Programozási tételek általánosítása 2.

Szlávi Péter
szlavip@elte.hu
2015

Tartalom

Sorozatok – A Tömb-típuskonstrukció

- A **tömb megvalósítása** az index- és az elemtípusokkal:
  Amikor erre gondolunk:

```
Típus
  TI = … indextípus …
  TE = … elemtípus …
  TI_Ek=Tömb(TI:TE)
```

akkor az előzőek után a tömböt így ábrázoljuk:

```
Típus
  TI = … indextípus …
  TE = … elemtípus …
  TI_Ek=Tömb(0..Számosság'TI-1:TE)
```

Tehetjük, mivel ∃`Sorszám` függvény, amely

`TI→0..Számosság'TI-1.`

Sorozatok – A Tömb-típuskonstrukció

- Példák:
  - 1..10 indexű 1..10 értékű tömb:

```
Típus
  TI_1_10_E_1_10k=Tömb(0..Számosság'TI_1_10:TE_1_10)
```

  - Konstans nevekkel indexelt ugyanazon neveket tartalmazó tömb:

```
Típus
  TI_Név_E_Névk=Tömb(0..Számosság'TI_Név:TE_Név)
```

- Feladat: Készítse el a **`Tömb(Típus TIndex:Típus TElem)`** tömb-típuskonstrukció megvalósítási modulját! Bővítse tömb-beolvasó és tömb-kiíró műveletekkel! (L. a Tömb exportmodul-ját.)

Sorozatok – A Tömb-típuskonstrukció

- Megoldás (a hiba kezeléssel csak elnagyoltan foglalkozunk):

```
Eljárás Be(Konstans kérdés:Szöveg, Változó t:Tömb)
  Változó
    i:TIndex
  Ki:kérdés
  Ciklus i=TIndex'Min-től TIndex'Max-ig
    Ciklus amíg Be(TIndex2Szöveg(i)+'-->',elemek(Sorszám(i)))
       [visszavezetés TElem-beli Be függvényre]
    Ciklus vége
  Ciklus vége
Eljárás vége.
Eljárás Ki(Konstans cím:Szöveg, t:Tömb)
  Változó
    i:TIndex
  Ki:kérdés
  Ciklus i=TIndex'Min-től TIndex'Max-ig
    Ki(TIndex2Szöveg(i)+'-->',elemek(Sorszám(i)),CrLf)
      [visszavezetés TElem-beli Ki eljárásra]
  Ciklus vége
Eljárás vége.
Modul vége.
```

Sorozatok – A Tömb-típuskonstrukció

Feladatok:

- Készítse el a tömb modul unitját (**OTomb_Sablon_Unit.pas**) építve a generic fogalomra! Definiálja a generic tömböt **OTomb<OIndex,OElem>** osztálysablon néven!

Letölt

- Írjon egy próbaprogramot az előbbi unit kipróbálására! Feladata lehetne ez is: „*a kurzus minden hallgatója nyilatkozik arról, hogy melyik társára adja voksát egy személyre vonatkozó kérdésben; igaz-e, hogy kölcsönösen egymásra szavaztak (X Y-ra és Y X-re)*”.
  A hallgatók tömbjének osztálya lehetne: **OI_Nev_E_Nevk**. (Indextípus: **OI_Nev**, elemtípus: **OE_Nev**.)

Letölt

- Készítsen egy egészeket tartalmazó **OE_Egész** modult és Pascalban írt megvalósító **OE_Egesz_Unit.pas** unitot, benne a **OE_Egesz** exportált típussal, elemtípus céljára!
- Az előbbi programot bővítse az alábbi feladattal: „*az egyes tagokra hány szavazat esett*”. Nyilván ehhez szükséges egy újabb tömb-típus (**OI_Nev_E_Egeszk**), azaz specializálni kell az **OTomb_Sablon_Unit.pas**-beli **OTomb<OIndex,OElem>** osztálysablont **OTomb<OI_Nev,OE_Egesz>** osztály létrehozásához.

Sorozatok – A tömb „indexmentesítése”

- Indexmentes változat – általánosított sorozatok:
  - Bemeneti sorozat típusa:

```
ExportModul TInputSorozat(Típus TE):
  Függvény
    Első(Változó s:TSorozat):TE
    Köv(Változó s:TSorozat):TE
    Vége?(Konstans s:TSorozat):Logikai
Modul vége.
```

  - Kimeneti sorozat típusa:

```
ExportModul TOutputSorozat(Típus TE):
  Eljárás
    Üres(Változó s:TSorozat)
    Végére(Változó s:TSorozat, Konstans e:TE)
Modul vége.
```

Ez egy későbbi témakörben szóba kerül. Most nem valósítjuk meg.

A Tulajdonság-függvények típusa – függvényparaméter

- Tulajdonság-függvények programozási tételekben
  - Feladat: soroljon föl néhány tételt, amelyben előfordul egy Tulajdonság-függvény!
- Függvény-típus fogalma:
  - Reprezentálás: adott szignatúrájú függvény memóriacíme
  - Implementálás: azonosság és értékadás művelet elegendő
  - Definiálás:

```
Típus FvTípusNév=Függvény(paraméterek):ÉrtékTípus
```

  - Deklarálás:

```
Változó fvNév:FvTípusNév
```

  - Felhasználási példa:

```
Típus T1VáltFv=Függvény(Konstans x:Egész):Egész
Függvény fv1(Konstans x:Egész):Egész
  …
Függvény vége.
Függvény fv2(Konstans x:Egész):Egész
  …
Függvény vége.
Változó f1,f2:T1VáltFv
…
f1:=fv1; f2:=fv2; y1:=f1(1)
Ha f1≠f2 akkor y2:=f2(1)
```

Tömbök és függvények tételekben

- Példák típusfogalmak felhasználására:
  - Adatleírás, általánosított tömbös változat:

```
Típus
  TI = … indextípus …
  TE = … elemtípus …
  TEk=Tömb(TI:TE)
  TKeresettI=Rekord(vanE:Logikai, melyik:TI) [index-kereséshez]
  TTulFv=Függvény(Konstans e:TE):Logikai
```

  - Keresés, általánosított tömbös változat (a „hagyományos” megoldással összevetve):

```
Függvény Keresés1(Konstans t:TEk, tFv:TTulFv):TKeresettI
  Változó i:TI
          k:TKeresettI
  i:=Min'TI
  Ciklus amíg i≤Max'TI és nem tFv(t(i))
    i:=Köv(i)
  Ciklus vége
  k.vanE:=i≤Max'TI [baj lesz!!!]
  Ha k.vanE akkor k.melyik:=i
  Keresés1:=k
Függvény vége.
```

```
Függvény Keresés1(Konstans t:TEk):TKeresettI
  Változó i:Egész
          k:TKeresettI
  i:=1
  Ciklus amíg i≤N és nem T(t(i))
    i:=i+1
  Ciklus vége
  k.vanE:=i≤N [miért nincs baj???]
  Ha k.vanE akkor k.melyik:=i
  Keresés1:=k
Függvény vége.
```

Tömbök és függvények tételekben

- Példák típusfogalmak felhasználására:
  - Adatleírás, általánosított tömbös változat:

```
Típus
  TI = … indextípus …
  TE = … elemtípus …
  TEk=Tömb(TI:TE)
  TKeresettI=Rekord(vanE:Logikai, melyik:TI) [index-kereséshez]
  TTulFv=Függvény(Konstans e:TE):Logikai
```

  - Keresés, általánosított tömbös változat (a „hagyományos” megoldással összevetve):

```
Függvény Keresés1(Konstans t:TEk, tFv:TTulFv):TKeresettI
  Változó i:TI
          k:TKeresettI
  i:=Min'TI
  Ciklus amíg i<Max'TI és nem tFv(t(i))
    i:=Köv(i)
  Ciklus vége
  k.vanE:=tFv(t(i))
  Ha k.vanE akkor k.melyik:=i
  Keresés1:=k
Függvény vége.
```

```
Függvény Keresés1(Konstans t:TEk):TKeresettI
  Változó i:Egész
          k:TKeresettI
  i:=1
  Ciklus amíg i<N és nem T(t(i))
    i:=i+1
  Ciklus vége
  k.vanE:=T(t(i))
  Ha k.vanE akkor k.melyik:=i
  Keresés1:=k
Függvény vége.
```

Tömbök és függvények tételekben

- Példák típusfogalmak felhasználására:
  - Adatleírás, általánosított sorozatokhoz:

```
Típus
  TI = … indextípus …
  TE = … elemtípus …
  TEk=TInputSorozat(TE)
  TKeresettE=Rekord(vanE:Logikai, melyik:TE) [elem-kereséshez]
  TTulFv=Függvény(Konstans e:TE):Logikai
```

  - Keresés, általánosított sorozatokra megfogalmazott változat (és a „hagyományos” megoldás):

```
Függvény Keresés2(Konstans t:TEk, tFv:TTulFv):TKeresettE
  Változó e:TE
          k:TKeresettE
  e:=Első(t)
  Ciklus amíg nem Vége?(t) és nem tFv(e)
    e:=Köv(t)
  Ciklus vége
  k.vanE:=nem Vége?(t)
  Ha k.vanE akkor k.melyik:=e
  Keresés2:=k
Függvény vége.
```

```
Függvény Keresés2(Konstans t:TEk):TKeresettE
  Változó i:Egész
          k:TKeresettE
  i:=1
  Ciklus amíg i≤N és nem T(t(i))
    i:=i+1
  Ciklus vége
  k.vanE:=i≤N
  Ha k.vanE akkor k.melyik:=e
  Keresés2:=k
Függvény vége.
```

A kódolás „absztrakciójának” mélyítése

- A sorozat-típusok (Free)Pascal megvalósítása (összefoglalás)

```
Program AltalanosTomb;
  Uses
    …,OI_I1_Unit{=>OI_I1},OE_E1_Unit{=>OE_E
    OI_I2_Unit{=>OI_I2},OE_E2_Unit{=>OE_E2}
    …
    OTomb_Sablon_Unit{=>OTomb<OIndex,OElem
  Type
    OI_I1_E_E1k=Specialize OTomb<OI_I1,OE_E
    OI_I2_E_E2k=Specialize OTomb<OI_I2,OE_E
    …
  Var
   t1:OI_I1_E_E1k;
   t2:OI_I2_E_E2k;
…
Begin
  … t1:=OI_I1_E_E1k.Create;
  …
End.
```

```
Unit OTomb_Sablon_Unit
 {$mode objfpc}{$h+}
 Interface
  Type
    Generic OTomb<OIndex,OElem>=Class(TOb
    … OTomb mezők …
    … műveletek …
```

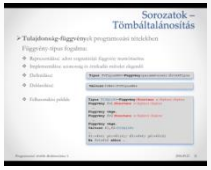
```
Unit OI_I1_Unit;
 {$mode objfpc}{$h+}
 Interface
  Type
   OI_I1 =Class
    … OI_I1 mezők …
   Public
    … OI_I1 osztály-függvények
     (Class Function Min …)
    Constructor Create;
    … OI_I1 további műveletek …
   End;
 Implementation
    … OI_I1 műveletek kifejtése …
 …
```

```
Unit OE_E1_Unit;
 {$mode objfpc}{$h+}
 Interface
  Type
   OE_E1 =Class
    … OE_E1 mezők …
   Public
    Constructor Create;
    … OE_E1 további műveletek …
   End;
 Implementation
    … OE_E1 műveletek kifejtése …
```

A kódolás „absztrakciójának” mélyítése

- A (Free)Pascal tulajdonság-függvénytípusa:

  A korábban elmondottak a kódra is érvényesek. Azaz



Típusdefiniálás:

```
Type TTulFv=Function(Const e:TE):Boolean;
```

Típusdeklarálás:

```
Var t:TTulFv;
…
Function Eldontes(Const s:TEk; Const tFv:TTulFv):Boolean;
  Var i:TI;
Begin
  i:=Min'TI;
  While (i<Max'TI) and not tFv(s[i]) do
    …
End;
```

Felhasználás:

```
Function ParosE(Const e:TE):Boolean;
…
Function PrimE(Const e:TE):Boolean;
…
t:=@ParosE; vanE:=Eldontes(s,t);
…
vanE:=Eldontes(s,@PrimE);
…
```

A kódolás „absztrakciójának” mélyítése

- **Házi feladat:**

  Az elemi tételek közül 2 elemi tétel algoritmusának megfogalmazása általánosított tömbre és általános sorozatra.