"Programozás"
beadandó feladat:
4. feladat

Készítette: Bárdosi Bence
Neptun-azonosító: VY9NJN
E-mail: bardosi.bence@gmail.com

2017-03-06

Tartalom

1 Dokumentáció

1.1 Feladat

Madarak életének kutatásával foglalkozó szakemberek n különböző településen m különböző madárfaj előfordulását tanulmányozzák. Egy adott időszakban megszámolták, hogy az egyes településen egy madárfajnak hány egyedével találkoztak. Volt-e olyan település, ahol mindegyik madárfaj előfordult?

1.2 Specifikáció

$$\mathbf{A} = (adat : \mathbb{N}^{nxm}, l : \mathbb{L})$$

$$\mathbf{Ef} = (adat = adat')$$

$$\mathbf{Uf} = \left(Ef \wedge (l, \_) = \overset{n}{\underset{i=1}{\text{SEARCH}}}\, mind(i) \right)$$

$$\mathbf{ahol}\, mind(i) : \mathbb{N} \to \mathbb{L}$$

$$\mathbf{és}\, \forall i \in [1..n] : mind(i) = \forall \overset{m}{\underset{j=1}{\text{SEARCH}}}\, adat[i,j] > 0$$

1.3 Algoritmus

A feladatot a lineáris keresés, az alfeladatot az optimista lineáris keresés programozási tételeire vezetjük vissza.

Lin. ker.

Tétel		Feladat
m	$\leftarrow$	1
n	$\leftarrow$	n
ind	$\leftarrow$	-
$\beta(i)$	$\leftarrow$	$mind(i)$

$l, i := hamis, 1$	
$\neg l \wedge i \leq n$	
	$l := mind(i)$
	$i := i + 1$

Opt. lin. ker.

Tétel		Feladat
m	$\leftarrow$	1
n	$\leftarrow$	m
i	$\leftarrow$	j
$\beta(i)$	$\leftarrow$	$adat[i,j] > 0$

l:=mind(i)

$l, j := true, 1$	
$l \wedge j \leq m$	
	$l := adat[i,j] > 0$
	$j := j + 1$

1.4 Implementáció

1.4.1 Adattípusok megvalósítása

A tervben szereplő mátrixot `vector<vector<int>>`-ként deklaráljuk. Mivel a vektor 0-tól indexelődik, azért a tervbeli ciklusok indextartományai a `0..n{1` és a `0..m{1` intervallumra módosulnak, ahol a `n`-re `t.size()` alakban, `m`-re pedig `t[i].size()` alakban hivatkozhatunk.

1.4.2 Bemenő adatok formája

Az adatokat be lehet olvasni egy szöveges állományból vagy meg lehet adni billentyűzetről. A program először megkérdezi az adatbevitel módját, majd a szöveges állományból való olvasást választva bekéri az állomány nevét. A billentyűzetről vezérelt adatbevitelt a program párbeszéd-üzemmódban irányítja, és azt megfelelő adat-ellenőrzésekkel vizsgálja. A szöveges állomány formája kötött, arról feltesszük, hogy helyesen van kitöltve, ezért ezt külön nem ellenőrizzük. Az első sor a városok és a madárfajok számát tartalmazza, szóközökkel vagy tabulátor jelekkel elválasztva. Ezt követően következnek a városonként megfigyelt madárfajok egyedeinek száma kötött sorrendben. A fájlt egy sorvége jel zárja.

1.4.3 Program váz

A program több állományból áll. A read csomag (`read.h, read.cpp`) felel az adatok helyes beolvasásáról és azok ellenőrzéséről. A "madar" csomag (`madar.h, madar.cpp`) felel feladat megoldásáért. Végül a "teszt" csomag (`catch.hpp, test.h, test.cpp`) az egységteszt megszervezéséért felel. A csomagokban található függvények/állományok feladatai az alábbi táblázatokból olvashatók.

Read csomag

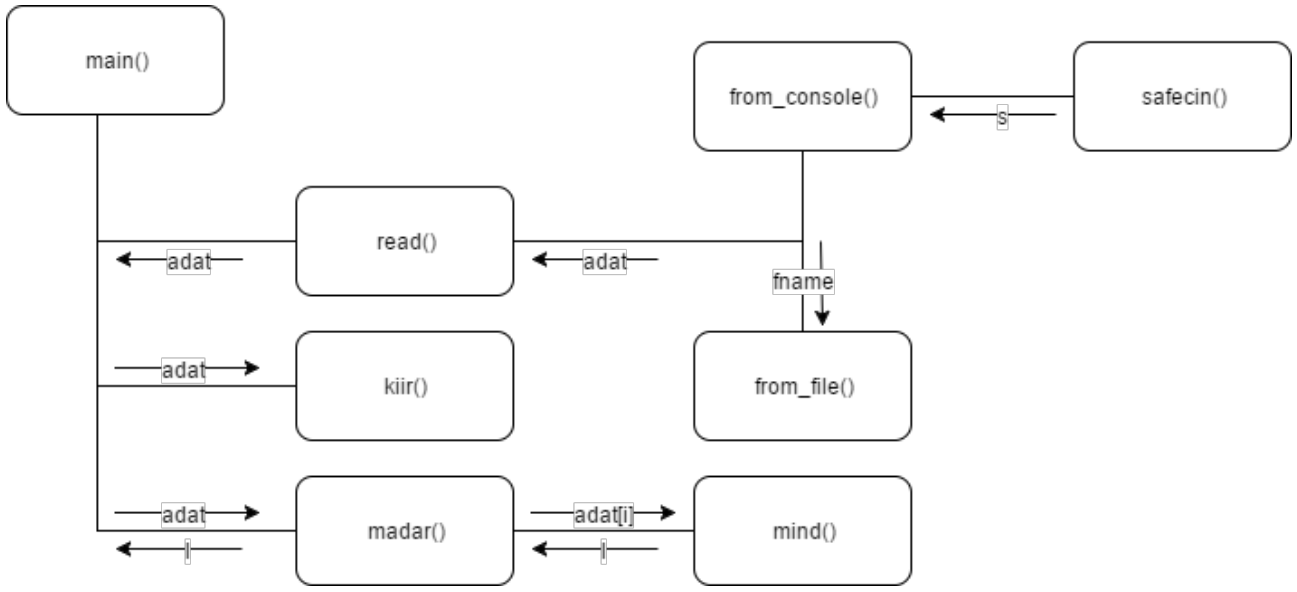
Függvény	Feladat
read	Megkérdezi a felhasználótól, hogy milyen módon kívánja az adatokat bevinni.
from_file	Megnyitja a megadott szöveges állományt,és beolvassa azadatokat.
from_console	Párbeszédes formában bekéri a felhasználótól az adatokat.
safecin	Ellenőrzi, hogy a kapott adat természetes szám-e. Hiba esetén értesíti a felhasználót.

Madar csomag

Függvény	Feladat
madar	Megadja, hogy az adott mátrixban létezik-e megfelelő tulajdonságú város.
mind	Megadja, hogy az adott tömbben minden madárfajból előfordul-e legalább 1.

Test csomag

Állomány	Feladat
test.h	Tartalmazza azt a flag-et, amely megszervezi, hogy programfutás, vagy egységteszt forduljon.
test.cpp	Egységteszteket tartalmazó állomány
catch.hpp	Külső könyvtár, mely megszervezi az automatikus egységtesztet.



1.5 Tesztelés

1.5.1 A feladat specifikációjára épülő (fekete doboz) tesztesetek:

Megszámlálás tétel tesztesetei:	
intervallum hossza szerint:	
1. *nulla* hosszú:	Egyetlen nap sincs be1.txt: [] - válasz: 0
2. *egy* hosszú:	Egyetlen nap be2.txt: [5.2] - válasz: 0
3. *kettő* hosszú:	Kettő, a feltételnek eleget nem tevő nap be3.txt: [5.2, 0] - válasz: 0 Kettő, a feltételnek eleget tevő nap be3.txt: [0, -2] - válasz: 1
4. *több* hosszú:	Több nap be5.txt: [0, -2, 0, 5, 6, 0, -0.5] - válasz: 2
intervallum eleje szerint:	
	Sorozat elején található csak a feltételnek megfelelő nappár be6.txt: [0, -2, 0, 5, 6, 0] - válasz: 1
intervallum vége szerint:	
	Sorozat végén található csak a feltételnek megfelelő nappár be7.txt: [0, 2, 0, 5, 6, 0, -2] - válasz: 1
tételre jellemző esetek szerint:	
1. Egyetlen megfelelő nappár van:	be7.txt: [0, 2, 0, 5, 6, 0, -2] - válasz: 1
2. Nincs megfelelő nappár:	be8.txt: [0, 1, 2, 3, 4] - válasz: 0
3. Egy ”0” napot több ”negatív” nap követ:	be9.txt: [1, 2, 0, -1, -2, -1] - válasz: 0

1.5.2 A megoldó programra épülő (fehér doboz) tesztesetek:

- Hibás vagy nem létező állománynév megadása.
- Állomány nevének megadása parancssorból.
- Olyan állomány olvasása, ahol egy sorban több érték is található egyetlen illetve több szóközzel és/vagy tabulátor jellel elválasztva (be10.txt).
- Olyan állomány olvasása, ahol minden érték külön sorban van (be9.txt).
- Olyan állomány olvasása, ahol az utolsó sort nem zárja sorvége jel, és éppen ennek a sornak a tartalma határozza meg az eredményt (adat: [1, 2, 3, 0, -1] – válasz: 1) (be11.txt).
- Főprogram ciklusának ellenőrzése: olyan bemenő adatokkal, amelyekre a ciklus egyszer sem fut le (Pl: be1.txt), pontosan egyszer fut le (Pl: be3.txt), vagy többször lefut(Pl:be5.txt).