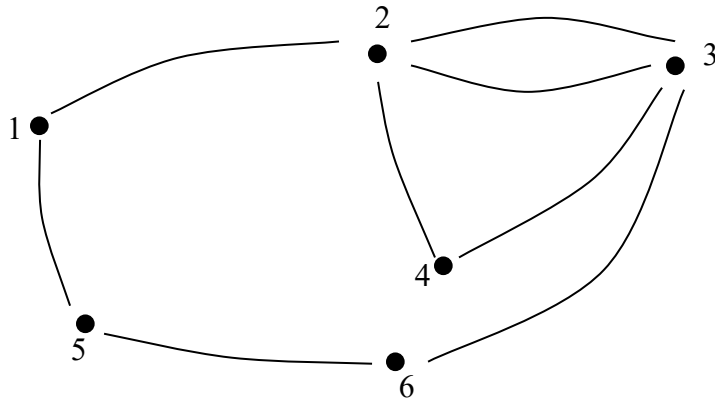
c) Graf neconex



d) Graf orientat



e) Graf orientat



f) Graf neorientat conex

