# 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

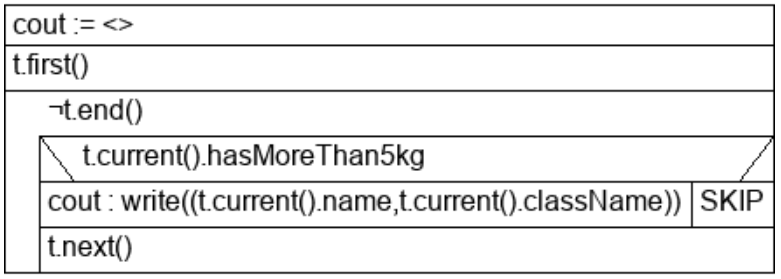
## Főprogram terve

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

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

StudentPaperDataOut = **rec**(name : 𝕊, className : 𝕊)

*Ef* = ( t = t’ )

*Uf =* ( cout 

## 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 az „x” szekvenciális inputfájlból, és amennyiben a beolvasás sikeres az „sx” státuszváltozót „**norm**” értékre állítja, amennyiben sikertelen az „sx” státuszváltozót „abnorm” értékre állítja be. Ezt követően 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”), 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:

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 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:

*Anext* = (x:infile(Contribution),dx:Contribution,sx:Status,act:StudentPaperData,end: 𝕃)

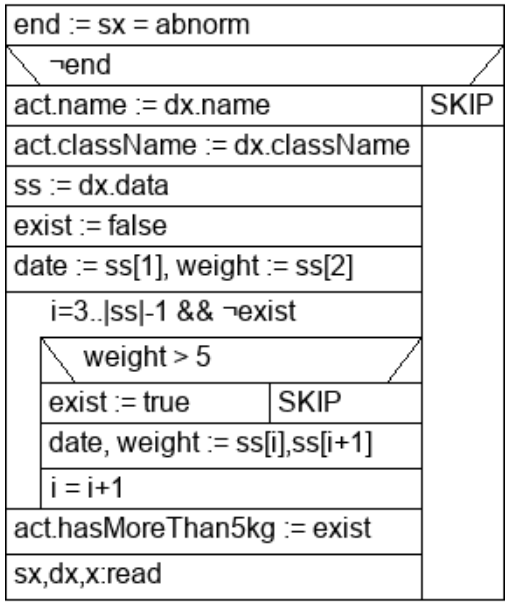
Contribution = **rec**(name: 𝕊,className: 𝕊,data: 𝕊)

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

*Efnext =* ( x = x’ ꓥ dx = dx’ ꓥ sx = sx’ )

*Ufnext =* ( end = (sx’ = abnorm) ꓥ (¬end ⟶ (act.name = dx’.name

ꓥ act.className = dx’.className ꓥ (act.hasMoreThan5kg =

 ))

*A megvalósítás kódja:*

*ha ss egy istringstream*

*ss >> date >> weight*

*ss >> date >> weight*

# (2) részfeladat megoldása

## Főprogram terve

*A* = ( t : enor(𝕃), c : ℕ)

*Ef* = ( t = t’ )

*Uf =* ( c )

## 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 sy : Status act : 𝕃 end : 𝕃 | y.first()  next() | lásd külön | **return** act | **return** end |