Az „Összegfokozatos listázás” téma 2. gyakorlata

Feladat:

Az alábbi szerkezetű file beolvasása és kilistázása
1. lapozottan "józanésszel",
2. úgy, hogy a típusfinomítással komolyabban élünk
(azaz ne használjuk ki, hogy az ElemKi figyeli a kiírt sorok számát),
3. az egy tanszéken dolgozók kiférjenek egy lapra
(a tanszék nyitórekorddal együtt)
Ef: nincs tanszék, ahol többen dolgoznának, mint a képernyősorok száma-2 ("-2" = fejsor + sorszámot tartalmazó lábléc)
4. minden szervezeti egységet egy ún. összegző rekorddal kiegészítve, és fenntartva az együttmaradás feltételét.
Ef: olyan, hogy megvalósítható legyen...

File-szerkezet:

1. szint: EGYETEM
szintkód, név, rektori hivatal cím, tel.szám
karok

2. szint: KAR
szintkód, név, dékáni hivatal cím, tel.szám
tanszékek (tanszéki egységek)

3. szint: TANSZÉKEK
szintkód, név, tanszékcím, tel.szám
dolgozók

4. szint: DOLGOZÓ
szintkód, név, lakcím, tel.szám, fizetés, ...

Feltételezés:
A file strukturálisan helyes, nem üres: legalább 1 dolgozó van.

Megjegyzés:
A file csupa azonos típusú elemből áll. Ezért a struktúra nyitó elemek kiegészülnek -"tartalom nélküli"- mezőkkel, amelyeket esetleg föl lehet használni az összegfokozatok utólagos tárolására.

Reprezentáció:

Input –

```
Type  TSzint=(Dolgozo,Tanszek,Kar,Egyetem,Ures);
Const KSzint:Array [TSzint] of String=(' ','T','K','E','');{kiirandó}
      szintKodH=1; nevH=28; cimH=26; telH=9; fizuH=6;      {kiíráshossz}
Type  TElem=Record
              szintKod:TSzint;
              nev:String[nevH]; cim:String[cimH]; tel:String[telH];
              fiz:LongInt;
      End;
Type  TFile=File of TElem;
```

Output:

```
Const MaxLapHossz=100; {egy-egy lapnyi maximum ennyi sorból állhat}
Type  TKepernyo= képernyő (Pascal) | TMemo (Lazarus) | TTabSheet (Lazarus)
                | Text ♣
Var   lapSzam,sor:byte; {a megnyitott lapok,
                         az aktuális lapra kiírt sorok száma}
```

Implementáció:

osszfok.inc – "legalapabb alap" rutinjai

ebben találhatók a lényeg szemponjából kevésbé fontos általános konstansok, változók, eljárások, függvények. Például:

- ***Procedure*** *ElemKi kiír egy elemet a fejléchez igazodva (izlésesen), ügyel a lapra kiírt sorok számára*
  TKepernyo-től függően más és más megoldás
- ***Procedure*** *Lapoz, ami lapot töröl, és fejlécet ír ki, sor-t nulláz*
  TKepernyo-től függően más és más megoldás
- ***Procedure*** *BillreVar, ami billentyű-lenyomásra vár*
  TKepernyo-től függően más és más megoldás
  (pl. a lapot megfelelő számú üres sorral tölt föl)
- ***Procedure*** *Megnyit file-t, megfelelő módon*
- ***Procedure*** *General adatfile-t a fenti szerkezetben*
- ***Procedure*** *Lezar file-t*
- ***Procedure*** *Inic, ami inicalizálja a kiírást (a Lapoz-on túliak)*
  TKepernyo-től függően más és más megoldás

Listázó rutinok:

1. **simaLista**
   -- egyszerü, lapozott listázás (természetes) virtuális típus bevezetésével
2. **maskentLista**
   -- (általánosítható) virtuális típus bevezetésével (útban a tanszékenkénti listázás felé)
3. **tanszekenkentLista**
   -- listázás 'tanszék dologozói együtt maradásával' feltétellel
4. **osszfokozatosLista**
   -- az előbbi listázás megfejelése az "egységeket" "szintenként" lezáró ún. összfokozati elemekkel (tanszék esetén a lezáró elem az egység együttmaradó része)

```
Procedure simaLista;
(*
  Virtuális típusok:
  Input:  TBeLap=TFile
          BeNyit,BeVége?,BeOlvas,BeZár
          (A reprezentáció egyszerüsége miatt:
          * BeNyit=Megnyit,
          * BeVége?=Eof,
          * BeOlvas=Read,
          * BeZár=Lezar)
  Output: TKiLap="Rekord"(sor:Egész♦,képernyö:TKépernyö)
          KiNyit,KiIr,KiZár
          (A reprezentáció egyszerüsége miatt:
          * KiNyit=inic + lapoz,
```

♣ Pirossal szedem, amik output-reprezentációfüggők.
♦ a képernyőn lévő sorok aktuális száma

```
            * KiIr=ElemKi (lapozásra ügyelve kiír),
            * KiZár=ha kell, lap alja)
  *)
  {
    Típus-reprezentációk:
  }
    Var  elem:TElem;
         f:TFile;
        {lapSzam,sor:Byte; <-- Globális}
  {
    Típus-implementációk:
  }
   Procedure KiNyit{output:TKiLap};
   Begin
      Inic; Lapoz;
   End;{KiNyit}
   Procedure KiZar{output:TKiLap};
   Begin
     If sor>0 then LapAlja;
   End;{KiZar}
  {
    A megoldás:
  }
  Begin{Listaz}
    Megnyit(f,Olvasas); Lapoz{output};
    While not Eof(f) do
    Begin
      Read(f,elem);
      ElemKi({output,}elem);
    End;
    Lezar(f); KiZar{output}
  End{Listaz};
(* ---------------------------------------------------------------------- *)
  Procedure maskentLista;
  (*
    Virtuális típusok:
    Input:  TBeLap=Rekord(db:Egész, sorok:Tömb(1..{Max}LapHossz:TElem))
            BeNyit,BeVége?,BeOlvas,BeZár
            * BeNyit=Megnyitas+belap-puffer inicializálás,
            * BeVége?=Eof,
            * BeOlvas=puffernyi olvasás,
            * BeZár=Lezar)
    Output: TKiLap="Rekord"(sor:Egész,képernyö:TKépernyö)
            KiNyit,KiIr(beLap-puffer!),KiZár
            * KiNyit=inic,
            * KiIr=lapoz
                  + beLap-puffer kiírása
                  + lap alja
            * KiZár=Lezár
  *)
  {
    Típus-reprezentációk:
  }
    Type TBeLap=Record
                  db:Byte; sorok:Array [1..{Max}LapHossz] of TElem;
                End;
    Var   egyLap:TBeLap;
          f:TFile;
```

```
          {lapSzam,sor:Byte; <-- Globális}
  {
    Típus-implementációk:
  }
    Procedure BeNyit(Var f:TFile);
    Begin
      Megnyit(f,Olvasas);
    End;{BeNyit}
    Procedure BeOlvas(Var f:TFile {;Var egyLap:TBeLap});
    Begin
      egyLap.db:=0;
      While Not Eof(f) and (egyLap.db<LapHossz-2)♥ do
      Begin
        Inc(egyLap.db);
        Read(f,egyLap.sorok[egyLap.db]);
      End;
    End;{BeOlvas}
    Procedure KiNyit{output:TKilap};
    Begin
      Inic;
    End;{KiNyit}
    Procedure KiIr{Var képenyő:TKépernyő, Var egyLap:TBeLap};
      Var i:Integer;
    Begin
      Lapoz;//a fejléc kiírása
      For i:=1 to egyLap.db do
      Begin
        ElemKi({képernyö,}egyLap.sorok[i]); {nem használom ki, hogy
                                             figyeli a sorok számát}
      End;
      LapAlja;//a lábléc kiírása
    End;{KiIr}
    Procedure KiZar{output:TKiLap};
    Begin
      //most semmi fontos
    End;{KiZar}
  {
    A megoldás:
  }
  Begin{maskentLista}
    Benyit(f); KiNyit{output};
    While not Eof(f) do
    Begin
      BeOlvas(f{,egyLap});
      KiIr{képernyö,egyLap};
    End;
    Lezar(f); KiZar{output}
  End{maskentLista};
(* ------------------------------------------------------------------ *)
  Procedure tanszekenkentLista;
  (*
    Virtuális típusok:
    Input:  bemeneti egység vagy egy tanszék összes dolgozója a nyitóval
                            vagy valami más elem
            a bementi eleme határát csak előreolvasási technikával
```

♥ TKépernyő-től függő, hiszen, ha a lapalja egy láblécet (is) jelent, akkor azt is bele kell számítani a laphosszba. A „füles lap” esetén ilyenre nincs szükség, tehát ott ez nem jelent többlet sort. 2, mert 1 fejléc + 1 lábléc

```
            tudjuk megtalálni
            TBeEgyseg=Rekord(db:Egész,
                             e:TElem [az elöreolvasott],
                             sorok:Tömb(1..MaxTszDolg+1:TElem))
            Elöfeltétel => MaxTszDolg+1=MaxLapHossz
            BeNyit,BeVége?,BeOlvas,BeZár
            * BeNyit=Megnyitas + elöreolvasás
            * BeVége?=Eof,
            * BeOlvas=egységolvasás elöreolvasással,
            * BeZár=Lezar)
  Output:   TKiLap="Rekord"(sor:Egész,képernyö:TKépernyö)
            KiNyit,KiIr(belap-puffer!),KiZár
            * KiNyit=Inic + lapoz,
            * KiIr=ha nem fér ki => lap alja
                                  + puffer inicializálása
                                  + lapoz
                  beLap-puffer kiírása+sor-állítás
            * KiZár=ha kell, lap alja
*)
{
  Típus-reprezentációk:
}
  Type TBeEgyseg=Record
                   db:Byte;
                   e:TElem;{az elöreolvasott}
                   sorok:Array [1..LapHossz] of TElem;
                 End;
  Var  egyTanszek:TBeEgyseg;
       f:TFile;
       {lapSzam,sor:Byte; <-- Globális}
{
  Típus-implementációk:
}
 Procedure BeNyit(Var f:TFile);
 {Ef: NOT Eof(f)}
 Begin
    Megnyit(f,fN,Olvasas);
    Read(f,egyTanszek.e);//előreolvasás
 End;
 Procedure BeOlvas(Var f:TFile {;Var egyTanszek:TBeLap});
 {Ef: NOT Eof(f) AND e=elöreolvasott elem}
 Begin
   egyTanszek.db:=1; egyTanszek.sorok[1]:=egyTanszek.e;
   Read(f,egyTanszek.e);
   While Not Eof(f) and (egyTanszek.e.szintKod=Dolgozo) do
   Begin
     Inc(egyTanszek.db); egyTanszek.sorok[egyTanszek.db]:=egyTanszek.e;
     Read(f,egyTanszek.e);
   End;
   {Uf: Eof(f)     => egyTanszek[1..egyTanszek.db]+e=egy egység AND
        NOT Eof(f) => e=következő dolgozó elem}
   If Eof(f) then
   Begin
     Inc(egyTanszek.db); egyTanszek.sorok[egyTanszek.db]:=egyTanszek.e;
     egyTanszek.e:=UresTElem;
   End;
   {Uf: egyTanszek[1..egyTanszek.db]=egy egység AND
        Eof(f)     => e=Üres AND
        NOT Eof(f) => e=következő dolgozó elem}
```

```
  End;
  Procedure KiNyit{output:TKilap};
  {Ef: a fájlnak legalább egy eleme van, így legalább egy lap kell}
  Begin
    Inic; Lapoz;
  End;//KiNyit
  Procedure KiIr{Var képernyö:TKépernyö, Var egyTanszek:TBeLap};
    Var i:Integer;
  Begin
    If sor+egyTanszek.db+1{lábléc}>=LapHossz then//nem fér rá az akt. lapra
    Begin
      LapAlja;//a lap "véglegesítése", lezárása
      Lapoz;  //új lap kezdése
    End;
    For i:=1 to egyTanszek.db do
    Begin
      ElemKi({képernyö,}egyTanszek.sorok[i]); {nem használom ki, hogy
                                               figyeli a sorok számát,
                                               csak, hogy kiírja és a
                                               sor-t növeli}
    End;
    egyTanszek.db:=0;
  End;
  Procedure KiZar{output:TKiLap};
  Begin
    If sor>0 then LapAlja;
  End;
 {
   A megoldás:
 }
 Begin{TanszekenkentLista}
   BeNyit(f); KiNyit{output};
   While not Eof(f) do
   Begin
     BeOlvas(f{,egyTanszek});
     KiIr{képernyö,egyTanszek};
   End;
   Lezar(f); KiZar{output};
 End{TanszekenkentLista};

(* ------------------------------------------------------------------ *)

 procedure osszfokozatosLista(const fN:string);
 (*
   Virtuális típusok:
   Input:  bemeneti egység vagy egy tanszék összes dolgozója a nyitóval
                          vagy valami más elem
           a bementi eleme határát csak előreolvasási technikával
           tudjuk megtalálni
           TBeEgyseg=Rekord(db:Egész,
                            elemek:Tömb(1..MaxTszDolg+1:TElem),
                            e:TElem [az elöreolvasott])
           Elöfeltétel => MaxTszDolg+1=LapHossz
           BeNyit,BeVége?,Beolvas,Bezár
           * BeNyit=Megnyitas+beLap-puffer inicializálás+elöreolvasás
           * BeVége?=Eof,
           * BeOlvas=egységolvasás elöreolvasással,
           * BeZár=Lezar
   Output: TKiLap="Rekord"(sor:Egész,képernyö:TKépernyö),
           KiNyit,KiIr(beLap-puffer!),KiZár
```

```
            * KiNyit=Inic + lapoz,
            * KiIr=ha nem fér ki => lap alja
                                   + lapoz
                          beLap-puffer kiírása + sor-állítás
            * KiZár=ha kell, lap alja
    Megjegyzés:
      Input, Output közt történik a beolvasott "egység" transzformációja,
      azaz a megfelelő számú lezáró, összegző rekord hozzátétele
  *)
  {
    Típus-reprezentációk:
  }
    Type  TBeEgyseg=Record
                      db:Byte;
                      elemek:Array [1..MaxLapHossz] of TElem;
                      e:TElem;{az elöreolvasott}
                    End;
    Var   egyTanszek:TBeEgyseg;
          osszFok:Array [TSzint] of TElem;
          f:TFile;
          {lapSzam,sor:Byte; <-- Globális}
          i,ig:TSzint;
  {
    Típus-implementációk:
  }
    Procedure BeNyit(Var f:TFile);
    Begin
      Megnyit(f,fN,Olvasas);
      Read(f,egyTanszek.e); //Elöreolvasás:egyetem
    End;{BeNyit}
    Procedure BeOlvas(Var f:TFile {;Var egyTanszek:TBeLap});
    {Ef: a fájl szintaktikailag helyes AND
         NOT Eof(f) AND e=elöreolvasott vmilyen szintü nyitó elem}
      Var i:TSzint;
    Begin
      egyTanszek.db:=0;
      For i:=egyTanszek.e.szintKod downto Tanszek do
      Begin
        Inc(egyTanszek.db);
        egyTanszek.e.fiz:=0;//nyitó elem fizu-ja 0-zandó
        egyTanszek.elemek[egyTanszek.db]:=egyTanszek.e;
        osszFok[i]:=egyTanszek.e;
        Read(f,egyTanszek.e);//elöreolvasás
      End;
      {Uf: egyTanszek.e=elöreolvasott dolgozó}
      {egy tanszék beolvasása, elkönyvelése:}
      While Not Eof(f) and (egyTanszek.e.szintKod=Dolgozo) do
      Begin
        Inc(egyTanszek.db); egyTanszek.elemek[egyTanszek.db]:=egyTanszek.e;
        Inc(osszFok[Tanszek].fiz,egyTanszek.e.fiz);
        Read(f,egyTanszek.e);
      End;
      {Uf: Eof(f)      => egyTanszek[1..egyTanszek.db]+e=egy egység AND
           NOT Eof(f) => e=következö nem dolgozó elem}
      If Eof(f) then
      Begin
        Inc(egyTanszek.db); egyTanszek.elemek[egyTanszek.db]:=egyTanszek.e;
        Inc(osszFok[Tanszek].fiz,egyTanszek.e.fiz);
        egyTanszek.e:=UresTElem;
```

```
      End;//If Eof
      {Uf: egyTanszek[1..egyTanszek.db]=egy egység AND
           Eof(f)     => e=üresTElem AND
           NOT Eof(f) => e=következö nem dolgozó elem}
    End;{BeOlvas}
    Procedure KiNyit{Var képernyö:TKépernyö};
    Begin
      Inic; Lapoz;
    End;{KiNyit}
    Procedure KiIr{Var képernyö:TKépernyö, Var egyTanszek:TBelap};
      Var i:Integer;
    Begin
      If sor+egyTanszek.db+1{lábléc}>LapHossz then
      Begin
        LapAlja; Lapoz;
      End;
      For i:=1 to egyTanszek.db do
      Begin
        ElemKi({képernyö,}egyTanszek.elemek[i]); {nem használom ki, hogy
                                                  figyeli a sorok számát,
                                                  csak, hogy kiírja és a
                                                  sor-t növeli}
      End;
    End;{KiIr}
    Procedure KiZar{output:TKiLap};
    Begin
      If sor>0 then LapAlja;
    End;
  {
    A megoldás:
  }
  begin{osszfokozatosLista}
    BeNyit(f); KiNyit{képernyö};
    {az osszFok-transzformáció előkészítése:}
    For i:=Dolgozo to Egyetem do osszFok[i]:=UresTElem;
    While not Eof(f) do
    Begin
      {Beolvasás:}
      BeOlvas(f{,egyTanszek});
      {"Feldolgozás" = zárórekord(ok) hozzávétele:}
      If Eof(f) then ig:=Egyetem else ig:=egyTanszek.e.szintKod;
      For i:=Tanszek to ig do
      Begin
        Inc(egyTanszek.db);
        egyTanszek.elemek[egyTanszek.db]:=osszFok[i];
        Inc(osszFok[Succ(i)].fiz,osszFok[i].fiz);//összfokozat göngyölítés
        osszFok[i]:=UresTElem;
      End;
      {Kiírás:}
      Kiir{képernyö,egyTanszek};
    End;
    Lezar(f); KiZar{képernyö};
  end;//osszfokozatosLista
```