Neumann János Tehetséggondozó Program
Gráfalgoritmusok II.

Horváth Gyula
horvath@inf.elte.hu

1. A szélességi bejárás alkalmazásai.

1.1. Szélességi bejárás

Bemenet: $G = (V,E)$ (irányított vagy irányítatlan) gráf és egy $r \in V$ pont.
Kiszámítandó minden p pontra az r-ből p-be vezető legrövidebb út hossza, és legrövidebb út.
Kimenet: $D : V \to \mathbb{N}$, $Apa : V \to V$, hogy
$D(p) = \delta(r,p)$ és
az $F = (\overline{V},\overline{E})$ gráf, ahol
$\overline{V} = \{p : Apa(p) \neq null \vee p = r\}$,
$\overline{E} = \{(p,q) : Apa(q) = p \wedge p \neq null\}$
Az F gráf a G gráf r-gyökerű legrövidebb utak feszítőfája (LUF).

1.2. Elvi algoritmus

Legyen $G = (V,E)$ terszőleges irányított, vagy irányítatlan gráf.
Jelölje $S(t)$ azon p pontok halmazát, amelyek t távolsára vannak r-től.

$$S(t) = \{p \in V : \delta(r,p) = t\}$$

Nyilvánvaló, hogy $S(0) = \{r\}$.
Jelölés: $D(p) = \delta(r,p)$
Nyilvánvaló, hogy minden $t > 0$-ra és minden $q \in S(t)$ van olyan $p \in S(t-1)$, hogy $(p,q) \in E$
Fordítva, ha $p \in S(t-1)$ és $(p,q) \in E$, akkor $D(q) \leq t+1$, hiszen van $t+1$ hosszú út r-ből q-ba, mert van t hosszú (legrövidebb) út r-ből p-be és van él p-ből q-ba: $r \rightsquigarrow p \to q$.
Tehát az $S(t)$ halmaz előállítható az $S(t-1)$ halmazból.

```
Inf := |V|;  //pontok száma
for p ∈ V do D(p) := Inf;
D(r) := 0;
S := {r};
repeat
    St := ∅;
    for p ∈ S do begin
        for q ∈ KiEl(p) do //  minden p → q  élre
            if D(q) = Inf then begin  //q benemjárt
                D(q) := D(p) + 1;
                Apa(q) := p;  //p-ből megyünk q-ba
                St := St + {q};
            end;
    end;
    S := St;
until S <> ∅ ;
```

A szélességi bejárás megvalósítása Pascal nyelven

```
1  procedure SzeltBejar(r:PontTip; N:longint; const G:Graf;
2      var D:array of longint; var Apa:array of longint);
3  //r–ből induló legrövidebb utak meghatározása
4    var
5      S:Sor;
6      p,q:PontTip;   i:integer;
7    begin{SzeltBejar}
8      for p:=1 to N do D[p]:=Inf;      { inicializálás}
9      D[r]:=0; Apa[r]:=0;
10     Letesit(S);
11     Sorba(S,r);
12     while Elemszam(S)>0 do begin
13       p:=SorBol(S);
14       for i:=1 To KiFok[p] Do Begin { p–>q élre }
15         q:=G[p,i];
16         if D[q]=Inf then begin       { ha q meg nem elért }
17           Apa[q]:=p;
18           D[q]:=D[p]+1;
19           Sorba(S,q);
20         end;
21       end{for p–>q};
22     end{while};
23   end{SzeltBejar};
```

1.2.1. Egy legrövidebb út kiíratása

```
1  procedure UtKiIro(r,p:PontTip);
2  //Globalis: Apa
3  var u,i:longint;
4    Ut:array[1..maxP] of longint;
5  begin
6    u:=0;
7    while p<>r do begin
8      inc(u);
9      Ut[u]:=p;
10     p:=Apa[p];
11   end;
12   write(r);
13   for i:=u downto 1 do begin
14     write('->',Ut[i]);
15   end; writeln();
16 end;
```

2. Feladat: Szállítási feladat

Egy nagyvállalat n városban üzemeltet gyárat, a gyártmányokat k városban telepített raktárban tárolja. Minden városból olyan raktárba kell szállítani a gyártmányokat, amely útvonal a lehető legkevesebb városon halad át.
Adjunk olyan algoritmust, amely minden városra meghatározza a hozzá legközelebbi raktárt!

Bemenet

A standard bemenet első sora három egész számot tartalmaz, a városok n számát ($1 \leq n \leq 20000$), a raktárak k számát ($1 \leq k \leq n$), és a városok közötti közvetlen utak m ($1 \leq m \leq 100000$) számát. A következő m sor mindegyike két egész számot tartalmaz, egy-egy olyan város sorszámát, amelyek között van közvetlen kétirányú út.

Kimenet

A standard kimenet pontosan n sort tartalmazzon. Az i-edik sor az i-edik városba vezető legrövidebb raktárba vezető utat adja meg. Ha valamely városból nincs út raktárba, akkor a -1 számot kell kiírni a sorba. Több megoldás esetén bármelyik megadható.

Példa bemenet és kimenet

bemenet	kimenet
11 3 12 1 4 9 1 2 2 5 2 3 4 5 5 6 6 7 6 8 9 10 9 11 10 6 4 10 10 11	1 1 2 1 2 3 4 4 5 4 5 6 4 5 6 7 9 10 6 8 9 9 10 9 11

Legyen $S(0)$ a raktárak sorszámait tartalmazó halmaz.

$$S(t) = \{p \in V : (\exists q \in S(t-1))(q \to p \in E)\}$$

```
1  procedure SzeltBejar(N:integer; const G:Graf; K:integer;R:array integer;
2                       var Apa:array of integer);
3    var
4      S:Sor;
5      p,q:PontTip;  i:integer;
6    begin{SzeltBejar}
7      for p:=1 to N do Apa[p]:=-1;      { inicializálás}
8      Letesit(S);
9      for i:=1 to K do begin
10       Sorba(S,R[i]); Apa[R[i]]:=0;
11     end;
12     while Elemszam(S)>0 do begin
13       p:=SorBol(S);
14       for i:=1 To KiFok[p] Do Begin { p->q élre }
15         q:=G[p,i];
16         if Apa[q]=-1 then begin       { ha q meg nem elért }
17           Apa[q]:=p;
18           Sorba(S,q);
19         end;
20       end{for p->q};
21     end{while};
22   end{SzeltBejar};
```

```
23 begin
24   Beolvas(N,G,K,R);
25   SzeltBejar(N,G,K,R,Apa);
26   for i:=1 to N do begin
27     if Apa[i]=-1 then begin{nincs út i-ből raktárba}
28       writeln(-1);
29       continue;
30     end;
31     u:=0; p:=i;
32     while p<>0 do begin
33       inc(u);
34       Ut[u]:=p;
35       p:=Apa[p];
36     end;
37     for i:=u downto 1 do
38       write(Ut[i],' ');
39     writeln();
40   end{for i};
41 end;
```

3. Feladat: Térképszínezés

Az országban n régiót tartalmazó térképét kell kiszínezni két kölönböző színnel úgy, hogy bármely két szomszédos régió kölönböző színü legyen.

Bemenet

A standard bemenet első sora két egész számot tartalmaz, a régiók n számát ($1 \leq n \leq 20000$), és a szomszédos régiók m ($1 \leq m \leq 100000$) számát. A következő m sor mindegyike két különböző egész számot tartalmaz, egy-egy olyan régió sorszámát, amelyek szomszédosak.

Kimenet

A standard kimenet első és egyetlen sora azon régiók sorszámait tartalmazza (egy-egy szóközzel elválasztva), amelyek az egyik színnel lesznek színezve. Ha nincs megoldás, akkor az első sor a -1 számot tartalmazza. Több megoldás esetén bármelyik megadható.

Példa bemenet és kimenet

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

A probléma gráfmodellje: tekintsük azt a $G = (V, E)$ irányítatlan gráfot, amelynek pontjai a régiók, és két pont között pontosan akkor van él, ha a két régió szomszédos. Az nyilvánvaló, hogy akkor és csak akkor van megoldás, ha a V ponthalmaz felbontható két olyan A és B diszjunkt részhalmazra, hogy

$$A \cap B = \emptyset \text{ és } V = A \cup B$$

$$(\forall (p,q) \in E)(p \in A \text{ és } q \in B \text{ vagy } p \in B \text{ és } q \in A)$$

```
type
  Paletta=(Feher, Piros, Kek);
var
  Szin:array[1..maxP] of Paletta;
  Apa:array[1..maxP] of integer;
  p,q,i,j:integer;
  S:Sor;
  Van:boolean;
begin
  Beolvas;
  for p:=1 to N do
    Szin[p]:=Feher;
  Van:=true;
  Letesit(S);
```

```
15    for i:= 1 to N do
16      if Szin[i]=Feher then begin
17        Szin[i]:=Piros; Sorba(S,i);
18        while Elemszam(S)>0 do begin
19          p:=SorBol(S);
20          for j:=1 to KiFok[p] do begin
21            q:=G[p,j];
22            if Szin[p]=Szin[q] then begin
23              Van:=false; break;
24            end;
25            if Szin[q]=Feher then begin
26              if Szin[p]=Piros then
27                Szin[q]:=Kek
28              else
29                Szin[q]:=Piros;
30              Sorba(S,q);
31            end;
32          end;
33          if not Van then break;
34        end{while};
35    end{for i};
```

```
36    if not Van then begin
37      writeln(-1);
38    end else begin
39      for p:=1 to N do
40        if Szin[p]=Piros then
41          write(p,' ');
42      writeln;
43    end;
44  end.
```

4. Feladat: Adott ponton átmenő legrövidebb kör

Adott $G = (V, E)$ irányítatlan gráf és egy r pontja. Számítsuk ki az r ponton átmenő legrövidebb kört (ha van ilyen).

```
1  const
2    Inf=maxP;
3  var
4    Szin:array[1..maxP] of Paletta;
5    D,Apa,Os,Ut:array[1..maxP] of integer;
6    p,q,i,j:integer;
7    S:Sor;
8    p1,p2,mink:integer;
```

```
 9  begin
10    Beolvas;
11    for p:=1 to N do
12      D[p]:=Inf;
13    mink:=Inf;
14    p1:=0;
15    Letesit(S);
16    Sorba(S,R);
17    while Elemszam(S)>0 do begin
18      p:=SorBol(S);
19      for i:=1 to KiFok[p] do begin
20        q:=G[p,i];
21        if D[q]=Inf then begin
22          D[q]:=D[p]+1;
23          Apa[q]:=p;
24          if p=R then
25            Os[q]:=q
26          else
27            Os[q]:=Os[p];
28          Sorba(S,q);
29        end else if (Os[p]<>Os[q]) and (D[p]+D[q]<mink) then begin
30          p1:=p; p2:=q;
31          mink:=D[p]+D[q];
32        end;
33      end;
```

```
34      end{while};
```

```
35    if p1=0 then
36      writeln(-1)
37    else begin
38      i:=0;
39      while Apa[p1]<>0 do begin
40        inc(i);
41        Ut[i]:=p1;
42        p1:=Apa[p1];
43      end;
44      for i:=i downto 1 do
45        write(Ut[i],' ');
46      while (Apa[p2]>0) do begin
47        write(p2);
48        p2:=Apa[p2];
49      end;
50      writeln();
51    end;
52  end.
```

5. A mélységi bejárás alkalmazásai.

6. Feladat: Iskolahálózat (IOI'96)

Néhány iskola számítógépes hálózatba van kötve. Az iskolák a szabadon terjeszthető programok elosztására megállapodást kötöttek: minden iskola egy listát vezet azokról az iskolákról ("címzett iskolákról"), amelyek számára a programokat továbbküldi. Megjegyzés: ha **B** iskola szerepel az **A** iskola terjesztési listáján, akkor **A** nem feltétlenül szerepel **B** terjesztési listáján.

Írjon programot, amely meghatározza azt a legkisebb számot, ahány iskolához egy új programot el kell juttatni ahhoz, hogy - a megállapodás alapján, a hálózaton keresztül terjesztve - végül minden iskolába eljusson.

Bemenet

A bementi állomány első sora egy egész számot, a hálózatba kapcsolt iskolák n számát tartalmazza ($2 \leq n \leq 100$). Az iskolákat az első n pozitív egész számmal azonosítjuk. A következő n sor mindegyike egy-egy terjesztési listát ír le. Az $i+1$-edik sor az i-edik iskola terjesztési listáján szereplő iskolák azonosítóit tartalmazza. Minden lista végén egy 0 szám áll. Az üres lista csak egy 0 számból áll.

Kimenet

A kimenti állomány első sorába azt a legkisebb m számot kell írni, ahány iskolához egy új programot el kell juttatni ahhoz, hogy - a megállapodás alapján, a hálózaton keresztül terjesztve - végül minden iskolába eljusson. A második sor pontosan m egész számot tartalmazzon egy-egy szóközzel elválasztva, azon iskolák sorszámait, ahova kezdetben el kell juttatni az új programot.

Egy elvi megoldás

Jelölje $G = (V, E)$; $V = \{1, \ldots, n\}$ az iskolahálózat gráfját.

Legyen $D \subseteq V$ egy megoldáshalmaz, tehát

$$(\forall q \in V)(\exists p \in D)(p \rightsquigarrow q)$$

A D halmazt lépésenként építhetjük:
Adott $p \in V$ pontra jelölje $Eler(p)$ a p pontból elérhető pontok halmazát:

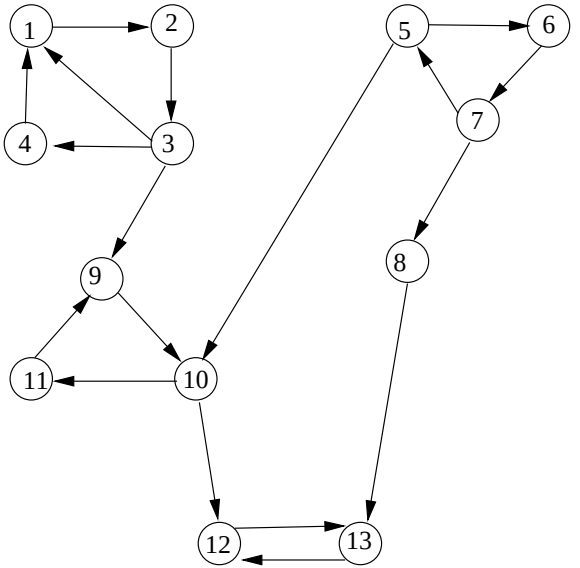
$$Eler(p) = \{q : p \rightsquigarrow q\}$$

Terjesszük ki ezt halmazokra:

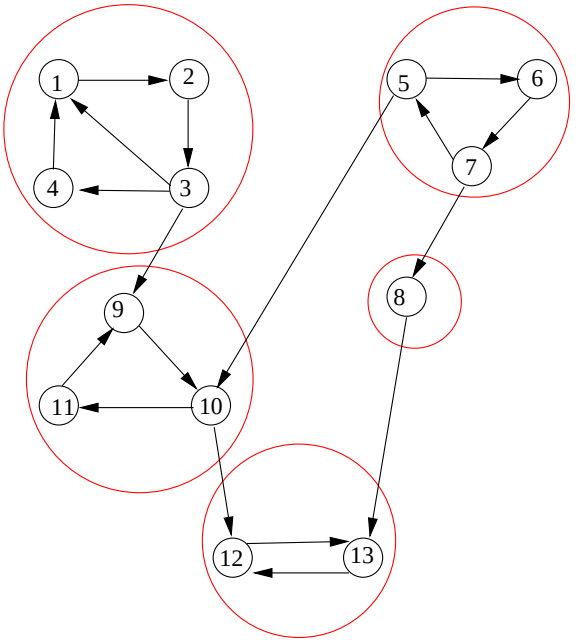
$$Eler(D) = \bigcup_{p \in D} Eler(p)$$

begin
  $D := \emptyset$;
  $DbolElert := \emptyset$;
  for $p \in V$ **do**
    if $p \notin DbolElert$ **then** **begin**
      $D := D - Eler(p) \bigcup \{p\}$
      $DbolElert := DbolElert \bigcup Eler(p)$
    end
end

Az $Eler(p)$ halmaz elemei mélységi bejárással kiszámíthatók.
Az algoritmus futási ideje legrosszabb esetben $O(n^3)$.
Az algoritmus legrosszabb esete, ha a bemenetben p szomszédai: $\{1, 2, \ldots, p-1\}$. Ha n 1000 is lehet, akkor ez az algoritmus biztosan nem elég gyors megoldás.



1. ábra.



2. ábra.

Gyorsítási ötlet: elsőször rakjuk a pontokat olyan sorrendbe, hogy ha $p \rightsquigarrow q$, de nincs $q \rightsquigarrow p$, akkor p előbb álljon a sorozatban, mint q.
Egy ilyen kívánt sorrendet egyetlen mélységi bejárással előállíthatunk:
a p pontra hívott (rekurzív) mélységi bejárásban, miután p-ből kiinduló összes élet bejártunk, rakjuk p-t a sorozat elejére. (Tehát a kívánt sorozatban a pontok elhagyási idő szerint csökkenő sorrendben lesznek.)
Az így módosított algoritmus két mélységi bejárást végez, tehát futási ideje $O(E)$

```
1   Const
2     MaxP=1000;                    { a pontok max. sz ma }
3   Type
4     PontTip=1..MaxP;
5     GrafTip=???;
6     Paletta=(Feher, Szurke, Fekete, Piros);
7   Var
8     N,u,M,p: Word;                      { a pontok száma}
9     G: GrafTip;
10    Szin:Array[PontTip] Of Paletta;
11    T,D : Array[PontTip] Of Word;
12  Procedure Beolvas; Begin End;
```

```
13 Procedure MelyBejar1(p:PontTip);
14 {Global: G, Szin, u,T}
15  Var
16     q:PontTip;
17 Begin{MelyBejar1}
18   Szin[p]:=Szurke;
19   For q In Ki(G,p) Do       { a p->q élet vizsgáljuk }
20     If Szin[q]=Feher Then   { q még nem érintett pont}
21       MelyBejar1(P);
22   Szin[p]:=Fekete;
23   T[u]:=p;
24   u:=u-1;
25 End{MelyBejar1};
26
27 Procedure MelyBejar2(p:PontTip);
28 {Global: G, Szin}
29  Var
30     q:PontTip;
31 Begin{MelyBejar2}
32   Szin[p]:=Piros;
33   For q In Ki(G,p) Do       { a p->q élet vizsgáljuk }
34     If Szin[q]=Fekete Then  { q még nem érintett pont}
35       MelyBejar2(P);
36 End{MelyBejar2};
```

```
37 Begin
38   Beolvas;
39   For p:=1 To N Do
40     Szin[p]:=Feher;
41   u:=N;
42   For p:=1 To N Do
43     If Szin[p]=Feher Then
44       MelyBejar1(p);
45   M:=0;{a megoldáshalmaz elemszáma}
46   for u:=1 to N do begin
47    p:=T[u];
48    if Szin[p]=Fekete then begin
49       inc(M);
50       D[M]:=p;       {p felvétele a dominátorba}
51       MelyBejar2(p);
52    end;
53   end;
54   writeln(M);        {a megoldás dominátor kiíratása}
55   for u:=1 to M do
56   write(D[u],' ');  writeln;
57 End.
```

7. Feladat: Minimálisan hány élet kell hozzáadni egy G irányított gráfhoz, hogy legyen olyan pontja, amelyből bármely másik elérhető?

8. Feladat: Összefüggő hálózat

Egy számítógépes hálózat csomópontokból és csomópont párokat összekötő egyirányú kommunikációt biztosító vonalakból épül fel. A hálózatot úgy tervezték, hogy bármely két csomópont között van átvitelt biztosító útvonal mindkét irányban. A hálózat két csomópontja közötti vonal meghibásodott. Adjunk meg két csomópont közötti új vonalat, amely biztosítja, hogy a hálózat ismét összefüggő legyen.

9. Feladat: Irányított gráf nem körben lévő pontjai

Adott $G = (V, E)$ irányított gráfra számítsuk ki az összes olyan pontját, amely nincs benne egyetlen körben sem.

```
1   Const
2     MaxP=1000;          { a pontok max. száma }
3   Type
4     PontTip=1..MaxP;
5     GrafTip=word;
6     Paletta=(Feher, Szurke, Fekete, Piros);
7   Var
8     N,u,M,p: Word;
9     G,GT: GrafTip;
10    Szin:Array[PontTip] Of Paletta;
11    T,ESzam : Array[PontTip] Of Word;
12  Procedure Beolvas; Begin
13  {Előállítja a GT transzponált gráfot is}
14  End;
```

```
15 Procedure MelyBejar1(p:PontTip);
16 {Global: G, Szin, T}
17  Var
18     q:PontTip;
19 Begin{MelyBejar}
20   Szin[p]:=Szurke;
21   For q In Ki(G,p) Do       { a p->q élet vizsgáljuk }
22     If Szin[q]=Feher Then   { q még nem érintett pont}
23       MelyBejar1(P);
24   Szin[p]:=Fekete;
25   T[u]:=p;
26   u:=u-1;
27 End{MelyBejar1};
28
29 Procedure MelyBejar2(os,p:PontTip);
30 {Global: GT, Szin}
31  Var
32     q:PontTip;
33 Begin{MelyBejar}
34   Szin[p]:=Piros;
35   inc(ESzam[os]);
36   For q In Ki(GT,p) Do       { a p->q élet vizsgáljuk }
37     If Szin[q]=Fekete Then   { q még nem érintett pont}
38       MelyBejar2(os,P);
39 End{MelyBejar2};
```

```
40 Begin
41   Beolvas;
42   For p:=1 To N Do begin
43     Szin[p]:=Feher;
44     ESzam[p]:=0;
45   end;
46   u:=N;
47   For p:=1 To N Do
48     If Szin[p]=Feher Then
49       MelyBejar1(p);
50   M:=0;
51   for u:=1 to N do begin
52    p:=T[u];
53    if Szin[p]=Fekete then
54       MelyBejar2(p,p);
55   end;
56
57   for p:=1 to N do
58    if ESzam[p]=1 then
59      write(p,'␣'); writeln;
60 End.
```

10. Feladat: Irányítatlan gráf nem körben lévő pontjai

Adott $G = (V, E)$ irányítatalan gráfra számítsuk ki az összes olyan pontját, amely nincs benne egyetlen körben sem.

```
1  Procedure MelyBejar(p:PontTip);
2  {Global: G, Szin, Apa, Korben}
3    Var
4      q,r:PontTip;
5  Begin{MelyBejar}
6    Szin[p]:=Szurke;
7    For q In Ki(G,p) Do              { a p->q élet vizsgáljuk }
8      If Szin[q]=Feher Then Begin    { q még nem érintett pont}
9        Apa[q]:=p;
10       MelyBejar(P);
11     End Else If (Apa[p]<>q)and(Szin[q]=Szurke) Then Begin
12       r:=p;
13       While r<>q Do Begin
14         Korben[r]:=true;
15         r:=Apa[r];
16       End;
17       Korben[q]:=True;
18     End{while};
19   Szin[p]:=Fekete;
20 End{MelyBejar};
```

```
21 Begin
22   Beolvas;
23   For p:=1 To N Do Begin
24     Szin[p]:=Feher;
25     Korben[p]:=False;
26   End;
27   For p:=1 To N Do
28     If Szin[p]=Feher Then
29       MelyBejar(p);
30
31   for p:=1 to N do
32    if not Korben[p] then
33     write(p,' '); writeln;
34 End.
```

Megjegyezzük, hogy a fenti algoritmus nem hatékony, legrosszabb esetben a pontok számával négyzetesen arányos a futási idő. Van olyan módszer, amely az élek számával arányos futási időt ad.

11. Legrövidebb utak minden pontpárra (Floyd-Warshall algoritmus)

Input: $G = (V, E, c)$ irányított súlyozott gráf; $c : E \to \mathbb{R}^+$
Output:Minden u, v pontpárra $\delta(u, v)$ és egy $u \rightsquigarrow v$ legrövidebb út.

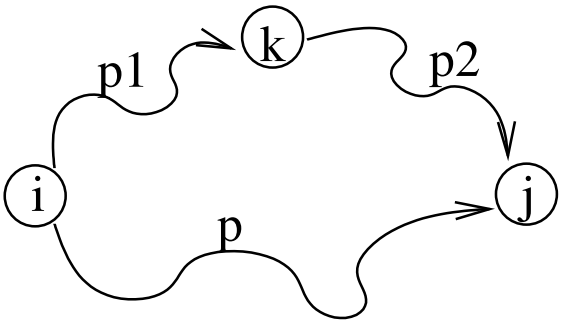
Megoldás dinamikus programozás módszerével.
Tfh. $V = 1..n$ és $c(i, j) = \infty$, ha $(i, j) \notin E$, azaz $G[i, j] = c(i, j)$ és $G[i, i] = 0, i = 1, \cdots, n$

Részproblémákra bontás:
$\forall i, j \in \{1, \ldots, n\}$-ra és $\forall k \in \{0, 1, \cdots, n\}$-re
$D^k(i, j) =$ az i-ből j-be vezető olyan utak hosszának minimuma, amelyek legfeljebb az $\{1, \cdots, k\}$ (belső) pontokon mennek keresztül.
$D^n(i, j) = \delta(i, j)$

Rekurzív összefüggés a részproblémák között.

$$\begin{aligned} D^0(i,j) &= G[i,j] \\ D^k(i,j) &= Min\{D^{k-1}(i,j), D^{k-1}(i,k) + D^{k-1}(k,j)\} \quad k > 0 \end{aligned}$$

$D[i,j]$ helyben számolása; egy D tömbben.



3. ábra. $p : i \rightsquigarrow j$ az {1..k-1} pontokon át haladó út, $p_1 : i \rightsquigarrow k$, $p_2 : k \rightsquigarrow j$ {1..k-1} pontokon át haladó legrövidebb út.

A k-adik iterációban felülírt elemek: $D^{k-1}(i,k)$ és $D^{k-1}(k,j)$.
De

$$D^k(i,k) = Min\{D^{k-1}(i,k), D^{k-1}(i,k) + D^{k-1}(k,k)\} = D^{k-1}(i,k)$$

```
1   for i:=1 to N do
2     for i:=1 to N do begin
3       D[i,j]:=G[i,j];
4       Elso[i,j]:=j;
5     end;
6   for k:=1 to N do
7     for i:=1 to N do
8       for j:=1 to N do begin
9         Dikj:=D[i,k]+D[k,j];
10        if Dikj<D[i,j] then begin
11          D[i,j]:=Dikj;
12          Elso[i,j]:=Elso[i,k];
13        end;
```

p-ből q-ba vezető legrövidebb út kiíratása:

```
1   if D[p,q]<>Inf then begin{van út p-ből q-ba}
2     repeat
3       write(p,'->');
4       p:=Elso[p,q];
5     until p=q;
6     writeln(q);
7   end;
```