Kis Gergely Domonkos **2. beadandó/ 25.feladat** 2019.04.05
VMT982
tianarath30@gmail.com
6. csoport

Feladat

Egy általános iskola alsó tagozatán papírgyűjtő versenyt rendeztek. A verseny 2018. szeptember 1-től, december 31-ig tartott. Feljegyezték a verseny adatait, és egy szöveges állományban tárolták el. A fájl egy sorának felépítése: elsőként a tanuló neve szerepel (két vagy több szóközök nélküli sztring), majd az osztálynak az azonosítója (1-4 számjeggyel kezdődő, szóközt nem tartalmazó sztring, például 1a, 2b, 4c), majd a papírgyűjtés adatai: dátum-súly (a dátum EEEE/HH/NN alakú sztring, a súly egy pozitív valós szám: a gyűjtött papír súlya kilogrammban megadva) formájában. A sor dátum szerint rendezett. Az adatok szóközökkel vagy tabulátorjelekkel vannak egy soron belül elválasztva. A szöveges állomány sorait osztály-azonosító szerint rendezték. Feltehetjük, hogy a szöveges állomány helyesen van kitöltve. Példa az állomány egy sorára:

Nagyon Szorgalmas Eszter 4c 2018/09/10 4.5 2018/09/22 3.5 2018/11/05 1.2

(1) Listázzuk ki azokat a tanulókat (nevüket és osztályukat), akik legalább egyszer, több mint 5 kg papírt hoztak!

(2) Hány olyan osztály van, melynek minden tanulója (aki a versenyben részt vett) legalább egyszer, több mint 5 kg papírt hozott?

(1) Részfeladat megoldása

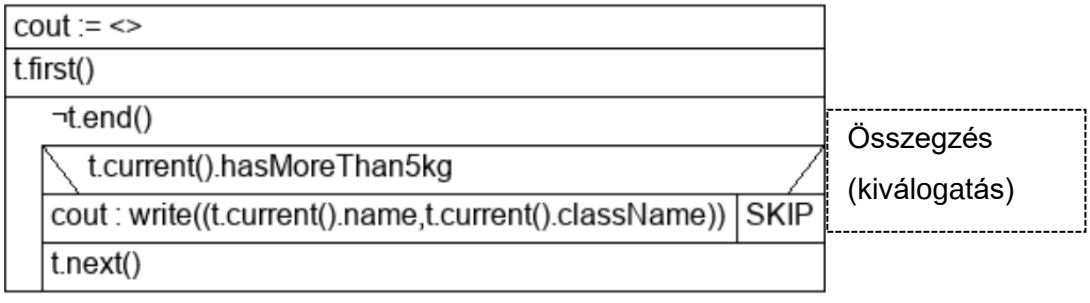
Főprogram terve

A = (t : enor(StudentPaperData), cout : outfile(StudentPaperDataOut))

StudentPaperData = **rec**(name : $\mathbb{S}$, className : $\mathbb{S}$, hasMoreThan5kg : $\mathbb{L}$)
StudentPaperDataOut = **rec**(name : $\mathbb{S}$, className : $\mathbb{S}$)

Ef = (t = t')

Uf = (cout = $\underset{\text{e.hasMoreThan5kg}}{\oplus_{e \in t'}} < (e.name, e.className) >$



Kis Gergely Domonkos **2. beadandó/ 25.feladat** 2019.04.05
VMT982
tianarath30@gmail.com
6. csoport

Diákok papírgyűjtési adatainak felsorolója

Felsoroló:

t:enor(StudentPaperData) Contribution = **rec**(name: 𝕊,className: 𝕊,data: 𝕊)

StudentPaperData*	first()	next()	current() : StudentPaperData	end() : 𝕃
x : infile(Contribution) dx : Contribution sx : Status act : StudentPaperData end : 𝕃	sx,dx,x:read next()	lásd külön	**return** act	**return** end

StudentPaperData = **rec**(name: 𝕊,className: 𝕊,hasMoreThan5kg: 𝕃)

Az enor(StudentPaperData) first() művelete beolvas egy sort a read() művelet segítségével, majd meghívja a next() műveletet.

A read() művelet az „x” szekvenciális inputfájl egy sorát beolvassa, és az „sx” státuszváltozót állítja „norm” értékre sikeres olvasás esetén, „abnorm” értékre sikertelen olvasás esetén.

Először beolvassa az egész sort egy változóba („iss”) , majd azt kezdi el feldolgozni. Egy másik változóba kezdi el gyűjteni a beolvasott nevet, mivel több tagból is állhat („oss”). Az „iss” változó értékét szavanként összegzi az „oss” változóba, ameddig a beolvasott érték első karaktere nem egy 1 és 4 között lévő szám, azaz a tanuló osztályának azonosítója. Ezt követően a „dx” változó „name” mezőjének a képzett összeget adja („oss”-teljes név), a „className” mezőjének pedig az utoljára beolvasott osztályazonosítót. A „data” mező a hátralévő karakterlánc értékét kapja meg. Amennyiben bármelyik lépés sikertelen a „sx” változó „abnorm” értéket kap.

A next() művelet először beállítja az „end” változó értékét az „sx” státuszváltozó értékétől függően, igaz értékre amennyiben „abnorm” , és hamis értékre amennyiben „norm”. Ezt követően beállítja a felsorolás aktuális elemének („act”) „name” és „className” mezejét a beolvasott „dx” változó „name” és „className” mezőjének értékeire. Utána a „dx” változó „data” mezőjét szavanként (szóközökkel elválasztva) elkezdi feldolgozni a következőféleképpen:

Kis Gergely Domonkos **2. beadandó/ 25.feladat** 2019.04.05
VMT982
tianarath30@gmail.com
6. csoport

Egyszerre két szót dolgoz fel: az első szó a hozott papír dátuma, a második pedig a hozott papír súlya. Ezeken a párokon végez pesszimista lineáris keresést (eldöntés) úgy, hogy amennyiben talál egy olyan párt, amelyhez tartozó súly nagyobb, mint 5 kg akkor a felsorolás aktuális elemének („act") „hasMoreThan5kg" mezejét igazra állítja, különben pedig hamison hagyja. A feldolgozást követően végez mégegy olvasást „read()", ezzel léptetve a felsorolást.

Next Művelet:

A^{next} = (x:infile(Contribution),dx:Contribution,sx:Status,act:StudentPaperData,end: $\mathbb{L}$)

Contribution = **rec**(name: $\mathbb{S}$,className: $\mathbb{S}$,data: $\mathbb{S}$)

StudentPaperData = **rec**(name: $\mathbb{S}$,className: $\mathbb{S}$,hasMoreThan5kg: $\mathbb{L}$)

Ef^{next} = (x = x' ∧ dx = dx' ∧ sx = sx')

Uf^{next} = (end = (sx' = abnorm) ∧ ($\neg$end ⟶ (act.name = dx'.name

∧ act.className = dx'.className ∧ (act.hasMoreThan5kg =

$\mathbf{SEARCH}_{i\in[1..|dx'.data|] \text{ és } i \bmod 2 = 0}\ (dx'.data[i] > 5)$))

end := sx = abnorm	
¬end	
act.name := dx.name	SKIP
act.className := dx.className	
ss := dx.data	
exist := false	
date := ss[1], weight := ss[2]	
i=3..|ss|-1 && ¬exist	
weight > 5	
exist := true	SKIP
date, weight := ss[i],ss[i+1]	
i = i+1	
act.hasMoreThan5kg := exist	
sx,dx,x:read	

A megvalósítás kódja:

ha ss egy istringstream

ss >> date >> weight

ss >> date >> weight

Pesszimista lineáris keresés (eldöntésre)

Kis Gergely Domonkos **2. beadandó/ 25.feladat** 2019.04.05
VMT982
tianarath30@gmail.com
6. csoport

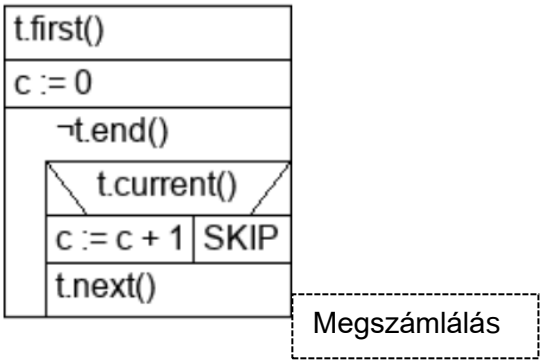
(2) részfeladat megoldása

Főprogram terve

A = (t : enor(𝕃), c : ℕ)

Ef = (t = t')

Uf = (c = $\sum_{e \in t'} 1$)



Iskolai osztályok felsorolója

Felsoroló:

t:enor(𝕃) StudentPaperData = **rec**(name: 𝕊,className: 𝕊,hasMoreThan5kg: 𝕃)

𝕃 *	first()	next()	current() : 𝕃	end() : 𝕃
y : enor(StudentPaperData) dy : StudentPaperData act : 𝕃 end : 𝕃	y.first() next()	lásd külön	**return** act	**return** end

Az enor(𝕃) „Iskolai osztályok felsorolója”-nak first() művelete meghívja az „y” adattagjának a first() műveletét, amely lépteti annak a felsorolását, és beállítja annak az „end” értékét. Ezt követően meghívja a next() műveletet.

A next() művelet az „y” adattag „end” mezőjének függvényében állítja be a felsoroló „end” mezejét, hiszen a külső felsorolás is akkor ér véget, amikor a belső. Ezután egy lineáris keresést hajt végre, ami addig megy, amíg vagy véget nem ér a felsorolás vagy a belső felsorolt elem „className” mezője eltér a kezdeti értéktől. Ezen belül, ha talál a belső felsorolásban a

„hasMoreThan5kg” mezőben hamis értéket, akkor a felsorolás „act” mezejébe hamis érték kerül, ha nem talál ilyet akkor pedig igaz.

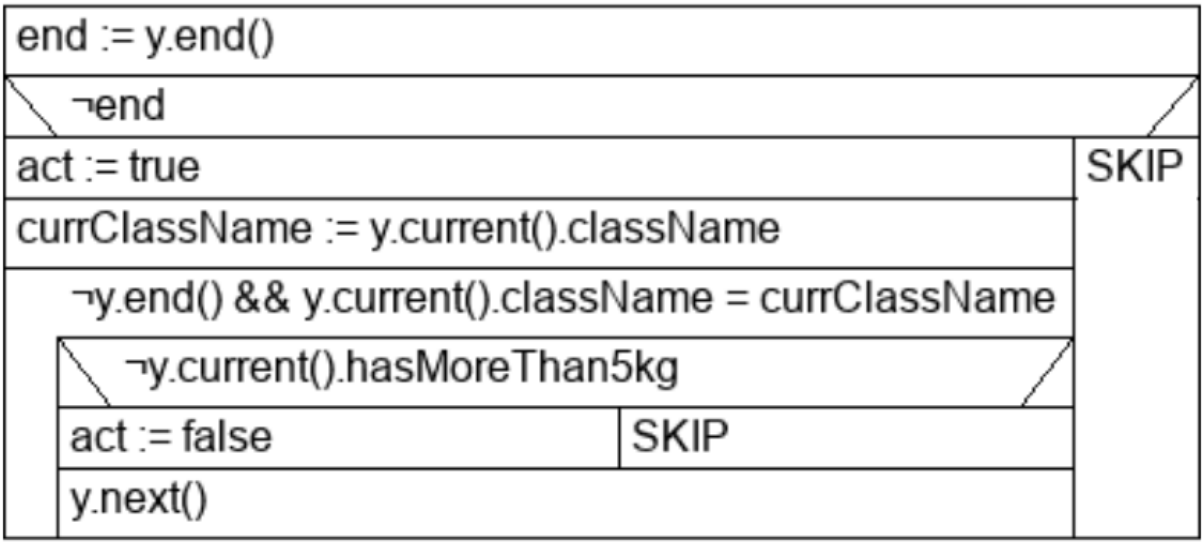
Next Művelet:

A^{next} = (y : enor(StudentPaperData), dy: StudentPaperData, act: $\mathbb{L}$, end: $\mathbb{L}$)

StudentPaperData = **rec**(name: $\mathbb{S}$,className: $\mathbb{S}$,hasMoreThan5kg: $\mathbb{L}$)

Ef^{next} = (y = y’ ∧ dy = dy’)
Uf^{next} = (end = (y’.end()) ∧ (¬end ⟶ akt =

$\forall \mathbf{SEARCH}_{dy \in y'}^{dy.className = y'.current().className}(dy.\,hasMoreThan5kg)$))



Kis Gergely Domonkos **2. beadandó/ 25.feladat** 2019.04.05
VMT982
tianarath30@gmail.com
6. csoport

Tesztelési terv:

A megoldásban négy programozási tételt alkalmaztunk:

összegzés (kiválogatás),megszámlálás, lineáris keresés (eldöntésre), pesszimista lineáris keresés (eldöntésre)

A tesztesetekhez tartozó fájlok a ./tests/ mappában érhetőek el, a tesztek pedig az ./src/tests.h fájlban.

(1) Feladat| Az összegzés (kiválogatás) tesztesetei

Minden tesztesethez 2 fájl tartozik:

Egy **bemeneti** (source) fájl: **Test%%S.txt**

Egy **várt eredmény** (expected) fájl: **Test%%E.txt**

A tesztelő összehasonlítja a bemeneti fájlokhoz tartozó kimeneteket a várt eredményeket tartalmazó fájlokkal és hiba esetén jelenti a különbségeket, illetve azok számát.

-Tanulók, akik legalább egyszer több, mint 5kg papírt hoztak

- **bemenet hossza** szerint és **eredmények száma** szerint:
    1. nem létező bemeneti fájl
    2. 0 tanuló 0 eredmény - üres fájl
    3. 1 tanuló – 0 eredmény
    4. 1 tanuló – 1 eredmény
    5. 2 (több tanuló) - 0 eredmény
    6. 2 (több tanuló) – 1 eredmény
    7. 2 (több tanuló) – 2 eredmény
- kiválogatott **elemek elhelyezkedése** szerint (mindegyik több tanulós bemenet)
    8. eredmény a bemenet elején
    9. eredmény a bemenet közepén
    10. eredmény a bemenet végén
- Bemeneti tanulók nevének hosszai szerint (2,több)

Kis Gergely Domonkos **2. beadandó/ 25.feladat** 2019.04.05
VMT982
tianarath30@gmail.com
6. csoport

11.

(2) Feladat| A megszámlálás, lineáris keresés tesztesetei

Minden tesztesethez 1 fájl tartozik:

Egy **bemeneti** (source) fájl: **Test%%S.txt**

A tesztelő összehasonlítja a bemeneti fájlokhoz tartozó teszteredményeket a várt eredményekkel. Mivel a várt eredmény maga egy szám, ezért a tesztelőbe van belekódolva, nem pedig külön fájlban.
- Osztályok száma, ahol minden tanuló legalább egyszer több, mint 5 kg papírt hozott

- **Bemenet hossza** (osztályok száma) és **eredmény** (db) szerint
  Az eredmények számának ellenőrzésébe van **beépítve a lineáris keresés** (eldöntésre) tesztelése.
  12. Nem létező bemeneti fájl
  13. 0 osztály,0 tanuló – üres fájl – eredmény: 0
  14. 1 osztály, 1 tanuló – eredmény: 0
  15. 1 osztály, 1 tanuló – eredmény: 1
  16. 1 osztály, 2 (több) tanuló – eredmény: 0 (egyikre sem igaz)
  17. 1 osztály, 2 (több) tanuló – eredmény: 0 (nem mindre igaz)
  18. 1 osztály, 2 (több) tanuló – eredmény: 1 (mindre igaz)
  19. 2 (több) osztály – több tanuló – eredmény: 0
  20. 2 (több) osztály – több tanuló – eredmény: 1
  21. 2 (több) osztály – több tanuló – eredmény: 2

Közös|Pesszimista Lineáris (eldöntés) keresés tesztesetei

1 beolvasott sorban történő pesszimista lineáris keresés (eldöntésre) tesztjei. Ellenőrzi, hogy ha egy tanulóhoz több papírgyűjtési adat is tartozik, akkor van-e köztük 5kg-nál nagyobb.

Minden tesztesethez 1 fájl tartozik:

Egy **bemeneti** (source) fájl: **Test%%S.txt**
A tesztelő összehasonlítja a bemeneti fájlokhoz tartozó teszteredményeket a várt eredményekkel. A várt eredmény mindig vagy 0 – nincs 5kg-nál nagyobb - vagy 1 – van 5kg-nál nagyobb. Minden tesztfájlban maximum 1 ilyen adat lehet, és **annak a létezését, és különböző helyeit teszteljük.**

22. | eredmény: 0 – nincs 5kb-nál nagyobb adata
23. | eredmény: 1 – van, az első adat
24. | eredmény: 1 – van, a második (középső) adat
25. | eredmény: 1 – van, a harmadik (utolsó adat)