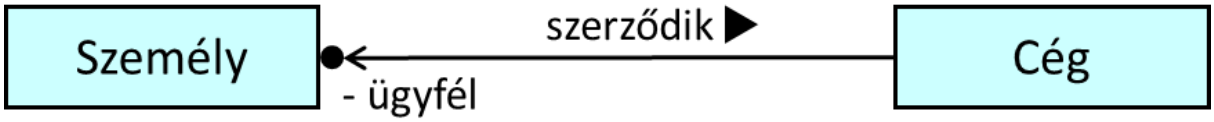
1. kérdés

1 / 1 pont

Egy biztosításokkal foglalkozó cég és egy személy között akkor jön létre kapcsolat, amikor a személy biztosítást köt a céggel.



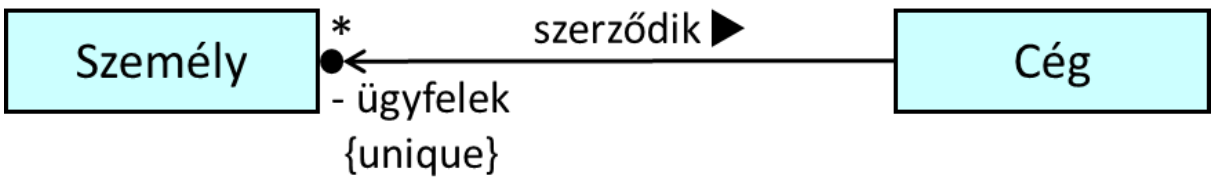
Mi a módja e kapcsolat felépítésének?

- A *Cég* osztály egy metódusa kapja meg paraméterként az ügyfél hivatkozását, amit ez a metódus csak akkor tárol el az *ügyfelek* nevű gyűjteményben, ha az még nem szerepel ott.
- A *Cég* osztály konstruktora kapja meg paraméterként a szerződést kötő személy hivatkozását, amit az *ügyfél* nevű adattagjában tárol el.
- A *Személy* osztály egy metódusa kapja meg paraméterként a cég hivatkozását, amit a saját *ügyfél* nevű adattagjában tárol el.
- A *Személy* osztály konstruktora kapja meg paraméterként a cég hivatkozását, amit az ügynök *ügyfél* nevű adattagjában tárol el.

2. kérdés

1 / 1 pont

Egy biztosításokkal foglalkozó cég és egy személy között akkor jön létre kapcsolat, amikor a személy biztosítást köt a céggel. Ilyenkor a cégnek már lehetnek korábban szerződött ügyfelei.



Mi a módja egy újabb személlyel történő kapcsolat felépítésének?

A *Cég* osztály egy metódusa kapja meg paraméterként az ügyfél hivatkozását, amit ez a metódus csak akkor tárol el az *ügyfelek* nevű gyűjteményben, ha az még nem szerepel ott.

A *Cég* osztály egy metódusa kapja meg paraméterként az ügyfél hivatkozását, amit ez a metódus hozzáad a cég *ügyfelek* nevű gyűjteményben tárolt ügyfelekhez

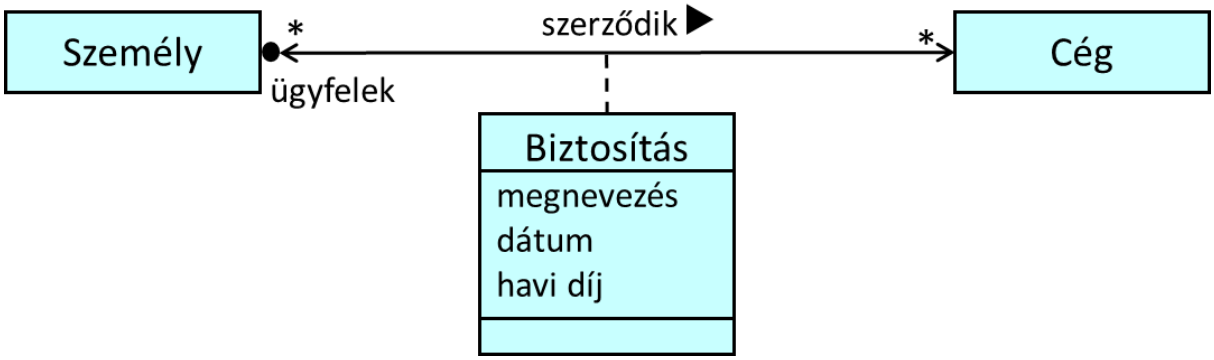
A *Személy* osztály egy metódusa kapja meg paraméterként a cég hivatkozását, és meghívja a *Cég* osztálynak azt a metódusát, amely hozzáadja a személy hivatkozását a cég *ügyfelek* nevű gyűjteményben tárolt ügyfelekhez.

A *Személy* osztály egy metódusa kapja meg paraméterként a cég hivatkozását, hogy azt az *ügyfelek* nevű gyűjteményébe elhelyezze – feltéve, hogy még nem szerepel ott.

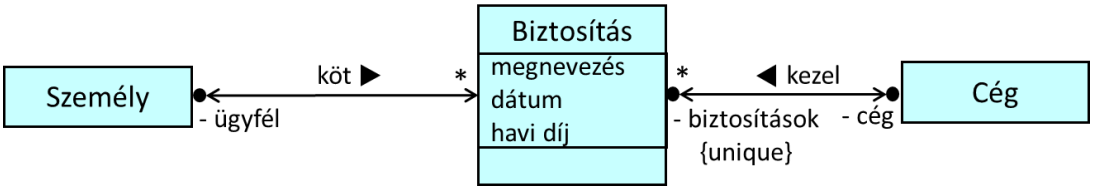
3. kérdés

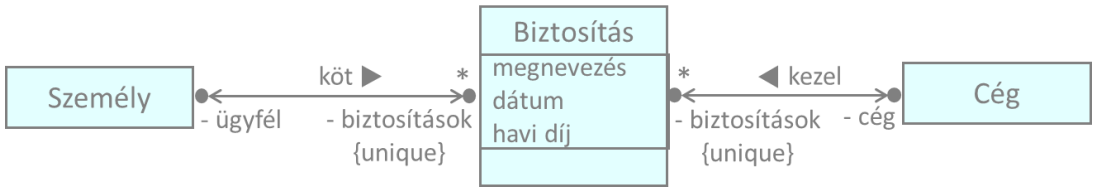
1 / 1 pont

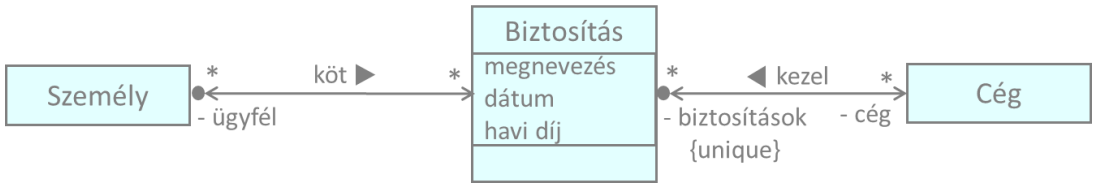
A személyek biztosításokkal foglalkozó cégekkel kötnek biztosításokat.

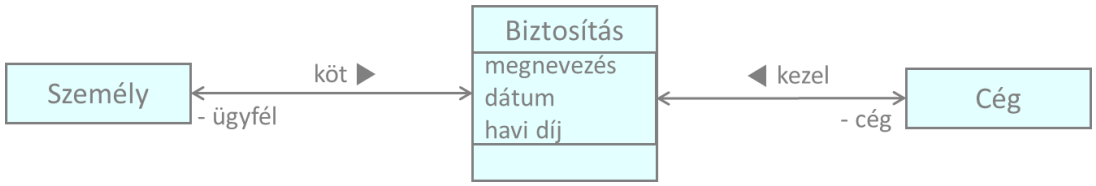


Melyik alábbi osztály diagram felel meg leginkább a fenti modellnek?





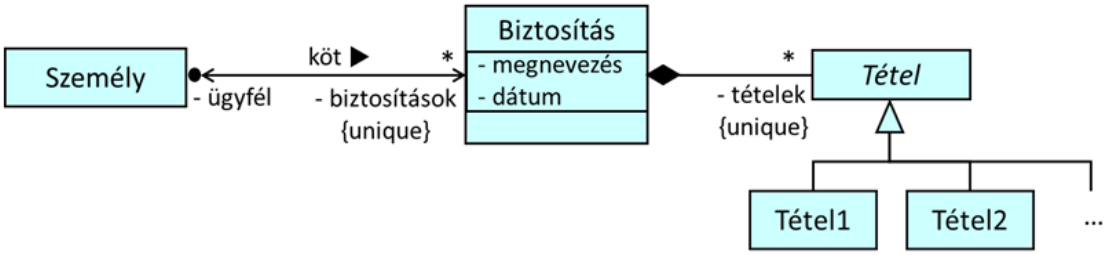




4. kérdés

1 / 1 pont

Egy biztosítási szerződés keretében több különböző kártétel szerepelhet, amelyeket a biztosítás megkötésekor kell megadni.



Milyen metódus(ok) szükséges(ek) egy új biztosítás megkötéséhez? Az alábbi megoldások közül melyek helyesek?

```
Személy osztály Kötés() metódusa:
Kötés(n:string, d:Dátum, lista:string*)
{
    new Biztosítás(n, d, this, lista)
}

Biztosítás osztály konstruktora:
Biztosítás(n:string, d:Dátum, ü:Személy, lista:string*)
{
   megnevezés := n; dátum := d; ügyfél := ü
   foreach(t in lista) loop
      switch ( t )
         case ”tétel1”:  tétel := new Tétel1()
         case ”tétel2”:  tétel := new Tétel2()
          …
      endswitch
      tételek.Add(tétel)
   endloop
}
```

```
Biztosítás osztály konstruktora:
Biztosítás(n:string, d:Dátum, ü:Személy, lista:string*)
{
   megnevezés := n; dátum := d; ügyfél := ü
   foreach(t in lista) loop
      switch ( t )
         case ”tétel1”:  tétel := new Tétel1()
         case ”tétel2”:  tétel := new Tétel2()
          …
      endswitch
      tételek.Add(tétel)
   endloop
}
```

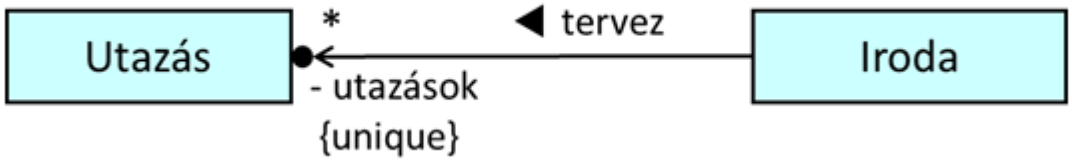
```
Biztosítás osztály osztályszintű Kötés() metódusa:
Kötés(n:string, d:Dátum, ü:Személy, lista:string*)
{
   b = new Biztosítás()
   b.megnevezés := n; dátum := d; ügyfél := ü
   foreach(t in lista) loop
      switch ( t )
         case ”tétel1”:  tétel := new Tétel1()
         case ”tétel2”:  tétel := new Tétel2()
          …
      endswitch
      b.tételek.Add(tétel)
   endloop
}
```

```
Személy osztály Kötés() metódusa:
Kötés(n:string, d:Dátum, lista:string*)
{
   b = new Biztosítás()
   b.megnevezés := n; dátum := d; ügyfél := ü
   foreach(t in lista) loop
      switch ( t )
         case ”tétel1”:  tétel := new Tétel1()
         case ”tétel2”:  tétel := new Tétel2()
          …
      endswitch
      b.tételek.Add(tétel)
   endloop
}
```

5. kérdés

1 / 1 pont

Egy utazási iroda csoportos körutazásokat hirdet. Milyen metódussal hozzunk létre új utazást?



Az *Iroda* osztályának egy metódusával, amelyik az utazások.Add(new Utazás(…)) utasítást is tartalmazza.

Az *Utazás* osztály konstruktorával, amelyik az utazások.Add(new Utazás(…)) utasítást is tartalmazza.

Az *Iroda* osztály konstruktorával, amelyik az utazások.Add(new Utazás(…)) utasítást is tartalmazza.

Az *Utazás* osztályának egy metódusával, amelyik az utazások.Add(new Utazás(…)) utasítást is tartalmazza.

6. kérdés

1 / 1 pont

Egy utazási iroda csoportos körutazásokat hirdet, ahol egy körutazásra megadott létszámkorlátig jelentkezhetnek az utasok. Mi kerüljön a Jelentkezik() metódus törzsébe?


```
if (|utas|=max) or (p in utas) then error endif; utas.Add(p)
```

```
if p in utas then error endif; utas.Add(p)
```

```
if |utas|=max then error endif ; utas.Add(p)
```

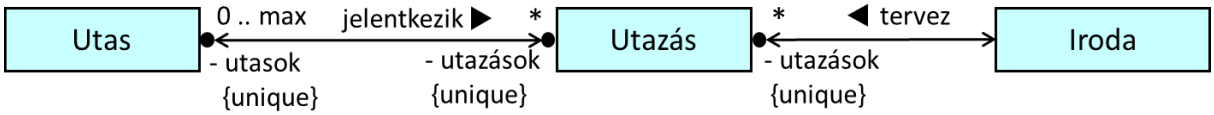
```
utas.Add(p)
```

7. kérdés

1 / 1 pont

Egy utazási iroda csoportos körutazásokat hirdet, ahol egy körutazásra megadott létszámkorlátig jelentkezhetnek az utasok. Hogyan válaszoljuk meg azt a kérdést,

hogy hány olyan utazás van, ahol van még szabad hely?



Ehhez az *Iroda* osztály alábbi törzsű publikus metódusára van szükség:

return $\sum_{ut \in utazasok} 1|ut.utasok| < max$

Ehhez az *Utazás* osztály alábbi törzsű publikus metódusára van szükség:

return $\sum_{ut \in utazasok} 1|ut.utasok| < max$

Ehhez a *Utas* osztály alábbi törzsű publikus metódusára van szükség:

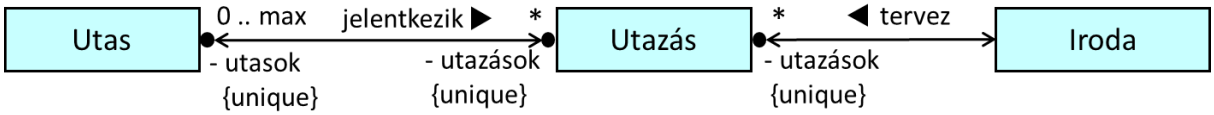
return $\sum_{ut \in utazasok} 1|ut.utasok| < max$

Ezt a kérdést nem lehet megválaszolni a modell alapján.

8. kérdés

1 / 1 pont

Egy utazási iroda csoportos körutazásokat hirdet, ahol egy körutazásra megadott létszámkorlátig jelentkezhetnek az utasok.



Hogyan válaszoljuk meg azt a kérdést, hogy melyik utazásra jelentkezett a legtöbb utas?

Ehhez az *Iroda* osztály alábbi törzsű publikus metódusára van szükség:

$$\textbf{if } |\text{utazasok}| = 0 \textbf{ then error end}$$
$$\textbf{return } \mathbf{MAX}_{\text{ut} \in \text{utazasok}} \text{ut.Hany}()$$

és az Utazás osztály Hány() metódusának törzse: **return** |utasok|

Ehhez az *Iroda* osztály alábbi törzsű publikus metódusára van szükség:

$$\textbf{if } |utazasok| = 0 \textbf{ then error end}$$
$$\textbf{return } \mathbf{MAX}_{\text{ut} \in \text{utazasok}} |\text{ut.utasok}|$$

Ehhez az *Iroda* osztály alábbi törzsű publikus metódusára van szükség:

$$\textbf{return } \mathbf{MAX}_{\text{ut} \in \text{utazasok}} \text{ut.Hany}()$$

és az Utazás osztály Hány() metódusának törzse: **return** |utasok|

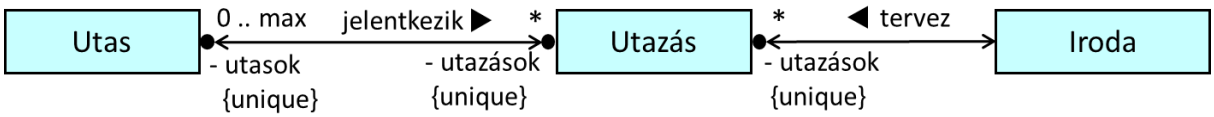
Ehhez az *Iroda* osztály alábbi törzsű publikus metódusára van szükség:

$$\textbf{return } \mathbf{MAX}_{\text{ut} \in \text{utazasok}} |\text{ut.utasok}|$$

9. kérdés

1 / 1 pont

Egy utazási iroda csoportos körutazásokat hirdet, ahol egy körutazásra megadott létszámkorlátig jelentkezhetnek az utasok.



Hogyan válaszoljuk meg azt a kérdést, hogy igaz-e, hogy minden utazásra jelentkezett már olyan utas, aki több utazásra is regisztrált?

Ehhez az *Iroda* osztály alábbi törzsű publikus metódusára van szükség:

$\mathbf{return}\ \forall \mathbf{SEARCH}_{\mathrm{ut}\ \mathbf{in}\ \mathrm{utazasok}}\ \mathrm{ut.Van}()$

ahol az *Utazás* osztály Van() publikus metódusa: $\mathbf{return}\ \sum_{e\ \mathbf{in}\ utasok} 1 \qquad e.UtakDb() > 1$

és az *Utas* osztály UtakDb() publikus metódusa: **return** |utazások|

Ehhez az *Iroda* osztály alábbi törzsű publikus metódusára van szükség:

$\mathbf{return}\ \forall \mathbf{SEARCH}_{\mathrm{ut}\ \mathbf{in}\ \mathrm{utazasok}}\ (\mathbf{SEARCH}_{\mathrm{e}\ \mathbf{in}\ \mathrm{ut.utasok}}\ e.UtakDb() > 1)$

ahol az *Utas* osztály UtakDb() publikus metódusa: **return** |utazások|

Ehhez az *Iroda* osztály alábbi törzsű publikus metódusára van szükség:

$\mathbf{return}\ \forall \mathbf{SEARCH}_{\mathrm{ut}\ \mathbf{in}\ \mathrm{utazasok}}\ ut.Van()$

ahol az *Utazás* osztály Van() publikus metódusa: $\mathbf{return}\ \sum_{e\ \mathbf{in}\ utasok} 1 \qquad |e.utazasok| > 1$

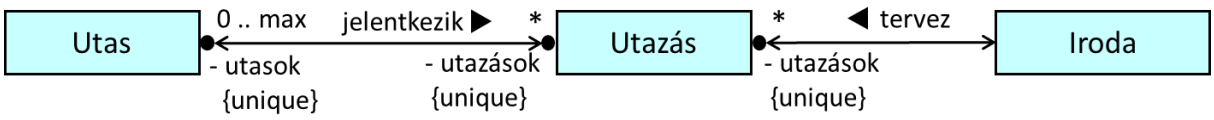
Ehhez az *Iroda* osztály alábbi törzsű publikus metódusára van szükség:

$\mathbf{return}\ \forall \mathbf{SEARCH}_{\mathrm{ut}\ \mathbf{in}\ \mathrm{utazasok}}\ (\mathbf{SEARCH}_{\mathrm{e}\ \mathbf{in}\ \mathrm{ut.utasok}}\ |\mathrm{e.utazasok}| > 1)$

10. kérdés

1 / 1 pont

Egy utazási iroda csoportos körutazásokat hirdet, ahol egy körutazásra megadott létszámkorlátig jelentkezhetnek az utasok. Keressük meg azt az utast, aki a legtöbb utazásra jelentkezett?



```
max := 0
foreach út in utazások loop
    foreach u in út.Utasok() loop
        if |e.Utazások()|>max then max, maxutas := |u.Utazások()|, u endif
    endloop
endloop
if max=0 then error endif
return maxutas

ahol az Utazás osztály Utasok() publikus metódusa egy utazás utasainak sorozatát,
az Utas osztály Utazások() publikus metódusa egy utas utazásainak sorozatát adja vissza.
```

```
max := 0
foreach út in utazások loop
   db, utas := MAX_{u in ut.Utasok()} u.Utazasok()
   if db>max then max, maxutas := db, utas endif
endloop
if max=0 then error endif
return maxutas

ahol az Utazás osztály Utasok() publikus metódusa egy utazás utasainak sorozatát,
az Utas osztály Utazások() publikus metódusa egy utas utazásainak sorozatát adja vissza.
```

```
max := 0
foreach út in utazások loop
   db, utas := MAX_{u in ut.utasok} u.Utazasok()
   if db>max then max, maxutas := db, utas endif
endloop
if max=0 then error endif
return maxutas

ahol az Utas osztály Utazások() publikus metódusa egy utas utazásainak sorozatát adja vissza.
```

```
max := 0
foreach út in utazások loop
   db, utas := MAX_{u in ut.utasok} |u.utazasok|
   if db>max then max, maxutas := db, utas endif
endloop
if max=0 then error endif
return maxutas
```