Lucrare de control -informatică –nr1

I.Se consideră un graf neorientat ,cu n noduri ,reprezentat prin matricea de adiacență a.
Răspundeți următoarelor cerințe:
1)Precizați care dintre următoarele secvențe pseudocod realizeztă construirea matricei de adiacență prin citirea celor n muchii de la tastatură:

a)
```
pentru k=1 la m execută
  bloc citeşte (u,v)
  a[u,v]=1
   a[v,u]=1
  sf bloc
```

```
bloc  repetă
citeşte (a[I,k])
până când  a[I,k]=0 sau a[I,k]=1
a[k,i]=a[I,k]
sf bloc
```

```
pentru k=1 la m execută
  bloc  repetă
citeşte (u,v)
până când   u>=1 şi u<=n şi v>=1
şi v<=n,a[u,v]=1,a[v,u]=1 sf bloc
```

b)
```
pentru k=1 la n execută a[k,k]=0
pentru i=1 la n execută
pentru k=i+1 la n execută
```

c)
```
pentru i=1 la n execută
  pentru k=1 la n execută
  a[I,k] =0
```

……

II.Modificați următoarea secvență reprezentată prin pseudocod astfel încât să determine gradul maxim al vârfurilor din graful dat .Matricea de adiacență se consideră completată.Explicați semnificația vectorului d şi a variabilei m.

```
{1} pentru i=1 la n execută
{2} pentru j=i la n execută
{3}dac[ a[I,j]=1 atunci
{4}d[i]<-d[i]+1
{5}d[j]<-d[i]
 Sf dacă
{6}m<-0
{7}pentru k=2 la n execută
{8}dacă d[k]>m atunci
{9} m<-d[k]
```

III.Parcurgeți în lățime graful din figură .
IV. Graf conex.Def. Ex.
V.Să se scrie procedura backtracking pentru aflarea ciclurilor de lungime l şi procedura de afişare a soluțiilor într-un fişier text ,fiecare soluție pe un rând.Să se explice semnificația variabilelor folosite.
VI.Să se scrie programul pentru verificarea conexității unui graf neorientat citind matricea de adiacență de la tastatură . Rezultatul să fie afişat într-un fişier text .

Lucrare de control -informatică –nr2

1)Precizați care dintre următoarele programe pseudocod realizează determinarea gradului unui vârf oarecare x:

a)
```
subprogram grad(x natural)
natural
k,g natural
început-grad
g<-0
pentru k=1 la n execută
pentru x=1 la n execută
dacă a[k,x]=1 atunci  g<-g+1
sf dacă
grad <-g
  sf -grad
```

b)
```
) subprogram grad(x natural)
natural
k,g natural
început-grad
g<-0
pentru k=1 la n execută
dacă a[k,x]=1 atunci  g<-g+1
sf dacă
grad <-g
  sf -grad
```

c)
```
subprogram grad(x natural)
natural
k,g natural
început-grad
k<-0  g<-0
repetă
dacă a[k,x]=1 atunci  g<-g+1
sf dacă
k<-k+1
până când   k=x
grad <-g      sf -grad
```

……

II.Modificați următoarea secvență reprezentată prin pseudocod astfel încât să determine condițiile interne necesare validării unui element pentru un ciclu elementar ,al grafului dat .Matricea de adiacență se consideră completată.Explicați semnificația vectorului s şi a variabilei k.

```
{1}v<-adevărat
{2} dacă a[s[k],s[k-1]]=1
{3}atunci v<-fals sf _dacă
{4} dacă v atunci
{5}pentru i=1 la k-1 do execută
{6}bloc dacă s[i]=s[i+1]
{7}atunci v<-fals sf_dacă sf-bloc
```

III.Parcurgeți în lățime graful din figură .
IV.Componentă conexă.Def.Ex.
V. Să se scrie procedura bactracking pentru aflarea tuturor lanțurilor elementare care au ca extremități două vârfuri x,y şi procedura de citire a matricei de adiacență dintr-un fisier text .Să se explice semnificația variabilelor folosite.
VI.Să se scrie programul pentru descompunerea unui graf în componente conexe cunoscând matricea de adiacență de la tastatură.
Ex1-0,25 Ex2-1,25 Ex3-0,5 ex4-0,5 ex5-2,5 ex6-3p

Parcurgerea grafurilor neorientate –recursiv

Parcurgere BF-recursiv (în pp citesc start ,inițializez coada şi vectorul viz cu 0,pi=1, pun vf start în coadă şi îl vizitez ,p-poziția curentă la care s-a ajuns în coadă,în pp se începe de la poziția 1 din coadă deci BF(1)).

```
Procedure Bf (p:integer);
Begin
For  k:=1 to n do
If a[c[p],k]=1 and(viz[k]=0)  then begin
  Pi:=pi+1;
C[pi]:=k;
Viz[k]:=1;
End;
If p<=pi  then Bf(p+1);
End;
```

Parcurgerea DF- are ca parametru nodul curent asupra căruia se aplică parcurgerea.
Se afişează nodul asupra căruia se aplică şi-l marchează cu 1 ca fiind vizitat,apoi pentru fiecare vecin nevizitat de-al nodului curent se face autoapel asupra nodului curent.

```
Procedure Df (p:integer);
Begin
Write(start:4);viz[start]:=1;
For  k:=1 to n do
If a[c[start],k]=1 and(viz[k]=0)  then
  Df(k);
End;
```

Parcurgere df
Citesc nodul de plecare ,îll pun în coadă şi îl vizitez,apoi while pi:'1 do df; sfârşit .

```
Begin
Z:=c[pi];
K:=urm[z]+1;
While (k<=n )and((a[z,k]=0)or(a[z,k]=1))and(viz[k]=1) do k:=k+1;
Urm[z]:=k;
If k=n+1 then pi:=pi-1
Else begin
        Viz[k]:=1;pi:=pi+1;c[pi]:=k;
         Write(k:3);
End;
End;
```