Burian Sándor | AWXYHE

ELTE IK - 2017 IP-08ePROGEG

Programozás 2. beadandó dokumentáció

Tartalom

[Feladat 2](#_Toc483429482)

[Specifikáció 2](#_Toc483429483)

[Absztrakt program 3](#_Toc483429484)

[Implementáció 4](#_Toc483429485)

[Tesztelés 5](#_Toc483429486)

# Feladat

Egy szöveges állományban tároljuk egy évfolyam ZH eredményét. Az

állomány minden sorában egy-egy hallgató adatait találjuk: a hallgató

EHA kódját és mellette a 6 db feladatra kapott pontszámát. Az adatok

között szóköz(ök) és/vagy tabulátor jelek találhatók. Az egyes feladatokra 0-10-ig lehetett pontot kapni, ha valaki nem oldott meg egy feladatot, akkor X-et írtak a pont helyére. (Ez csak jelzés a feladat hiányára.) Az osztályzat az összpontszámtól függ: 20 ponttól kettes, 30-tól hármas, 40-től négyes, ötventől ötös. Feltehetjük, hogy az állomány helyesen van kitöltve. Listázza ki a hallgatók kódját és összpontszámát a konzolra, valamint mondja meg, hány ötös, négyes, hármas, kettes és egyes jegy született!

Például az input fájl adatai:

EEADAAI.ELTE X 9 6 0 10 5

MDADAAI.ELTE 10 9 7 8 10 9

NVADAAT.ELTE 9 10 10 8 9 X

# Specifikáció

A = (f: infile(Student), neptun: String, s: ℝ, notes: ℝ5)

Student = rec(neptun: String\*, points: {0..10}6 )

Ef = (f = f’)

Uf = (neptun, s, notes) =

# Absztrakt program

Nevezetes felsorolót használó összegzést futtató számlálás egyedi felsorolóval.

|  |  |
| --- | --- |
| **Összegzés** | **Számlálás** |
| E ~ points: {0..10}6 | E~ Student |
|  | f(e) ~ f(e.points) |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Állapottér | | First() | | Next() | | End() | |
| **Összegzés**  **f:** points: {0..10}6 ⇾ℤ | **Számlálás** | **Összegzés** | **Számlálás** | **Összegzés** | **Számlálás** | **Összegzés** | **Számlálás** | |
| E ~ points: {0..10}6 | E~ Student  t~f | - | Next() | - | s:=Összegzés(e)  notes[s] = notes[s]+1 | - | sf=abnorm | |
|  | f(e) ~ f(e.points) |

v:= points: {0..10}6

s:= 0

v.First()

⌉v.End()

s := s + f(v.Current())

v.Next()

notes := <>

f.First()

⌉f.End()

\ Összegzés(e) ≥ 50 \ Összegzés(e) ≥ 40 \ Összegzés(e) ≥ 30 \ Összegzés(e) ≥ 20 \Egyéb

notes[5] := notes[5] +1 notes[4] := notes[4] +1 notes[3] := notes[3] +1 notes[2] := notes[2] +1 notes[1] := notes[1] +1

f.Next()

# Implementáció

Az f inputfájl sorait bejáró felsorolót SeqInFilet osztállyal valósítjuk meg. A Student típus gyakorlatilag azonnal feldolgozódik, így önállóan nem létezik. A SeqInFile osztály First() és Next() művelete egy sf,e,x:read, mely a szöveges állomány következő sora alapján állítja elő az e Strudent rekordot. Ha nincs következő sor akkor az sf abnormra állítódik és ekkor elérkezünk az End()-hez.

|  |  |
| --- | --- |
| **main.cpp** | **seqinfile.h – seqinfile.cpp** |
| Main() | Class SeqInFile  Enum Status{abnorm, norm};  First()  Next()  End() |

# Tesztelés

Üres fájl:

Bemenet: ures.txt

kimenet: Ures a fajl, vagy csupa hibas rekordot tartalmaz! A feldolgozas leall!

Helyes adatok:

Bemenet: 1.txt

Kimenet:

ALMA 21

KORTE 11

DIO 30

Jegyek:

1: 1

2: 1

3: 1

4: 0

5: 0