**Kocsis Ábel 2. beadandó** 2018. április 16.

FGSDV2

[fgsdv2@inf.elte.hu](mailto:fgsdv2@inf.elte.hu)

1. csoport

## Feladat

Egy társasház közös költségének 2014-ben történt befizetéseit egy szöveges állományban tároljuk. Minden lakás minden befizetése egy-egy sor az állományban, amelyik tartalmazza a lakás azonosítóját (sztring), a befizetés dátumát (EEEE.HH.NN formájú sztring) és összegét (természetes szám). Egy soron belül az adatokat elválasztójelek (szóközök, tabulátor jelek) határolják egymástól. A sorok a lakásazonosítók szerint, azon belül a befizetési dátumok időrendje szerint rendezettek. Adja meg egy olyan lakásnak az azonosítóját, amelyik a legkevesebb összeget fizette be azok közül, akik legalább háromszor fizettek.

Például az input fájl adatai: NagyCsalad 2014.02.11 1201

NagyCsalad 2014.02.13 1320

MagyarUr 2014.02.01 1230

## Specifikáció

A = (y: String, (fy: ℕ, l:𝕃), t:enor(Csalad))

Befiz = rec(azon:String, ido:String, befizmenny:ℕ)

Csalad = rec(azon:String, befizossz:ℕ, befizszam:ℕ)

Ef = (t = t0 ∧ tazon ↑ )

Uf = ( y = min e∈t0 e.befizossz)

e.befizszam ≥ 3

### Részfeladat: t bejáró specifikációja

t:enor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Csalad\* | first | next | end | current |
| x: infile  dx: Befiz  sx: Status  \_end: 𝕃  akt: Csalad | sx,dx,x:read;  next; | \* | \_end | akt |

A =

Ef = ( x = x’ ∧ sx = sx' ∧ dx = dx' ∧ xazon↑)

Uf = \_end = (sx' = abnorm) ∧

¬\_end → akt.azon = dx'.azon ∧

akt.befizossz = ∑ dx∈xdx.befizmenny ∧

dx.azon = dx'.azon

akt.befizszam = ∑ dx∈x 1

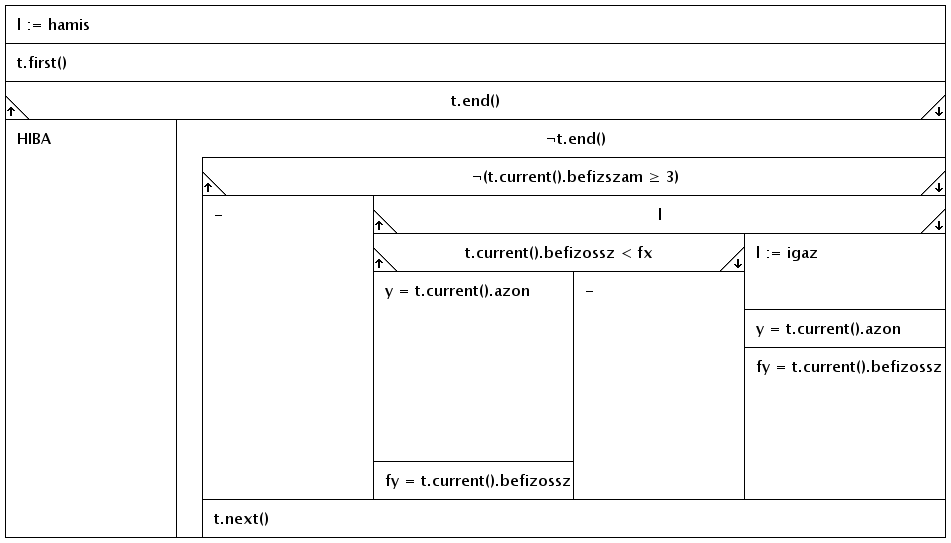
dx.azon = dx'.azon

### Részfeladat: t bejáró absztrakt programja

|  |  |  |
| --- | --- | --- |
| **összegzés** | | |
| t: enor(E) | ~ | t: enor (Csalad), x: infile (Befiz), sx, dx  first: ∅ |
| E | ~ | Csalad |
| f(e) | ~ | dx.azon = akt.azon |
| H, +, 0 | ~ | Csalad\*, +, 0 |

## Absztrakt program

**feltételes minimumkeresés**

****

## Implementáció

Az x szekvenciális inputfájl sorait bejáró felsorolót a *Beolv* osztállyal, a *Befiz* és *Csalad* típust a velük azonos nevű osztállyal valósítjuk meg.

A *Beolv* osztály *first()* és *next()* művelete addig olvas *Befiz* típusokat, amíg azokhoz azonos azonosító tartozik. A befizetések értékét összegzi, a befizetések darabszámát pedig számolja. Ha a fájl végére ért, az *sx* státuszt *abnorm*-ra állítja. Ez a státusz az *end()*, a beolvasott megfelelő azonosítójú rekordok a *current()* segítségével kérdezhetők le.

read()

t.current()

t.end()

t.next()

t.first()

main()

A program több állománybül áll: main.cpp, enor.h, enor.cpp

|  |  |
| --- | --- |
| **main.cpp** | **enor.h-enor.cpp** |
| main() | class Csalad  class Befiz  first()  next()  current()  end() |

## Tesztelés

A t bejárás tesztesetei:

**intervallum hossza szerint**

Egyetlen sorból álló fájl

Két sorból álló fájl

Sok sorból álló fájl

**felsorolandó tételek szerint**

egy felsorolandó

két felsorolandó

több felsorolandó

**fájl létezése szerint**

nemlétező fájl megnyitása

**bejárásra jellemző tesztesetek**

first()

next()

end()

A feltételes minimumkiválasztás tesztesetei:

**minimum helye szerint**

Elsőben a minimum

Utolsóban a minimum

Egyéb helyen a minimum

**feltételnek megfelelőek mennyisége szerint**

Egy megfelelő

Két megfelelő

Több megfelelő