Laborator 7

Grafuri orientate. Conexitate. Drumuri minime si maxime in grafuri

Probleme rezolvate

Aplicatie:

Enunt: Pentru alimentarea cu apa a unui cartier exista o statie de pompare a apei la care sunt conectate cateva blocuri (nu toate). Stiind ca mai sunt conectate si unele blocuri intre ele determinati daca pentru o configuratie data de conectare a blocurilor si a statiei de pompare exista posibilitatea de alimentare cu apa a intregului cartier.

Datele de intrare au urmatoarea forma:

N-numarul de blocuri;
Ns-nodul de start;nr blocului unde este statia de pompare
x y
.
.
.
.
} exista conexitate intre blocul x si blocul y;

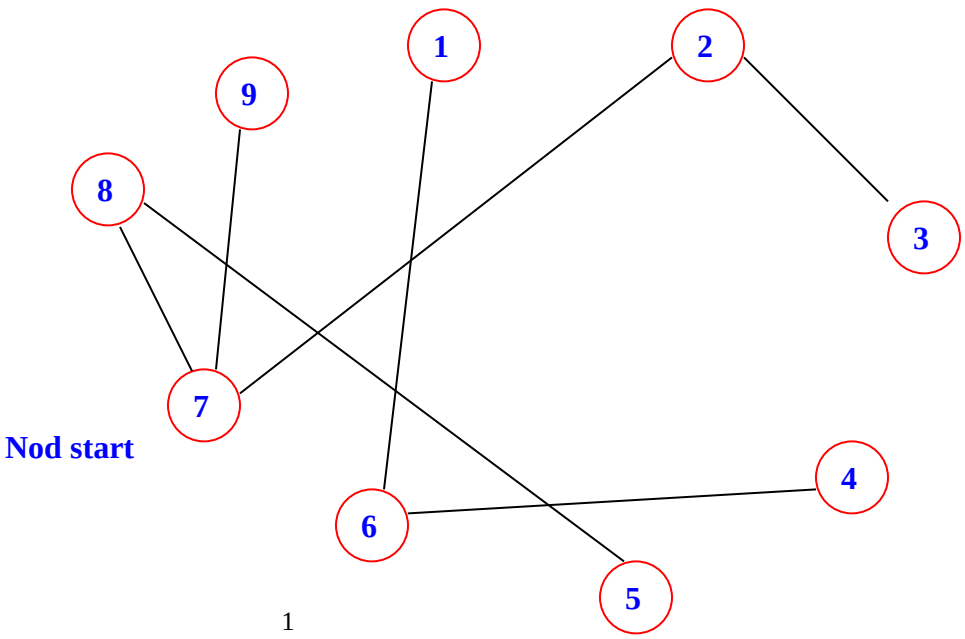
Rezolvare:

Cu datele problemei se construieste un graf in care nodurile grafului reprezinta conexiunile dintre blocuri. Se parcurge graful in latime avand drept nod de start blocul in care se afla statia de pompare. Daca in urma parcurgerii s-au vizitat toate nodurile, inseamna ca se poate realiza alimentarea cu apa a intregului cartier.

De exemplu, pentru configuratia urmatoare:

```
9
7
7 2
7 9
7 8
2 3
8 5
6 1
6 4
```

Se obtine graful:

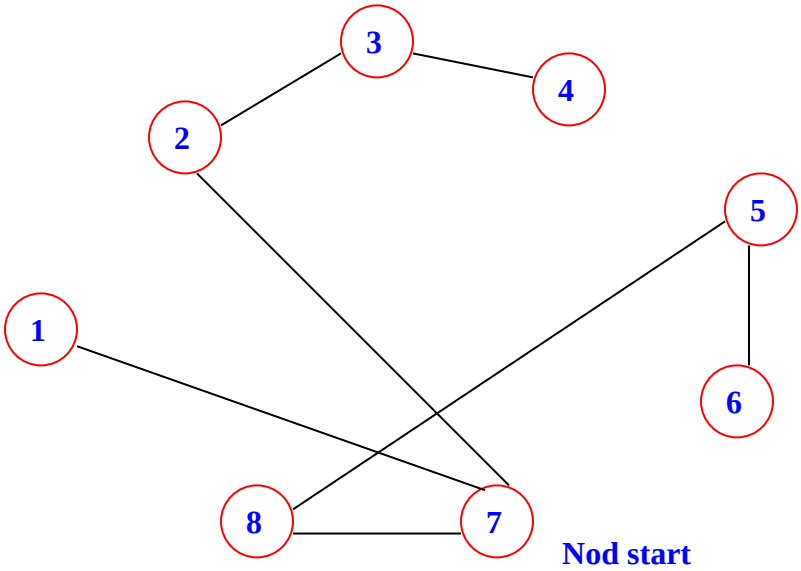


Si programul afiseaza:
"Nu se poate alimenta cu apa intreg cartierul";

Pentru cel de-al doilea exemplu:

8
7
7 1
7 2
7 8
2 3
3 4
8 6
8 5

Se obtine graful:



Si programul afiseaza:
"Se poate alimenta cu apa intreg cartierul";

Codul sursa este urmatorul:

```
#include<iostream.h>
int c[100],v[100],a[20][20],nr,ns,n,m;
void citire()
{
        int x,y,i,j;
        cout<<"Dati numarul de noduri n = ";cin>>n;
        cout<<"Dati nodul de start ns =  ";cin>>ns;
        cout<<"Dati numarul de muchii m = ";cin>>m;
        for(i=1;i<=m;i++)
        {
                cout<<"dati x = ";cin>>x;
                cout<<"dati y = ";cin>>y;
                a[x][y]=1;
                a[y][x]=1;
        }
}
int prim_nevizitat()
{
        int i=1;
        while(v[i]!=0 && i<=n)
                i++;
        if(i<=n) return 1;
        else     return n+1;
}
void parcurgere()
```

```
{
        int i, lc, x;
        nr=0;
        while(ns<=n)
        {
                nr=nr+1;
                c[1]=ns;
                cout<<ns<<" ";
                v[ns]=1;
                lc=1;
                while(lc!=1)
                {
                        x=c[1];
                        for(i=1;i<=n;i++)
                                if(a[x][i]==1 && v[i]==0)
                                {
                                        lc++;
                                        c[lc]=i;
                                        cout<<" "<<i;
                                        v[i]=1;
                                }
                                for(i=1;i<=lc-1;i++)
                                        c[i]=c[i+1];
                                lc=lc-1;
                }
                ns=prim_nevizitat();
        }
}

int main()
{
        citire();
        parcurgere();
        if(nr==1)
                cout<<"Se poate alimenta cu apa intreg cartierul"<<endl;
        else
                cout<<"NU se poate alimenta cu apa intreg cartierul"<<endl;
}
```

Probleme propuse spre rezolvare

1. Să se ruleze programul prezentat mai sus, urmărind apelurile şi valorile parametrilor de apel.

2. Determinati vecinii unui varf al unui graf orientat.

3. Determinati gradele exterioare si interioare ale varfurilor unui graf, gradul exterior minim, gradul exterior maxim, gradul interior minim si gradul interior maxim.

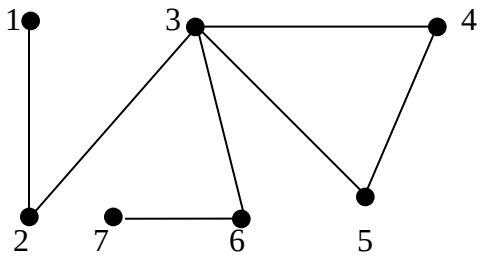
4. Verificati daca un graf este simetric/antisimetric.

5. Se citeste matricea de adiacenta a unui graf orientat. Sa se afiseze toate nodurile pentru care $d^+(x)=d^-(x)$ (gradul exterior este egal cu gradul interior). Pentru un nod x citit de la tastatura sa se afiseze toate nodurile adiacente cu acestea.

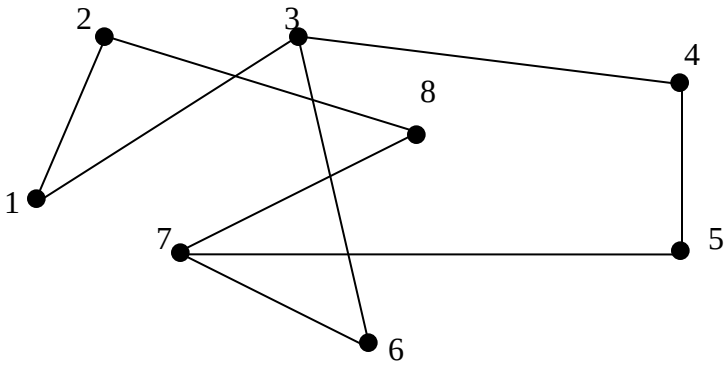
6. Sa se verifice daca o secventa de noduri data reprezinta un drum elementar sau ne-elementar intr-un graf orientat. Numarul de noduri, matricea de adiacenta, si secventa de noduri se citesc de la tastatura.

7. Descrieti in pseudocod sau in C++ algoritmii de cautare in adancime si in latime in grafuri. Parcurgeti in adancime, respectiv in latime urmatoarele grafuri:

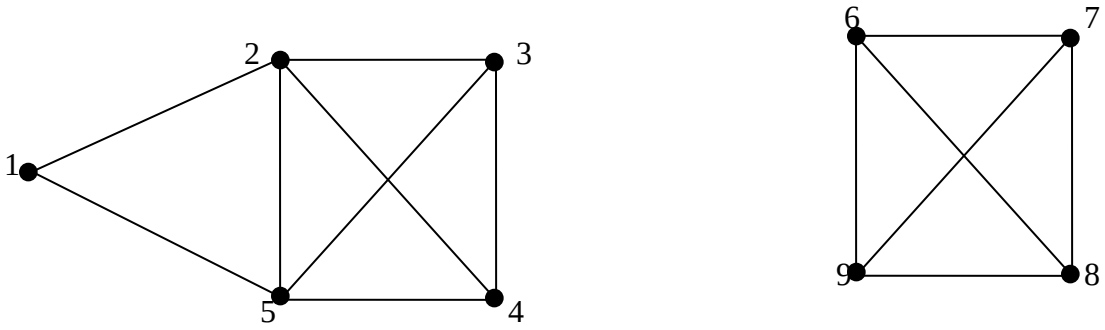
a) Graf neorientat



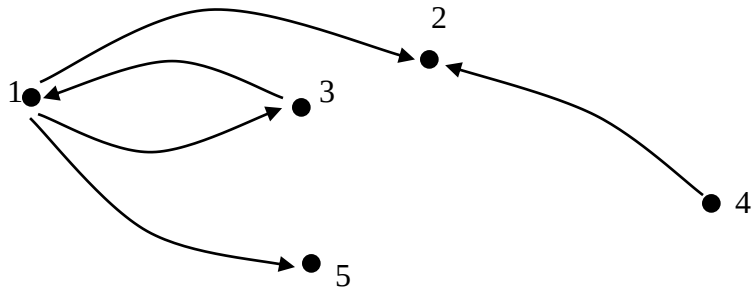
b) Graf neorientat



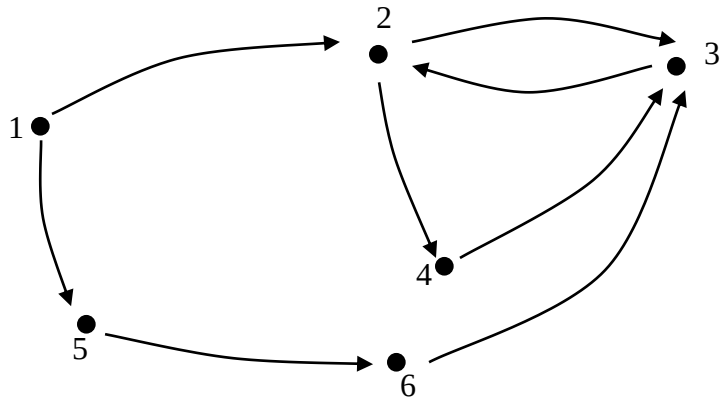
c) Graf neconex



d) Graf orientat



e) Graf orientat



f) Graf neorientat conex