Az első fájlos program

Tartalom

Több lépésben oldunk meg néhány, rokon feladatot, hogy könnyen legyen felfedezhető minden algoritmikus és Pascal nyelvi elem tudnivalói. Az első lépés még csak „bemelegítésül” szolgál: abban csak a probléma algoritmikus vetületével foglalkozunk. Csak a második lépésben jutunk el a fájlokkal kapcsolatos gondolatokig.

1. Első lépés

1.1. A feladat

A klaviatúráról beolvassuk az alábbi adatokat:

- év1 hó1 nap1 (köztük egy-egy szóköz)
  Ez a felhasználó születési idejének a dátuma.
- év2 hó2 nap2 (köztük egy-egy szóköz)
  Ez a felhasználó munkába lépésének a dátuma. Világos, hogy elvárható:
  év1.hó1.nap1<év2.hó2.nap2.
- nem ('N' vagy 'F')
  A felhasználó neme: nő⇒'N', férfi⇒'F'.

Adjuk meg, hogy hány nap múlva megy legkorábban nyugdíjba! Tudjuk, hogy a hatályos törvények szerint[1]:

- ha valaki nő, akkor 40 éves munkaviszony után is nyugdíjba mehet, vagy (akár nő, akár férfi) a 62. évét betöltve.

Programozási „újdonság” (az ElsoProgramB-hez és a TombosProgramB-hez): „új” alprogramokat célszerű definiálni. Nevezetesen:

- szökő-évet eldöntő logikai függvény
- dátum évbeli nap-sorszáma függvény
- dátum 1900-hoz képest napok száma függvény
- beolvasáshoz/feldolgozáshoz/kiíráshoz eljárás

1.2. Specifikáció

Be: É,H,N, É1,H1,N1, É2,H2,N2∈ℕ *[az aktuális dátum (op.rendszertől), a születés és a munkába lépés dátuma]*
NEM∈𝕂 *[nő/férfi: egyetlen karakter]*
Konstans hóHossz∈ℕ[12]=(31,28,31,30,31,30,31,31,30,31,30,31)

Ki: NSZ∈ℕ *[ennyi nap múlva mehet nyugdíjba]*

Ef: E1≥1900 ∧ E1<E ∧ H1∈[1..12] ∧ N1∈[1..31] ♣ ∧
E2>E1 ∧ E2<E ∧ H2∈[1..12] ∧ N2∈[1..31] ♣ ∧
NEM∈{'N','F'}

Uf: NEM='F' ∧ (É,H,N)–(É1,H1,N1)<62 →
NSZ=NapSorszám1900tól(É1+62,H1,N1)–NapSorszám1900tól(É,H,N) ∧
NEM='N' ∧ ((É,H,N)–(É1,H1,N1)<62) ∧ ((É,H,N)–(É2,H2,N2)<40) →
NSZ=Min(NapSorszám1900tól(É1+62,H1,N1), NapSorszám1900tól(É2+40,H1,N1))–
NapSorszám1900tól(É,H,N)
NEM='F' ∧ (É,H,N)–(É1,H1,N1)≥62 ∨ … → NSZ=0

---

[1] 2009 nyarán. ;)

♣ Hogy lehetne még precízebbé tenni a dátummal szembeni elvárásokat?

Def: $- : \mathbb{N}^3 \times \mathbb{N}^3 \to \mathbb{N}$ *[dátum×dátum|→betöltött év]*,

$$(é,h,n)-(é1,h1,n1) := \begin{cases} é - é1 - 1 & \text{, ha NapSorszámÉvben}(é,h,n) < \text{NapSorszámÉvben}(é,h1,n1) \\ é - é1 & \text{, különben} \end{cases}$$

NapSorszámÉvben: $\mathbb{N}^3 \to \mathbb{N}$

$$\text{NapSorszámÉvben}(é,h,n) := \begin{cases} \sum_{i=1}^{h-1} \text{hóHossz}_i + 1 + n & \text{, ha } h > 2 \text{ és Szökő?}(é) \\ \sum_{i=1}^{h-1} \text{hóHossz}_i + n & \text{, különben} \end{cases}$$

Szökő?: $\mathbb{N} \to \mathbb{L}$

Szökő?(é):= (é Mod 400)=0 ∨ (é Mod 4)=0 ∧ (é Mod 100≠0)

NapSorszám1900tól: $\mathbb{N}^3 \to \mathbb{N}$

NapSorszám1900tól(é,h,n):= $(é-1)\cdot 365 + (é-1)\,\text{Div}\,4 + \text{NapSorszámÉvben}(é,h,n)$, ha é∈[1900..2099]

1.3. Algoritmus

A specifikációban szereplő függvényeket az algoritmusban is függvényként fogalmazzuk újra. E mellett célszerű a **felülről-lefelé tervezés elv**ét a feladat megoldására is alkalmazni, ami azt jelenti, hogy a legfelsőbb szintű megoldást eleve több részfeladatot megoldó alprogramra bontjuk. Nevezetesen: a beolvasásra, a lényegi számítást elvégzőre és az eredményt megjelenítőre. Mivel ezek a (specifikációból közvetlenül származtatott) globális adatokkal operálnak, és a végrehajtás során ezeket csak egyszer használjuk, paraméter nélküli eljárásokként definiáljuk.

```
Program FájlosProgram A:
  Változó
    é,h,n,hn,       [a mai nap]
    é1,h1,n1,       [születés]
    é2,h2,n2:Egész  [munkába állás]
    nem:Karakter    [nő/férfi]
    nsz:Egész [napok száma a nyugdíjig]    ← Globális adatok.
  Konstans
    hóHossz:Tömb(1..12:Egész)=(31,28,31,
             30,31,30,31,31,30,31,30,31)
    nőiMunkaviszony:Egész=40
    nyugdíjKorhatár:Egész=62
  Dátum(é,h,n,hn)
  Beolvas                                  ← A fő program, amelyben a Dátum függvényt (aktuális)
  Feldolgoz                                  paraméterekkel használjuk, míg a finomítás során definiált
  Kiir                                       eljárásokat paraméter nélkül. Ők közvetlenül manipulálják
                                             a globális adatokat.
Program vége.
```

A legfelsőbb szintű rutinok kifejtése:

```
Eljárás Beolvas:
  Be:é1,h1,n1  [é1≥1900 És é1<é ÉS
                h1>0 És h1≤12 És
                n1>0 És n1≤31]
  Be:é2,h2,n2  [é2≥é1 És é2<é ÉS
                h2>0 És h2≤12 És          ← A beolvasásnál csak a lényegre koncentrálunk.
                n2>0 És n2≤31]              Hogy lehet tovább precizírozni a feltételeket?
  Be:nem  [NagyBetű(nem)='N' Vagy
           NagyBetű(nem)='F']
Eljárás vége.                              ← Az eljárást lezáró utasítás.
```

```
Eljárás Feldolgoz:
  [lokális adatok:]
  Változó
    kor,
    éNy1,éNy2:Egész; [nyugdíjba vonulás éve kor/munkaviszony miatt]
```

```
  [betöltött évek száma (kora):]
  kor:=é-é1
  [ha még nem töltötte be => korrekció:]
  Ha NapSorszámÉvben(é,h,n)<NapSorszámÉvben(é,h1,n1) akkor kor:=kor-1
  Ha nem='N' akkor [nő]
    Ha ((é-é2<nőiMunkaviszony) Vagy
        (é-é2=nőiMunkaviszony) És (NapSorszámÉvben(é,h,n)<NapSorszámÉvben(é,h2,n2)))
       És kor<nyugdíjKorhatár
    akkor [még nem ment nyugdíjba]
      éNy1:=é1+nyugdíjKorhatár [kora miatti nyugdíj korhatár]
      éNy2:=é2+nőiMunkaviszony [munkaviszony miatti nyugdíj korhatár]
      Ha éNy1<éNy2 akkor [kora miatt nyugdíjas]
        nsz:=NapSorszám1900tól(éNy1,h1,n1)-NapSorszám1900tól(é,h,n)
      különben [munkaviszonya miatt nyugdíjas]
        nsz:=NapSorszam1900tól(éNy2,h2,n2)-NapSorszam1900tól(é,h,n)
      Elágazás vége
    különben [már nyugdíjban lehet]
      nsz:=0
    Elágazás vége
  különben [férfi]
    Ha kor<nyugdíjKorhatár akkor [még nem ment nyugdíjba]
      éNy1:=é1+nyugdíjKorhatár
      nsz:=NapSorszám1900tól(éNy1,h1,n1)-NapSorszám1900tól(é,h,n)
    Elágazás vége
  Elágazás vége
Eljárás vége.

Eljárás Kiír:
  Ki:nsz
Eljárás vége.
```

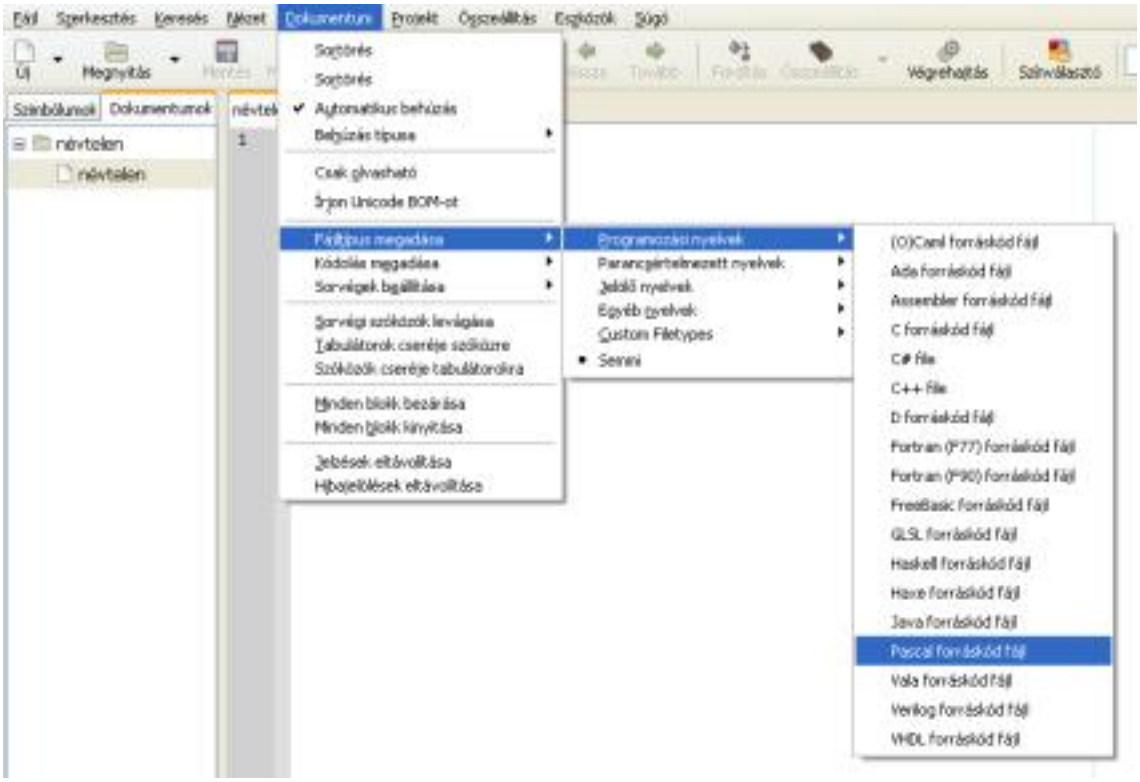
A specifikációból származó, illetve az előző finomítási szinten (a felhasználás által) definiált alprogramok:

```
Függvény Szökő?(Konstans e:Egész):Logikai   ← A formális paraméter Konstans, mert a függvényben nem fog megváltozni.
  Szökő?:=(e Mod 400)=0 Vagy                ← A függvény értékét a függvény nevével megegyező lokális
          (e Mod 4)=0 És (e Mod 100)≠0        változónak adott érték jelenti.
Függvény vége.                              ← A függvényt lezáró utasítás.

Függvény NapSorszámÉvben(                   A paraméterek csak „befelé” viszik az információt, tehát
              Konstans e,h,n:Egész):Egész     Konstansok.
  Változó                                   ← Lokális adatok.
    nDb,i:Egész
  nDb:=0
  Ciklus i=1-től h-1-ig                     ← Az adott hónap első napja előtt levő napok össz számát meg-
    nDb:=nDb+hóHossz(i)                       határozzuk.
  Ciklus vége
  nDb:=nDb+n
  Ha h>2 akkor
    Ha Szökő?(e) akkor nDb:=nDb+1           ← Ha az esetleges szökő-nap után vagyunk, korrigálunk.
  Elágazás vége
  NapSorszámÉvben:=nDb                      ← A függvény értékének megadása.
Függvény vége.

Függvény NapSorszám1900tól(                 A paraméterek csak „befelé” viszik az információt, tehát
             Konstans e,h,n:Egész):Egész    Konstansok.
  NapSorszám1900tól:=(e-1)*365+
                     ((e-1) Div 4)+
                     NapSorszámÉvben(e,h,n)
Függvény vége.
```

1.4. Kód

Az alábbi kód magát kommentálja. A Geanyben viszont most „0-ról kell indulni”. Egyetlen probléma, hogy az új dokumentum megnyitása után, hogyan lehet a Geany számára közölni, hogy ez egy Pascal nyelvű program lesz. Ezt nyilván amiatt kell tudnia, mert a nyelvhez

tartozó fordítóprogramot kell majd elindítania. Kevés próbálkozás után megtalálható a mód: „Dokumentum” menücsoport (`[Alt+D]`), „Fájltípus megadása” alcsoport (`[T]`), Programozási nyelvek” al-alcsoport s végül „Pascal”.



1. ábra: A programozás nyelv beállítása egy új program beírásakor, Geanyben.

```
Program FajlosProgram_A;

  Uses
    Dos,Crt;

  Var
    e,h,n,hn,       {a mai dátum és hét-napja}
    e1,h1,n1,       {születési dátum}
    e2,h2,n2:Word;  {munkába állási dátum}
    nem:Char;       {nő/férfi}
    nsz:Word;       {a nyugdíjig hátra lévő napok száma}
  Const
    cim:String='Nyugdíjszámítás';
    hoHossz:Array [1..12] of Word=(31,28,31,30,31,30,31,31,30,31,30,31);
    noiMunkaviszony:Integer=40;
    nyugdijKorhatar:Integer=62;

{$i AltRutinok.inc}

  Function SzokoE(Const e:Word):Boolean;
  Begin
    SzokoE:=((e Mod 400)=0) {400-zal osztható évszázad} or
            ((e Mod 4)=0) and ((e Mod 100)<>0) {4-gyel osztható nem évszázad}
  End;

  Function NapSorszamEvben(Const e,h,n:Word):Word;
    Var
      nDb,i:Word;
  Begin
    nDb:=0;
    For i:=1 to h-1 do nDb:=nDb+hoHossz[i];
    nDb:=nDb+h;
```

```
  {Szökő-év figyelembe vétele:}
  If h>2 then {ha az esetleges szökő-nap után vagyunk}
  Begin
    If SzokoE(e) then nDb:=nDb+1;
  End;
  NapSorszamEvben:=nDb;
End;

Function NapSorszam1900tol(Const e,h,n:Word):Word;
{Figyelembe véve: e ELEME 1900..2099}
Begin
  NapSorszam1900tol:=(e-1)*365+((e-1) Div 4)+NapSorszamEvben(e,h,n);
End;

Procedure Beolvas;
Begin
  Repeat
    Write('Születési dátum (''év hó nap'' alakban):');
    Readln(e1,h1,n1);
  Until (e1>=1900) and (e1<e) and (h1>0) and (h1<=12) and
        (n1>0) and (n1<=31);
  Repeat
    Write('Munkába lépés dátuma (''év hó nap'' alakban):');
    Readln(e2,h2,n2);
  Until (e2>E1) and (e2<e) and (h2>0) and (h2<=12) and
        (n2>0) and (n2<=31);
  Repeat
    Write('Neme (N/F):');
    Readln(nem); nem:=UpCase(nem); {nagybetűssé alakítása}
  Until (nem='N') or (nem='F');
End;

Procedure Feldolgoz;
{az eljárás lokális adatai:}
  Var
    kor,
    eNy1,eNy2:Word; {nyugdíjba vonulás éve kor/munkaviszony miatt}
Begin
  kor:=e-e1;
  {ha még nem töltötte be => korrekció:}
  If NapSorszamEvben(e,h,n)<NapSorszamEvben(e,h1,n1) then Dec(kor);
  If nem='N' then {nő}
  Begin
    If ((e-e2<noiMunkaviszony) or
        (e-e2=noiMunkaviszony) and
        (NapSorszamEvben(e,h,n)<NapSorszamEvben(e,h2,n2)) and
       (kor<nyugdijKorhatar)) then {még nem ment nyugdíjba}
    Begin
      eNy1:=e+nyugdijKorhatar-kor;
      eNy2:=e2+noiMunkaviszony;
      If eNy1<eNy2 then {kora miatt megy előbb nyugdíjba}
      Begin
        nsz:=NapSorszam1900tol(eNy1,h1,n1)-NapSorszam1900tol(e,h,n)
      End
        else {munkaviszonya miatt megy előbb nyugdíjba}
      Begin
        nsz:=NapSorszam1900tol(eNy2,h2,n2)-NapSorszam1900tol(e,h,n);
      End;
    End
      else {már nyugdíjba ment}
    Begin
      nsz:=0;
    End;
  End
```

```
      else {férfi}
    Begin
      If kor<nyugdijKorhatar then {még nem ment nyugdíjba}
      Begin {férfi => kora miatt megy nyugdíjba}
        eNy1:=e1+nyugdijKorhatar;
        nsz:=NapSorszam1900tol(eNy1,h1,n1)-NapSorszam1900tol(e,h,n)
      End;
    End;
  End;

  Procedure Kiir;
  Begin
    Writeln('A nyugdíjig még szükséges napok száma:',nsz);
  End;

Begin
  UjLap(cim);
  GetDate(e,h,n,hn);
  Beolvas;
  Feldolgoz;
  Kiir;
  BillreVar;
End.
```

2. Második lépés

2.1. A feladat

Szöveg fájlból olvassuk be az alábbi adatokat (soronként):

- db

  A személyek száma.

  Minden egyes személyhez az alábbi adatok tartoznak, azaz 3*db sorban:

- év1 hó1 nap1 (köztük egy-egy szóköz)

  Ez a felhasználó születési idejének a dátuma.

- év2 hó2 nap2 (köztük egy-egy szóköz)

  Ez a felhasználó munkába lépésének a dátuma.

  Világos, hogy elvárható: év1.hó1.nap1<év2.hó2.nap2.

- nem ('N' vagy 'F')

  A felhasználó neme: nő⇒'N', férfi⇒'F'.

A fájlbeli adatok helyesek.

Adjuk meg személyenként, hogy hány nap múlva megy legkorábban nyugdíjba! Minden személyhez 1-1 sor tartozik a kimeneti fájlban. Kezdjük a kiírást a személyek számával!

Tudjuk, hogy a hatályos törvények szerint:

- ha valaki nő, akkor 40 éves munkaviszony után is nyugdíjba mehet, vagy (akár nő, akár férfi) a 62. évét betöltve.

Programozási „újdonságok”:

- „új” alprogramok:
  - szöveg fájl megnyitásához
  - szöveg fájl lezárásához
- szöveg fájlkezelő műveletek

Gyakorlásképpen az adatokat tömbbe olvassuk, és onnan dolgozzuk föl. A dátumkezelő saját függvényeket elkülönítjük a `DatumRutinok.inc` fájlba.

2.2. Specifikáció

A specifikációval most nem törődünk.

2.3. Algoritmus

Az algoritmus nyelvi újdonsága a **fájlkezelés**hez kapcsolódik. A fájlok azonosításához a programban két adatra van szükség: 1) az ún. **fájlváltozó**, ami magát azt az adatsorozatot jelképezi, amely a háttértáron van/lesz tárolva; 2) a fájl operációs rendszer számára ismert neve, azaz, a **fájlnév** (egy szöveges típusú adat). Mi csak az ún. **szövegfájl**okkal foglalkozunk. Ezek karakterekből állnak.

A szövegfájl deklarációja:

```
Változó f:SzövegFájl
```

A következő műveletekkel lehet kezelni:

- megnyitni olvasásra vagy írásra – ez a hozzáférést lehetővé tevő művelet, csak ezt követőleg lehet az alábbi műveleteket végrehajtani:

```
Függvény OlvasásraNyit(Változó f:SzövegFájl, Konstans fn:Szöveg):Logikai
[Igaz, ha a megnyitás sikerült; hamis, egyébként]
Függvény ÍrásraNyit(Változó f:SzövegFájl, Konstans fn:Szöveg):Logikai
[Igaz, ha a megnyitás sikerült; hamis, egyébként]
```

- olvasni belőle – feltéve, hogy olvasásra nyitottuk meg

```
Eljárás Olvas(Változó f:SzövegFájl, változók)
[a fájl aktuális sorából feltölti a változókat]
Eljárás SorOlvas(Változó f:SzövegFájl, változók)
[a fájl aktuális sorából feltölti a változókat, a sor maradékát eldobja]
```

- írni hozzá – feltéve, hogy írásra nyitottuk meg

```
Eljárás Ír(Változó f:SzövegFájl, kifejezések)
[a fájl aktuális sorába írja a kifejezések értékeit]
Eljárás SorÍr(Változó f:SzövegFájl, kifejezések)
[a fájl aktuális sorába írja a kifejezések értékeit, és sorvégjelet ír]
```

- lezárni – eztán már nem lehet a fájllal (a programban semmit csinálni; persze az újra megnyitásig!)

```
Eljárás Lezár(Változó f:SzövegFájl)
```

Az algoritmustól most eltekintünk.

2.4. Kód

A szöveges fájl kezelésének Pascal minimuma:

A fájlváltozó **deklaráció**ja:

```
Var f:Text {másként: f:File of Char}
```

Megnyitás olvasásra:

```
Assign(f,fájlnév); Reset(f); {ha nem létezik a fájl, akkor megállás}
```

Megnyitás írásra:

```
Assign(f,fájlnév); Rewrite(f); {ha már létezik a fájl, akkor felülírás}
```

Olvasás pont úgy történik, mint a klaviatúráról:

```
Read(f,változók); {olvasás a fájl aktuális sorából a változókba}
Readln(f,változók); {olvasás a fájl aktuális sorá a változókba, a sor
                     maradéka elvész}
```

Írás pont úgy történik, mint a képernyőre:

```
Write(f,kifejezések); {a kifejezések értékének írása a fájl utolsó sorába}
```

```
Writeln(f,kifejezések); {a kifejezések értékének írása a fájl utolsó
                         sorába, majd sorvégjel írása}
```

Lezárás:

```
Close(f); {az írott fájl csak ekkor „véglegesítődik” a háttértáron}
```

A program kódja rejteget némi további újdonságokat. Lássuk:

```
Program FajlosProgram_B;
  Uses
    Dos,Crt;
  Const
    maxSzemely=10; {a próbához ennyi elég}
  Type                                          ← Tömb-típusok definiálása, maximális mérettel.
    WTomb=Array [1..maxSzemely] of Word;
    CTomb=Array [1..maxSzemely] of Char;
  Var
    szDb,
    e,h,n,hn:Word;
    e1,h1,n1,
    e2,h2,n2:WTomb;
    nem:CTomb;                                  ← A definiált típusok felhasználása a deklarációban.
    nsz:WTomb;
  Const
    cim:String='Nyugdíjszámítás';
    hoHossz:Array [1..12] of Word=(31,28,31,
                  30,31,30,31,31,30,31,30,31);
    noiMunkaviszony:Integer=40;                 ← Feladatot pillanatnyilag meghatározó konstans
    nyugdijKorhatar:Integer=62;                   paraméterek. (A könnyű általánosíthatóság kedvéért.)
{$i AltRutinok.inc}                             ← A beillesztendő fájlokra hivatkozás.
{$i DatumRutinok.inc}
  Function FajlNyitasOlvasasra(
         Var f:Text; Const fN:String):Boolean;
  Begin
    Assign(f,fN);
    {$i-}Reset(f);{$i+}                         ← Ellenőrzött fájlnyitás, olvasásra.
    FajlNyitasOlvasasra:=IOResult=0;            ← A nyitás sikerességének az ellenőrzése.
  End;
  Function FajlNyitasIrasra(
         Var f:Text; Const fN:String):Boolean;
  Begin
    Assign(f,fN);
    {$i-}Reset(f);{$i+}                         ← Olvasásra nyitás, ellenőrzéssel!
    If IOResult=0 then                          ← A nyitás sikerességének az ellenőrzése.
    Begin {ilyen fájl már van!}
      FajlNyitasIrasra:=False                   ← Sikerült: az baj!
    End
      else
    Begin {ilyen fájl még nincs}
      Rewrite(f);                               ← Ilyen fájl még nincs, megnyitható írásra.
      FajlNyitasIrasra:=IOResult=0
    End;
  End;
  Procedure Beolvas;
    Var
      fN:String; {fájlnév}
      nn:Char;
      i:Word;
      f:Text;     {bemeneti adatfájl}
  Begin
    Repeat
      Write('A bemeneti fájl neve:');           ← A bemeneti fájlnyitás szertartása: a fájlnév bekérése
      Readln(fN);                                 és megnyitása.
    Until FajlNyitasOlvasasra(f,fN);
    Readln(f, szDb); {személyek száma}          ← A fájl első sorának beolvasása szDb-be.
```

```
    For i:=1 to szDb do
    Begin
      Readln(f, e1[i],h1[i],n1[i]);
      Readln(f, e2[i],h2[i],n2[i]);                ← A fájl további sorainak beolvasása a megfelelő
      Readln(f, nn);                                 tömbökbe.
      nn:=UpCase(nn); {nagybetűssé alakítás}
      nem[i]:=nn;
    End;
    Close(f);                                      ← A fájl lezárása, az eljárásnak vége.
  End;
  Procedure Feldolgoz;
    Var
      i,kor,
      eNy1,eNy2:Word; {nyugdíj éve kor/
                       munkaviszony miatt}
  Begin
    For i:=1 to szDb do                            ← Ciklus-szervezés: szDb elemet kell feldolgozni.
    Begin
      kor:=e-e1[i];
      {ha még nem töltötte be => korrekció:}
      If NapSorszamEvben(e,h,n)<                   ← A kor kiszámítása.
         NapSorszamEvben(e,h1[i],n1[i])
      then Dec(kor);
      If nem[i]='N' then                           ← Ha nő az illető…
      Begin {nő}
        If ((e-e2[i]<noiMunkaviszony) or
            (e-e2[i]=noiMunkaviszony) and
            (NapSorszamEvben(e,h,n)<               ← Megvizsgáljuk, hogy nyugdíj előtt áll-e?
             NapSorszamEvben(e,h2[i],n2[i]))
           and (kor<nyugdijKorhatar)) then
        Begin {még nem ment nyugdíjba}
          eNy1:=e1[i]+nyugdijKorhatar;             ← Mikor mehetne nyugdíba a két szempont miatt?
          eNy2:=e2[i]+noiMunkaviszony;
          If eNy1<eNy2 then                        ← Melyik szempon szerint mehet előbb?
          Begin {kora miatt előbb}
            nsz[i]:=NapSorszam1900tol(
                          eNy1,h1[i],n1[i])-       ← Hány nap múlva mehet nyugdíba?
                    NapSorszam1900tol(e,h,n)
          End
            else
          Begin {munkaviszony miatt előbb}
            nsz[i]:=NapSorszam1900tol(
                          eNy2,h2[i],n2[i])-       ← Hány nap múlva mehet nyugdíba?
                    NapSorszam1900tol(e,h,n);
          End;
        End
          else
        Begin {már nyugdíjba ment}
          nsz[i]:=0;                               ← Ha nyugdíjas lehet, akkor 0…
        End;
      End
        else
      Begin {férfi}                                ← Ha férfi az illető…
        If kor<nyugdijKorhatar then
        Begin {még nem ment nyugdíjba}
          eNy1:=e1[i]+nyugdijKorhatar;
          nsz[i]:=NapSorszam1900tol(
                        eNy1,h1[i],n1[i])-
                  NapSorszam1900tol(e,h,n)         ← Hány nap múlva mehet nyugdíba?
        End;
      End;
    End;
  End;
```

```
  Procedure Kiir;
    Var
      fN:String; {fájlnév}
      i:Word;
      f:Text;    {kimeneti adatfájl}
  Begin
    Repeat
      Write('A kimeneti fájl neve:');
      Readln(fN);
    Until FajlNyitasIrasra(f,fN);
    Writeln(f, szDb);
    For i:=1 to szDb do
    Begin
      Writeln(f, nsz[i]);
    End;
    Close(f);
  End;
Begin
  UjLap(cim);
  GetDate(e,h,n,hn);
  Beolvas;
  Feldolgoz;
  Kiir;
  BillreVar;
End.
```

← *A kimeneti fájlnyitás szertartása: a fájlnév bekérése és megnyitása.*

← *A fájl első sorának kiírása* `szDb`*-ből.*

← *A fájl további sorainak kiírása az* `nsz` *tömbökből.*

← *A fájl lezárása, az eljárásnak vége.*

← *A fő program.*

A `DatumRutinok.inc` fájl:

```
(*
  Dátumkezelő rutinok:
  * SzokoE(Const e:Word):Boolean
  * NapSorszamEvben(Const e,h,n:Word):Word;
  * NapSorszam1900tol(Const e,h,n:Word):Word;
*)
Function SzokoE(Const e:Word):Boolean;
Begin
  SzokoE:=((e Mod 400)=0) or
          ((e Mod 4)=0) and ((e Mod 100)<>0)
End;
Function NapSorszamEvben(
                      Const e,h,n:Word):Word;
  Var
    nDb,i:Word;
Begin
  nDb:=0;
  For i:=1 to h-1 do nDb:=nDb+hoHossz[i];
  nDb:=nDb+h;
  {Szökő-év figyelembe vétele:}
  If h>2 then {az esetleges szökő-nap után}
  Begin
    If SzokoE(e) then nDb:=nDb+1;
  End;
  NapSorszamEvben:=nDb;
End;
Function NapSorszam1900tol(
                      Const e,h,n:Word):Word;
{Figyelembe véve: e ELEME 1900..2099}
Begin
  NapSorszam1900tol:=(e-1)*365+((e-1) Div 4)+
                     NapSorszamEvben(e,h,n);
End;
```

← *A* `SzokoE` *függvény.*

← *A* `NapSorszamEvben` *függvény.*

← *A* `NapSorszam1900tol` *függvény.*

Érdemes a megnyitási szertartásokat külön –mondjuk `FajlosRutinok.inc` nevű– fájlba elkülöníteni, majd próbaképpen e programban felhasználni.