Feladat

Egy kutya kiállításon n kategóriában m kutya vesz részt.   
Minden kutya minden kategóriában 0 és 10 közötti pontszámot kap.

Van-e olyan kategória, ahol a kategória nyertesei között holtverseny (azonos pontszám) alakult ki?

# Specifikáció

Tegyük fel, hogy kategóriákra lebontva rendelkezésre állnak az értékelések. Azt kell megállapítanunk, hogy van-e olyan kategória, ahol van a maximummal megegyező pontszám. Ez tehát egy *pesszimista lineáris keresés*.

A bemenő adatokat fogjuk fel tehát úgy, hogy adott egy elemű mátrix, ahol a kategóriák száma m pedig a kutyák száma.

ahol az egy olyan függvény lesz, ami akkor ad igazat, ha a paraméterül kapott sorban található olyan szám, ami megegyezik a sor maximumával.

Ez alapján:

és

Az kiszámításához jól láthatóan szükség lesz a sorozat elemeinek *maximumára*, ezt egy külön alprogramban fogjuk megírni, amit az számításakor függvényszerűen meg is fogunk tudni hívni.

Ez alapján:

és

Ez alapján specifikálhatom a belső tételt is ( kiszámítása):

# Visszavezetés és algoritmus

## Külső tétel:

pesszimista lineáris keresés

|  |  |  |
| --- | --- | --- |
|  | ~ |  |
|  | ~ |  |
| ~ elhagyva | | |
|  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | |  |  |
|  | | | |  |
|  | |  | |  |
|  | |  | |  |

## Belső tétel:

pesszimista lineáris keresés

[m..n] ~ [1..n]

β(i) ~ max (mx[i] )=mx[i][j]

ind ~ elhagyva

|  |  |  |
| --- | --- | --- |
| i,l := 1, ↑ | | i: N |
| l Λ i <= n | |  |
|  | l :=(max (mx[i] )=mx[i][j] ) |  |
|  | i := i + 1 |  |

## Belső tétel:

maximum keresés

|  |  |  |
| --- | --- | --- |
|  | ~ |  |
|  | ~ |  |
|  | ~ |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | |
|  |  |  |  |  |
|  | | | | |
|  | | | | |
|  |  | | | |
|  |  | | SKIP | |

# Implementáció

A programot C++ nyelven valósítottuk meg.

## Adattípusok

A feladat megfogalmazásakor definiált mátrixot, egy olyan vectorral fogjuk reprezentálni, aminek elemei maguk is vectorok.

Az adatokat az változóba olvassuk be, melynek típusa tehát vector<vector<int>>. Az osztályzatok 0 és 10 közötti egész számok lehetnek, ezt a feltételt a beolvasás során ellenőrizzük.

A mátrix vízszintes sorai a kategóriákat jelentik, a függőleges sorai pedig a kutyákat. A beolvasás során a soron következő értékelést a vector.push\_back() metódusával szúrjuk a megfelelő kategóriát jelképező vector végére.

Az beolvasása előtt még meghatározunk egy nevű segédvektort is.

A struktogrambeli nem más, mint az mérete, azaz az matrix.size(), az pedig az i-edik sorozatra az osztalyzatok[i].size() kifejezés értékével fogalmazható meg, amely minden sorban ugyanaz az érték.

A tömbindexelést minden esetben 0-tól végezzük.

## Adatok beolvasása

Billentyűről való beolvasás esetén először a kategóriák és a kutyák számát kérjük be, majd ennek adatai alapján fel tudjuk építeni az értékelések „mátrixát” is.

A beolv\_matrix nevű függvényt használjuk.

### Kategóriák és kutyák:

A kategóriák és a kutyák bekérésénél, egy nullánál nagyobb természetes számot várunk. Addig kérjük be az adatokat, amíg nem kapunk helyes értéket.

input:

kategóriák száma:5

kutyák száma: 6

Hibalehetőségek:

* A megadott kategóriák és kutyák értéke nem nagyobb mint 0 természetes szám

Ezekben az esetekben az adatokat újra bekérjük.

### Az értékelések (matrix) beolvasása:

Megadhatjuk fájlból vagy billentyűzetről. A beolvasás mechanizmusa, fájl esetén meg kell adni a fájlnevet, ellenőrizzük a fájl meglétét és megnyithatóságát, majd feldolgozzuk azt. Billentyűs beolvasáskor pedig egyesével kell megadnunk az adatokat és minden hibázáskor van lehetőségünk javítani.

Az adatokat egyesével kell megadni, a megfelelő kategóriához és kutya indexéhez tartozva, az értékelés pedig egy 0 és 10 közé eső természetes érték. Kategóriánként az értékeket berakjuk egy vektorba, majd ha végig mentünk az adott kategórián, akkor berakjuk a matrixunkba.

Az adatokat tartalmazó fájl megfelelő formátumára egy példa:

5 6 7 2 3

5 7 1 2 5

(természetes szám–[0..10]-beli értékekből áll; a számok között egy vagy több WS karakterrel)

Billentyűs bekérés esetén, a megvalósított függvény neve beolv\_matrix, míg fáljból való bekérés esetén a függvény neve beolvfajl. Billentyűs beolvasás esetén nem vár paramétert, míg a fájlból való beolvasás esetén a beolvasandó fájl neve a paraméterünk.

A függvény hívása fájlos beolvasás esetére:

beolvfajl(fájlnév);

Billentyűs beolvasásra:

beolv\_matrix();

## A kimenet

A sikeres beolvasás és feldolgozás után a feladatra adott válasz kerül kiírása a konzolra.

## A projekt felépítése

A következő modulokat használjuk:

* *main* – a main.cpp forrásfájl, a program belépési pontja (main függvény) valamint egy specifikált részfeladat implementációja található itt. A main végzi a program vezérlését, a tételek hívását, és az eredmény kiírását.
* *beolvasás* – beolv.h fejlécállomány valamint beolv.cpp forrásállomány. A fentiekben részletezett beolvasó függvények találhatók meg itt.
* *iostream* – külső könyvtár, mely a *konzolos kommunikáció* eszközeit teszi elérhetővé.
* *fstream* – külső könyvtár, mely a *fájlkezelés*ben nélkülözhetetlen, a beolvasáshoz használjuk
* *vector*– külső könyvtár, az std::vector típushoz

## Tesztelés

Az alábbiakban megadok néhány érvényes és érvénytelen tesztesetet

* *felület és adatbekérés* tesztelése billentyűről
  + „*menü*” használatának során olyan értékek beírása, amit nem szabadna 🡪 a hibás adat újra bekérés
  + kategóriák-kutyák számának megadása *nem 0-nál nagyobb természetes szám*🡪a hibás adat újra bekérése
  + matrix adatainak megadása [0..10] *intervallumon kívül*🡪a hibás adat újra bekérése
* *adatbekérés tesztelése fájlból*
  + matrix – *nemlétező* fájlnév megadása [f0.txt] 🡪 hibajelzés, kilépés
  + matrix – *üres* fájl [f1.txt] 🡪 hibajelzés, kilépés
  + matrix – értékek megadása [0..10] *intervallumonkívül [f2.txt]* 🡪 az ilyen értékeket kihagyja