# A

**1a. feladat:** **10**

*Adott egy összefüggő, nem irányított, súlyozott (>0) G gráf. Határozza meg azt a pontját, amelytől a **legkisebb távolság**ra vannak a gráf többi pontjai. Azaz*

$$p_s \in \text{Pontok}(G): \sum_{\substack{p \in \text{Pontok}(G) \\ p \neq p_s}} \text{Táv}(p_s, p) = \text{Min}\{ \sum_{\substack{p \in \text{Pontok}(G) \\ p \neq q}} \text{Táv}(q, p) \mid q \in \text{Pontok}(G)\}.$$

*Az algoritmusa **ábrázolás-független** legyen!*

**2a. feladat:** **7**

*Adott egy súlyozott (>0), irányított G gráf (*`TPont` *típusú pontokkal). Keresse meg az összes, „**nyelő**” tulajdonságú csúcsát, és adja meg –**nyelőnként**– a „belefolyó” össz-mennyiséget! Egy csúcs akkor nyelő tulajdonságú, ha nem izolált és Ki-foka 0. Az algoritmusa **ábrázolás-független** legyen!*

**3a. feladat:** **15**

*Adott egy város úthálózata (zsák-, egy- és kétirányú utcákkal). Az utcák **hossza** és **teherbírása** ismert.*

*α) Határozzuk meg a legrövidebb utat, amely egy adott súlyú teherautóval a város két rögzített pontja között végig autózható!*

*β) Maximum mekkora súlyú teherautóval lehet eljutni a kezdőpontból a végpontba?*

*A teherautó súlya (*`súly:Egész`*) és a két gráfpont az algoritmus paraméterei (*`tól,ig:TPont`*). A gráf* `TElem` *típusa, azaz a pontban tárolt érték típusa –a fentiekből következőleg–:*
**`Rekord`**`(hossz,maxSúly:Egész)`. *Az algoritmusa ábrázolás-független legyen!*

**4a. feladat:** **9**

*A nem irányított, súlyozatlan gráf ábrázolását az alábbi típusok írják le:*

```
TPont = Szöveg
TPontok = Lista(TPont) [növekvően rendezve]
TÉlek = Rekord(ból:TPont,ba:TPontok)
TGráf = Rekord(pontok:TPontok
               élek:Lista(TÉlek) [„ból”-szerint növekvően rendezve]
               hiba:Logikai)
```

*Implementálja a következő műveleteket a fenti ábrázolás mellett:*

```
Eljárás BeillesztPont(Változó g:TGráf, Konstans p:TPont)
  [egy új pontot beilleszt, izoláltként]
Eljárás Elszakít(Változó g:TGráf, Konstans p,q:TPont)
  [a p és q pontok közötti élet megszünteti]
```

*Ügyeljen arra is, hogy az egyes műveletek végrehajtásának feltételei vannak! Ha ez nem teljesül, az hibát jelent; így a műveletet végrehajtani nem szabad, és a hibát jelezni kell (l. hiba-flag)!*

**Tervezett ponthatárok:**

| | | | |
|---|---|---|---|
| *Kettes, ha legalább:* | *15* | *Négyes, ha legalább:* | *26* |
| *Hármas, ha legalább:* | *20* | *Ötös, ha legalább:* | *31* |
| | *Maximum:* | *10+7+15+9=41* | |

# $\mathcal{B}$

**1b. feladat:** **10**

*Adott egy összefüggő, nem irányított G gráf. Határozza meg azt a pontját, amelytől a* ***legkevesebb lépés****re vannak a gráf többi pontjai. Azaz*

$$p_s \in \text{Pontok}(G): \sum_{\substack{p\in \text{Pontok}(G)\\ p\neq p_s}} \text{LépésSzám}(p_s, p) = \text{Min}\ \{ \sum_{\substack{p\in \text{Pontok}(G)\\ p\neq q}} \text{LépésSzám}(q, p)\ |\ q\in \text{Pontok}(G)\}.$$

*Az algoritmusa* ***ábrázolás-független*** *legyen!*

**2b. feladat:** **7**

*Adott egy súlyozott (>0), irányított gráf (*`TPont` *típusú csúcsokkal). Keresse meg az összes, „****forrás****” tulajdonságú csúcsát, és adja meg a „belőlük* ***összesen*** *kifolyó” mennyiséget! Egy csúcs akkor forrás tulajdonságú, ha nem izolált és Be-foka 0. Az algoritmusa* ***ábrázolás-független*** *legyen!*

**3b. feladat:** **15**

*Adott egy város úthálózata (zsák-, egy- és kétirányú utcákkal). Az utcák* ***hossza*** *és pillanatnyi* ***túlterheltsége*** *ismert. Túlterheltség alatt értsük azt a szorzószámot, amennyiszer többet igényel a pillanatnyi átjutás az akadálymenteshez képest.*

*α) Adjunk meg azt a legrövidebb utat, amely a város két rögzített pontja között vezet, és csak egy előre adott korlátot meg nem haladó túlterheltségű útszakaszokon vezet!*

*β) Mennyire csökkenthető a korlát, hogy figyelembe vételével még el lehessen jutni a kezdőpontból a végpontba?*

*A túlterheltségi korlát (*`ttk:Valós`*) és a két gráfpont az algoritmus paraméterei (*`tól,ig: TPont`*). A gráf* `TElem` *típusa, azaz a pontban tárolt érték típusa –a fentiekből következőleg–:* `**Rekord**(hossz,túlTerh:Valós)`*. Az algoritmusa ábrázolás-független legyen!*

**4b. feladat:** **9**

*A nem irányított, súlyozatlan gráf ábrázolását az alábbi típusok írják le:*

```
TPont = Szöveg
TPontok = Lista(TPont) [csökkenően rendezve]
TÉlek = Rekord(ból:TPont,ba:TPontok)
TGráf = Rekord(pontok:TPontok
               élek:Lista(TÉlek) [„ból”-szerint csökkenően rendezve]
               hiba:Logikai)
```

*Implementálja a következő műveleteket a fenti ábrázolás mellett:*

```
Eljárás TörölPont(Változó g:TGráf, Konstans p:TPont)
  [egy izolált pontot kivesz a gráfból]
Eljárás Összeköt(Változó g:TGráf, Konstans p,q:TPont)
  [a p és q pontokat éllel kapcsolja össze, ha még nem voltak…]
```

*Ügyeljen arra is, hogy az egyes műveletek végrehajtásának feltételei vannak! Ha ez nem teljesül, az hibát jelent; így a műveletet végrehajtani nem szabad, és a hibát jelezni kell (l. hiba-flag)!*

**Tervezett ponthatárok:**

| | | | |
|---|---|---|---|
| *Kettes, ha legalább:* | *15* | *Négyes, ha legalább:* | *26* |
| *Hármas, ha legalább:* | *20* | *Ötös, ha legalább:* | *31* |
| | *Maximum:* | *10+7+15+9=41* | |

# *Megoldás*

## 1a. feladat: 10

*Adott egy összefüggő, nem irányított, súlyozott (>0) G gráf. Határozza meg azt a pontját, amelytől a* ***legkisebb távolság****ra vannak a gráf többi pontjai. Azaz*

$$p_s \in \text{Pontok(G)}: \sum_{\substack{p\in \text{Pontok(G)}\\ p\neq p_s}} \text{Táv}(p_s, p) = \text{Min}\ \{ \sum_{\substack{p\in \text{Pontok(G)}\\ p\neq q}} \text{Táv}(q, p)\ |\ q\in \text{Pontok(G)}\}.$$

*Az algoritmusa* ***ábrázolás-független*** *legyen!*

## 1. megoldás:

*A távolság-mátrixra és a Warshall algoritmusra építve:*

```
Típus
  TValósMátrix = Tömb(1..MaxN,1..MaxN:Valós)
  TValósVektor = Tömb(1..MaxN:Valós)
Függvény SúlyPont(Konstans g:TGráf):TPont                              (3 pont)
  Változó
    CsúcsMát,TavMát:TValósMátrix
    összTáv:TValósVektor
    N:Egész
  N:=PontSzám(g)
  CsúcsMát:=CsúcsMátrixGen(g,N)
  TávMát:=Warshall(CsúcsMát,N)
  összTáv:=ÖsszTávGen(TávMát,N)
  súlyPont:=MinKer(összTáv,N)
Függvény vége.
Függvény CsúcsMátrixGen(Konstans g:TGráf, N:Egész):TValósMátrix        (2 pont)
  Változó
    i,j:Egész
    tm:TValósMátrix
  tm(1..N,1..N):=+∞
  Ciklus i=1-től N-ig
    Ciklus j=1-től N-ig
      Ha VanÉl?(g,Pont(i),Pont(j)) akkor tm(i,j):=ÉlHossz(g,Pont(i),Pont(j))
    Ciklus vége
  Ciklus vége
  CsúcsMátrixGen:=tm
Függvény vége.
Függvény Warshall(Konstans csm:TValósMátrix, N:Egész):TValósMátrix     (1 pont)
  [1. jegyzet]
Függvény vége.
Függvény ÖsszTávGen(Konstans tm:TValósMátrix, N:Egész):TValósVektor    (2 pont)
  Változó
    i,j:Egész
    s:Valós
    öt:TValósVektor
  Ciklus i=1-től N-ig
    s:=0
    Ciklus j=1-től N-ig
      s:+tm(i,j)
    Ciklus vége
    öt(i):=s
  Ciklus vége
  ÖsszTávGen:=öt
Függvény vége.
```

```
Függvény MinKer(Konstans ötv:TValósVektor, N:Egész):TPont                (2 pont)
  Változó
    i,mini:Egész
    min:Valós
  Min:=ötv(1); mini:=1
  Ciklus i=2-től N-ig
    Ha ötv(i)>min akkor min:=ötv(i); mini:=i
  Ciklus vége
  MinKerIndex:=Pont(mini)
Függvény vége.
```

**2. megoldás:**

*A bejárásra, ill. a LegrövidebbÚt(ból,ba) –módosított–, vagy a LegrövidebbUtak Dijkstra, ill. Bellmann-Ford algoritmusára építve:*

```
Típus
  TValósVektor = Tömb(1..MaxN:Valós)
Függvény SúlyPont(Konstans g:TGráf):TPont                                 (3 pont)
  Változó
    öTáv,Táv:TValósVektor
    N:Egész
  N:=PontSzám(g)
  Ciklus i=1-től N-ig
    Táv:=LegrövidebbUtak(g,Pont(i)) [i-ből kiinduló utak hossza]
    őTáv(i):=0
    Ciklus j=1-től N-ig
      őTáv(i):+Táv(j)
    Ciklus vége
  Ciklus vége
  súlyPont:=MinKer(Táv,N)
Függvény vége.
Függvény LegrövidebbUtak(Konstans g:TGráf, p:TPont):TValósVektor         (5 pont[⊗])
  [l. jegyzet: Dijkstra v. Bellmann-Ford algoritmust]
Függvény vége.
Függvény MinKer(Konstans ötv:TValósVektor, N:Egész):TPont                (2 pont)
  Változó
    i,mini:Egész
    min:Valós
  Min:=ötv(1); mini:=1
  Ciklus i=2-től N-ig
    Ha ötv(i)>min akkor min:=ötv(i); mini:=i
  Ciklus vége
  MinKerIndex:=Pont(mini)
Függvény vége.
```

Ha i-vel közvetlenül, s nem a `Pont` függvény közvetítésével indexel, akkor hiszen felteszi, hogy `TPont=Egész` ami az árbázolás korlátozását jelenti. `-1 pont`

**1b. feladat:** **10**

*Adott egy összefüggő, nem irányított G gráf. Határozza meg azt a pontját, amelytől a **legkevesebb lépésre** vannak a gráf többi pontjai. Azaz*

$$p_s \in \text{Pontok(G)}: \sum_{\substack{p \in \text{Pontok(G)} \\ p \neq p_s}} \text{LépésSzám}(p_s, p) = \text{Min} \{ \sum_{\substack{p \in \text{Pontok(G)} \\ p \neq q}} \text{LépésSzám}(q, p) \mid q \in \text{Pontok(G)} \}.$$

*Az algoritmusa **ábrázolás-független** legyen!*

**1. megoldás:**

*A távolság-mátrixra és a Warshall algoritmusra építve a létező éleket 1 hosszúságúnak véve: l. az 1a./1. megoldást.*

---

[⊗] Csak akkor jár a pont, ha érdemi egyéb is van.

**2. megoldás:**

*A bejárásra, ill. a LegrövidebbÚt(ból,ba) –módosított–, vagy a LegrövidebbUtak Dijkstra, ill. Bellmann-Ford algoritmusára építve a létező éleket 1 hosszúságúnak véve:*
*l. az 1a./1. megoldást.*

**2a. feladat:** 7

*Adott egy súlyozott (>0), irányított G gráf (*`TPont` *típusú pontokkal). Keresse meg az összes, „**nyelő**” tulajdonságú csúcsát, és adja meg –**nyelőnként**– a „belefolyó” össz-mennyiséget! Egy csúcs akkor nyelő tulajdonságú, ha nem izolált és Ki-foka 0. Az algoritmusa **ábrázolás-független** legyen!*

**Megoldás:**

```
Típus
  TPontok = Tömb(1..MaxN:Rekord(pont:TPont,menny:Valós))
Eljárás Nyelők(Konstans g:TGráf,
               Változó ny:TPontok, db:Egész):                (3 pont)
  Változó
    i:Egész
    bele:Valós
  db:=0
  Ciklus i=1-től PontSzam(g)-ig
    Ha SzomszédPontokSzáma(g,Pont(i))=0 [≈KiFok(g,Pont(i))=0] akkor
      bele:=0
      Ciklus j=1-től PontSzám(g)-ig
        Ha VanÉl?(g,Pont(j),Pont(i)) akkor
           bele:+ÉlHossz(g,Pont(j),Pont(i))
      Ciklus vége                                            (2 pont)
      Ha menny>0 akkor [nem izolált pont ⇒ forrás]
        db:+1; ny(db).pont:=Pont(i)                          (2 pont)
        ny(db).menny:=bele
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

Ha i-vel közvetlenül, s nem a `Pont` függvény közvetítésével indexel, akkor hiszen felteszi, hogy `TPont=Egész` ami az árbázolás korlátozását jelenti. `-1 pont`

**2b. feladat:** 7

*Adott egy súlyozott (>0), irányított gráf (*`TPont` *típusú csúcsokkal). Keresse meg az összes, „**forrás**” tulajdonságú csúcsát, és adja meg a „belőlük **összesen** kifolyó” mennyiséget! Egy csúcs akkor forrás tulajdonságú, ha nem izolált és Be-foka 0. Az algoritmusa **ábrázolás-független** legyen!*

**Megoldás:**

*A 2a. megoldásban szereplő algoritmus értelemszerűen írható át a SzomszédPontokSzáma helyett a BeFok függvényt használva, és természetesen a TPontok típust egyszerűsítve, az összegzést „összesítve”...*

```
Típus
  TPontok = Tömb(1..MaxN:TPont)
Eljárás Források(Konstans g:TGráf,
                 Változó fo:TPontok, db:Egész, menny:Valós):  (3 pont)
  Változó
    i:Egész
    belőle:Valós
  db:=0; menny:=0
  Ciklus i=1-től PontSzam(g)-ig
    Ha BeFok(g,Pont(i))=0 akkor [lehet, hogy forrás]
      belőle:=0
      Ciklus j=1-től PontSzám(g)-ig
        Ha VanÉl?(g,Pont(i),Pont(j)) akkor
           belőle:+ÉlHossz(g,Pont(i),Pont(j))
      Ciklus vége                                            (2 pont)
```

```
      Ha belőle>0 akkor [nem izolált pont ⇒ forrás]
        db:+1; ny(db):=Pont(i)                                    (2 pont)
        menny:+belőle
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

Ha i-vel közvetlenül, s nem a `Pont` függvény közvetítésével indexel, akkor hiszen felteszi, hogy `TPont=Egész` ami az árbázolás korlátozását jelenti. `-1 pont`

*BeFok nélkül:*

```
Típus
  TPontok = Tömb(1..MaxN:TPont)
Eljárás Források(Konstans g:TGráf,
                 Változó fo:TPontok, db:Egész, menny:Valós):      (3 pont)
  Változó
    i:Egész
    belőle:Valós
    bele:Logikai
  db:=0; menny:=0
  Ciklus i=1-től PontSzam(g)-ig
    Ha SzomszédPontokSzáma(g,Pont(i))>0 akkor [lehet, hogy forrás]
      bele:=Hamis
      belőle:=0; j=1
      Ciklus j≤PontSzám(g) és nem bele
        Ha VanÉl?(g,Pont(i),Pont(j)) akkor
          belőle:+ÉlHossz(g,Pont(i),Pont(j))
        bele:=VanÉl?(g,Pont(j),Pont(i))
        j:+1
      Ciklus vége                                                 (2 pont)
      Ha nem bele akkor [nincs bele futó él ⇒ forrás]
        menny:+belőle
        db:+1; fo(db):=Pont(i)                                    (2 pont)
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

Ha i-vel közvetlenül, s nem a `Pont` függvény közvetítésével indexel, akkor hiszen felteszi, hogy `TPont=Egész` ami az árbázolás korlátozását jelenti. `-1 pont`

**3a. feladat:** **7+8**

*Adott egy város úthálózata (zsák-, egy- és kétirányú utcákkal). Az utcák **hossza** és **teherbírása** ismert.*

*α) Határozzuk meg a legrövidebb utat, amely egy adott súlyú teherautóval a város két rögzített pontja között végig autózható!*

*β) Maximum mekkora súlyú teherautóval lehet eljutni a kezdőpontból a végpontba?*

*A teherautó súlya (*`súly:Egész`*) és a két gráfpont az algoritmus paraméterei (*`tól,ig:TPont`*). A gráf* `THossz` *típusa, azaz az élhez tárolt érték típusa –a fentiekből következőleg–:* **`Rekord`**`(hossz,maxSúly:Egész)`. *Az algoritmusa ábrázolás-független legyen!*

**Megoldás:**

*α) A* `LegrövidebbÚt(g,x,y)` *eljárás olyan módosítása, amelyben* `(7 pont)`

*(i) a* `THossz` *típusú* `ÉlHossz` *függvény értéke kétkomponensű, továbbá*

*(ii) a „*`VanÉl?(pont,i)`*" feltétel kibővül az „*`és súly≤ÉlHossz(g,pont,i).maxSúly`*" feltétellel, és az „*`ÉlHossz(pont,i)`*" minden előfordulása helyett az „*`ÉlHossz(pont,i).hossz`*" szerepel.*

*β) Ötlet:* `(8 pont)`

*(i) élek maxSúly-szerinti csökkenő (v. növekvő) sorrendbe rendezése,*

*(ii) a teherautó súlyára válasszuk az addig még nem választott legnagyobb (v. legkisebb) maxSúlyú él maxSúly értékét, és ezzel járjunk el α szerint,*

*(iii) ha (nem) található (legrövidebb) út a két végpont között, akkor folytassuk (ii)-vel, különben megvan a keresett maximális súly.*

*Lehetséges lenne az élek súly-tömbjében a megfelelőt logaritmikusan keresni, ami hatékonyabbá tehetné a fenti algoritmust.*

**3b. feladat:** **7+8**

*Adott egy város úthálózata (zsák-, egy- és kétirányú utcákkal). Az utcák **hossza** és pillanatnyi **túlterheltsége** ismert. Túlterheltség alatt értsük azt a szorzószámot, amennyiszer többet igényel a pillanatnyi átjutás az akadálymenteshez képest.*

*α) Adjunk meg azt a legrövidebb utat, amely a város két rögzített pontja között vezet, és csak egy előre adott korlátot meg nem haladó túlterheltségű útszakaszokon vezet!*

*β) Mennyire csökkenthető a korlát, hogy figyelembe vételével még el lehessen jutni a kezdőpontból a végpontba?*

*A túlterheltségi korlát (*`ttk:Valós`*) és a két gráfpont az algoritmus paraméterei (*`tól,ig:TPont`*). A gráf* `TElem` *típusa, azaz a pontban tárolt érték típusa –a fentiekből következőleg–:* **`Rekord`**`(hossz,túlTerh:Valós)`*. Az algoritmusa ábrázolás-független legyen!*

**Megoldás:**

*Előzőhöz hasonlóan…*

**4a. feladat:** **9**

*A nem irányított, súlyozatlan gráf ábrázolását az alábbi típusok írják le:*

```
TPont = Szöveg
TPontok = Lista(TPont) [növekvően rendezve]
TÉlek = Rekord(ból:TPont,ba:TPontok)
TGráf = Rekord(pontok:TPontok
               élek:Lista(TÉlek) [„ból”-szerint növekvően rendezve]
               hiba:Logikai)
```

*Implementálja a következő műveleteket a fenti ábrázolás mellett:*

```
Eljárás BeillesztPont(Változó g:TGráf, Konstans p:TPont)
  [egy új pontot beilleszt, izoláltként]
Eljárás Elszakít(Változó g:TGráf, Konstans p,q:TPont)
  [a p és q pontok közötti élet megszünteti]
```

*Ügyeljen arra is, hogy az egyes műveletek végrehajtásának feltételei vannak! Ha ez nem teljesül, az hibát jelent; így a műveletet végrehajtani nem szabad, és a hibát jelezni kell (l. hiba-flag)!*

**Megoldás:**

*A Benne? Függvény egy absztrakt típus-művelet az alábbi reláció operátorokkal:*

```
Operátor Infix Kisebb(
         Konstans x,y:TPont):Logikai
  Másként x<y
  Kisebb:=x<y
Operátor vége.
```

```
Operátor Infix Kisebb(
         Konstans x,y:TÉlek):Logikai
  Másként x<y
  Kisebb:=x.ból<y.ból vagy
          x.ból=y.ból és
          Benne?(Első(y.ba),x.ba)
Operátor vége.
```

```
Operátor Infix Azonos(
         Konstans x,y:TPont):Logikai
  Másként x=y
  Azonos:=x=y
Operátor vége.
```

```
Operátor Infix Azonos(
         Konstans x,y:TÉlek):Logikai
  Másként x=y
  Azonos:=x.ból=y.ból és
          Benne?(Első(y.ba),x.ba)
Operátor vége.
```

```
Függvény Benne?(Változó l:Lista(TElem), Konstans e:TElem):Logikai      (3 pont)
  Ha Üres(l) akkor
    Benne?:=Hamis
  különben
    Elejére(l)
    Ciklus amíg nem Utolsó?(l) és e<ElemÉrték(l)
```

```
      Következőre(l)
    Ciklus vége
    Benne?:=e=ElemÉrték(l)
  Elágazás vége
Függvény vége.

Eljárás BeillesztPont(Változó g:TGráf, Konstans p:TPont):                (Σ5 pont)
  [egy új pontot beilleszt, izoláltként]
  Ha nem Benne?(g.pontok,p) akkor
    Ha nem Üres?(g.pontok) akkor                                          (3 pont)
      [feltéve: a Benne? a p helye környékére állította az akt.elemet]
      Ha p<ElemÉrték(l) akkor
        Ciklus amíg nem Első?(g.pontok) és p<ElemÉrték(g.pontok)
          Előzőre(g.pontok)
        Ciklus vége
      különben
        Ciklus amíg nem Utolsó?(g.pontok) és p>ElemÉrték(g.pontok)
          Következőre(g.pontok)
        Ciklus vége
      Elágazás vége
      [megvan a helye]
      Ha p<ElemÉrték(g.pontok) akkor
        BeszútElé(g.pontok,p)
      különben
        BeszútMögé(g.pontok,p)
      Elágazás vége
    különben [beszúrás elsőként]                                          (1 pont)
      BeszúrElé(g.pontok,p)
    Elágazás vége
  különben [van már ilyen ⇒ hiba]                                         (1 pont)
    g.hiba:=Igaz
  Elágazás vége
Eljárás vége.

Eljárás Elszakít(Változó g:TGráf, Konstans p,q:TPont):                    (Σ4 pont)
  [a p és q pontok közötti élet megszünteti]
  Változó
    él:TÉlek
  él.ból:=p; Üres(él.ba); BeszúrElé(él.ba,q)
  Ha Benne?(g.pontok,p) és Benne?(g.pontok,q) és
     Benne?(g.élek,él) akkor
    [feltéve: a Benne? a (p,q) él helye környékére állította az akt.elemet]
    Ha él<ElemÉrték(l) akkor                                              (3 pont)
      Ciklus amíg él<ElemÉrték(g.élek)
        Előzőre(g.élek)
      Ciklus vége
    különben
      Ciklus amíg él>ElemÉrték(g.élek)
        Következőre(g.élek)
      Ciklus vége
    Elágazás vége
    [megvan a helye]
    Kihagy(g.élek)
  különben [ilyen él nincsen]                                             (1 pont)
    g.hiba:=Igaz
  Elágazás vége
Eljárás vége.
```

*A szokásos jelölés szerint a* `g` *komponenseire a „*`g.`*” kiírása nélkül is szabad hivatkozni.*

**4b. feladat:** **9**

*A nem irányított, súlyozatlan gráf ábrázolását az alábbi típusok írják le:*

```
TPont = Szöveg
TPontok = Lista(TPont) [csökkenően rendezve]
```

```
TÉlek = Rekord(ból:TPont,ba:TPontok)
TGráf = Rekord(pontok:TPontok
               élek:Lista(TÉlek) [„ból”-szerint csökkenően rendezve]
               hiba:Logikai)
```

*Implementálja a következő műveleteket a fenti ábrázolás mellett:*

```
Eljárás TörölPont(Változó g:TGráf, Konstans p:TPont)
  [egy izolált pontot kivesz a gráfból]
Eljárás Összeköt(Változó g:TGráf, Konstans p,q:TPont)
  [a p és q pontokat éllel kapcsolja össze, ha még nem voltak…]
```

*Ügyeljen arra is, hogy az egyes műveletek végrehajtásának feltételei vannak! Ha ez nem teljesül, az hibát jelent; így a műveletet végrehajtani nem szabad, és a hibát jelezni kell (l. hiba-flag)!*

**Megoldás:**

*A 4a. megoldás mintájára.*