Algoritmusok és adatstruktúrák

Vizsgaminta (v_03.pdf) feladatok megoldásai

1. ELSŐ RÉSZ

Készítsen adatszerkezeti táblázatot és algoritmust (struktúra diagram vagy pszeudokód) az alábbi feladatokra!

1.1. Első feladat

Input adatként adott valahány darab dátum adat, mint ÉÉ.HH.NN formátumú sztring. Az adatok számát nem ismerjük, az adatmegadás végét egy üres sztring jelzi. Kérjük be az adatokat és írjuk ki az egyes hónapokba eső adatok %-os előfordulási arányát! (2p)

1.2. Második feladat

Ha az eredménylista (amelyen az egyes hónapok sorszáma és előfordulási aránya szerepel) az előfordulási arány szerint csökkenően rendezett és nem tartalmazza a 0 adatú hónapokat. (3p)

1.3. Megoldás

A két feladat összefügg, ezért a megoldásokat egyben adjuk meg. Használunk egy rekordtípust (HONAP), a hónap sorszámának és a hozzá tartozó előfordulási aránynak a tárolására.

Típus
HONAP Rekord
    SSZ Egész
    ARANY Egész

Funkció	Azonosító	Típus	Jelleg
Egy dátum adat	DATUM	Sztring	I
Az egyes hónapok adatai	HO	Egydimenziós HONAP tömb[12]	M,O
A dátumok darabszáma	DB	Egész	M
Segédváltozók	I,J	Egész	M
Két hónap cseréjéhez	CS	HONAP	M

```
/* Kezdőértékek beállítása */
for I<-1,12
  HO[I].SSZ<-I
  HO[I].ARANY<-0
DB<-0
/* Adatok bekérése és feldolgozása */
Ki:"Adja meg a dátumokat ÉÉÉÉ.HH.NN alakban (kilépés: üres sztring)!"
repeat
  Be:DATUM
  if DATUM<>""
    I<-VAL(COPY(DATUM,6,2))
    HO[I].ARANY<-HO[I].ARANY+1
    DB<-DB+1
until DATUM=""
```

```
if DB=0
  Ki:"Nem adott meg adatot!"
else
  /* A százalékos arányok meghatározása */
  for I<-1,12
    HO[I].ARANY<-HO[I].ARANY/DB*100
  /* Rendezés buborékrendezéssel */
  for I<-1,11
    for J<-12,I+1,-1
      /* Arány szerint csökkenően */
      if HO[J].ARANY>HO[J-1].ARANY
        CS<-HO[J]
        HO[J]<-HO[J-1]
        HO[J-1]<-CS
  /* Eredménykiírás */
  for I<-1,12
    if HO[I].ARANY>0
      Ki:HO[I].SSZ,HO[I].ARANY
```

2. MÁSODIK RÉSZ

Tervezzen adatstruktúrát és készítsen feladatonként egy-egy teljesen paraméterezett szubrutint (adatszerkezeti táblázat és algoritmus (struktúradiagram vagy pszeudokód)) az alábbi feladatokra!

Egy fogadóirodában 10 ablaknál bonyolódnak a fogadások, amik valójában pénz be és kifizetések. Az ablakokat egyszerűen a sorszámukkal azonosítjuk. Az ablakok előtt sorban álló fogadók száma előre adott konstanssal nem korlátozható!

2.1. Első feladat

Érkezés: új kuncsaft érkezik és beáll a legrövidebb sor végére. A fizetés iránya (be ill. kifizetés) és mennyisége (forintban) véletlenszerűen generálandó! (3p)

A feladatok itt is összefüggnek, ezért a közös konstans és típus deklarációkat csak itt, az első feladat megoldásánál adjuk meg. A megoldásban egy 10 elemű tömböt (az egyes ablakokhoz), és egyirányban láncolt listákat használunk, ahol az egyes láncelemek egy fogadó adatait tárolják. A fogadó pénz adata előjeles, az előjel jelzi a fizetés „irányát" (pozitív: kifizetés, negatív: befizetés). A kapott ablak adatok helyesek (pl. a SORHOSSZ mező az adott ablak előtti sorban állók számát tárolja, az ELSO és UTOLSO mutatók a megfelelő fogadókra mutatnak, üres sor esetén NIL-re), ezeket a megfelelő módon használjuk, ill. módosítjuk (ezt jelzi az A paraméter O jellege).

Megjegyezzük, hogy az új fogadó rekordjának a megfelelő sor végére történő beláncolásánál használt változóhivatkozás (az értékadás baloldalán) az egyszerűbb (*A[I].UTOLSO).KOV alakban is megadható, de a plusz zárójelezés talán jobban kiemeli, hogy mit is szeretnénk hivatkozni.

Konstans
ABLAKDB 10
MAXPENZ 1000000

Típus
FOGADO Rekord
    PENZ Egész
    KOV FOGADO-ra mutató
ABLAK Rekord
    SORHOSSZ Egész
    ELSO FOGADO-ra mutató
    UTOLSO FOGADO-ra mutató
ABLAKOK Egydimenziós ABLAK rekordokból álló tömb[ABLAKDB]

Funkció	Azonosító	Típus	Jelleg
Az ablakok adatai	A	ABLAKOK	I,O
A művelet sikeressége	OK	Logikai	O
A legrövidebb sor hossza	MIN	Egész	M
A legrövidebb sor indexe	I	Egész	M
Az érkező fogadó	UJ	FOGADO-ra mutató	M
Segédváltozó	J	Egész	M

```
ERKEZES(A)
if VANHELY(MERET(FOGADO))
  OK<-igaz
  /* A legrövidebb sor meghatározása */
  MIN<-A[1].SORHOSSZ
  I<-1
  for J<-2,ABLAKDB
    if A[J].SORHOSSZ<MIN
      MIN<-A[J].SORHOSSZ
      I<-J
  /* Az új fogadó rekordjának elkészítése */
  HELYFOGLAL(UJ)
  /* Kifizetés */
  (*UJ).PENZ<-RANDOM(MAXPENZ)+1
  if RANDOM(2)=0
    /* Befizetés */
    (*UJ).PENZ<-(*UJ).PENZ*(-1)
  /* Beláncolás */
  if A[I].ELSO=NIL
    /* Az üres sor elejére */
    A[I].ELSO<-UJ
  else
    /* A sor végére */
    (*(A[I].UTOLSO)).KOV<-UJ
  A[I].UTOLSO<-UJ
  /* A sorbanállók száma */
  A[I].SORHOSSZ<-A[I].SORHOSSZ+1
else
  OK<-hamis
return OK
```

2.2. Második feladat

Terheltség: meghatározandó ablakonként és összesen a sorban állók száma. (2p)

Mivel az adatstruktúrát úgy definiáltuk, hogy az egyes ablak rekordok tárolják az előttük lévő sorban álló fogadók számát, ezért azokat már nem is kell meghatároznunk, így csak ezek összegét kell kiszámolnunk. Megjegyezzük azonban, hogy más adatstruktúra esetén (pl. ha nincs a SORHOSSZ mező az ABLAK rekordban) más algoritmust kellene készítenünk.

Funkció	Azonosító	Típus	Jelleg
Az ablakok	A	ABLAKOK	I
Az összes fogadó száma	OSSZ	Egész	M,O
Segédváltozó	I	Egész	M

```
TERHELTSEG(A)
OSSZ<-0
for I<-1,ABLAKDB
  OSSZ<-OSSZ+A[I].SORHOSSZ
return OSSZ
```