Feladat

Hány különböző szám található abban a mátrixban, amelynek elemei 0 és 99 közé eső egész számok lehetnek? A mátrix elemei egy szövegfájlban találhatóak: az első sorban a sorok illetve oszlopok száma szóközökkel elválasztva, utána soronként a mátrix egy-egy sorának elemei szóközökkel elválasztva.

Függvény absztrakciós megoldás

Specifikáció:

$A = matrix(\mathbb{Z}) \times \mathbb{N}$ $\qquad matrix(\mathbb{Z}) = \mathbb{Z}^{n \times m}$

$x \qquad s$

$B = matrix(\mathbb{Z})$

x'

$Q = (x = x')$

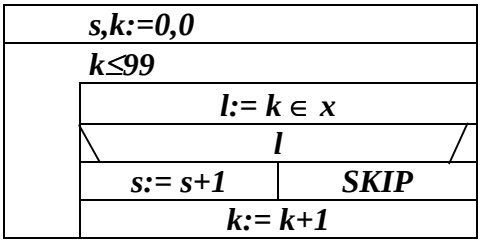
$$R = (x = x' \wedge s = \sum_{i=0}^{99} 1 \;)$$

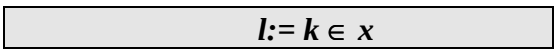
$k \in x$

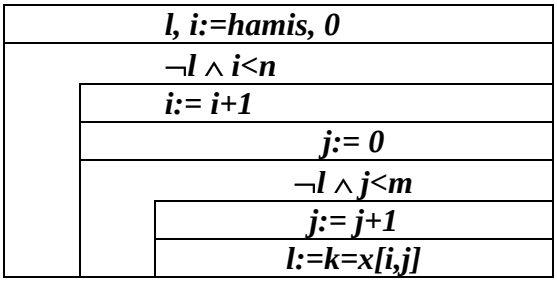
ahol

$k \in x = \exists i \in [1..n]: \exists j \in [1..m]: k = x[i,j]$

Absztrakt program:







Implementáció

Programváz:

```
int main()
{
  // Feladat változóinak deklarációja : x mátrix, s egész
  // Lokális változók deklarációja (2. megoldásban: h halmaz)
  // Beolvasás: az x mátrix méretének és elemeinek beolvasása
  // Absztrakt program: számlálás
  // Kiirás: s értékének kiírása
  return 0;
}
```

Függvény absztrakciós megoldás implementálása

Fő program:

```
/*****************************************************************/
/* Feladat: Különböző elemek megszámolása                        */
/*          egy szöveges állományban elhelyezett mátrixban       */
/*****************************************************************/

// Fordítási direktrívák

#include <iostream>
#include ”halmaz.h”

int main()
{
      int **x, n, m;

    // Beolvasás

      olv(x,n,m);

   // A mátrix eltérő elemeinek megszámolása

      s=0;
      for(int k=0;k<100;k++){
          if (benne(k,x,n,m)) s++;
      }
  // Kiirás

      cout<<"A megadott mátrixban "<<s
          <<" darab különböző szám található!"<<endl;

      char ch;
      cin>>ch;
      return 0;
}
```

matrix.h:

```
#ifndef MATRIX_H
#define MATRIX_H

void olv(int** &x, int &n, int &m);
bool benne(const int k, int** x, const int n, const int m);

#endif
```

matrix.cpp:

```
#include "matrix.h"
#include <fstream>

// Mátrix létrehozása és feltöltése szöveges állományból

void olv(int** &x, int &n, int &m)
{
      ifstream f("input.txt");
      if (f.fail()){
          cerr<<"Az input file nem létezik"<<endl;
          char ch;
          cin>>ch;
          exit(2);
      }

      f>>n>>m;
      x=new int* [n];
      for (int i=0;i<n;i++){
          x[i]=new int [m];
          for (int j=0;j<m;j++){
             f>>x[i][j];
          }
      }
}

// Eldönti a k elemről, hogy szerepel-e a mátrixban

bool benne(const int k, int** x, const int n, const int m)
{
     int i,j;
     bool l;

     l=false, i=-1;
     while(!l && i<n-1) {
          i++;
          j=-1;
          while(!l && j<m-1) {
              j++;
              l=k==x[i][j];
          }
     }

     return l;
}
```

Adat-absztrakciós megoldás

Specifikáció:

$A = matrix(\mathbb{Z}) \times \mathbb{N}$ $\qquad matrix(\mathbb{Z}) = \mathbb{Z}^{n \times m}$

$x \qquad s$

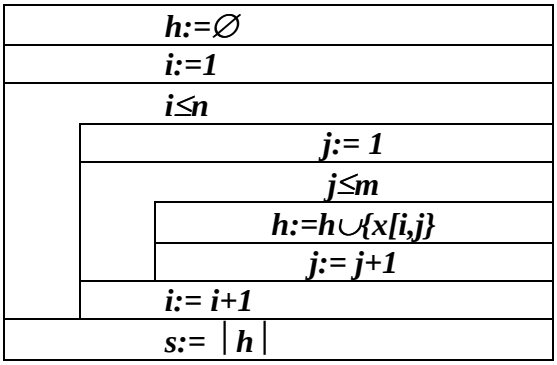
$B = matrix(\mathbb{Z})$

x'

$Q = (x = x')$

$R = (x = x' \wedge h \in 2^{\{0..99\}} \wedge h = \bigcup_{i=1}^{n} \bigcup_{j=1}^{m} x[i,j] \wedge s = |h|)$

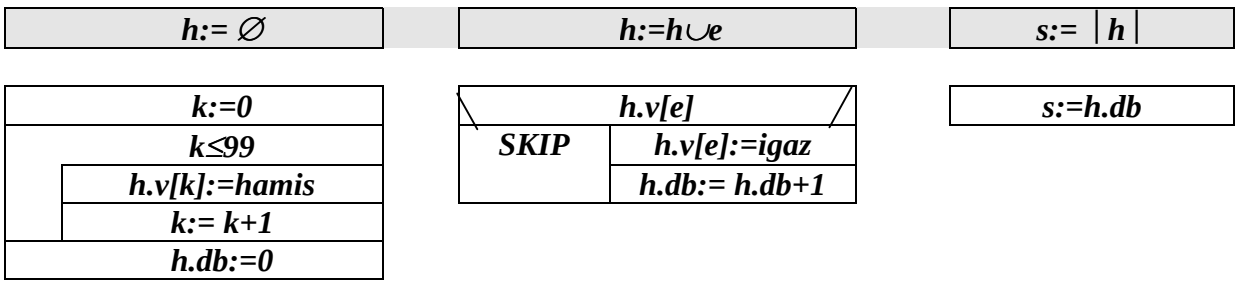
Absztrakt program:



Ebből leolvasható a ***h*** típusának specifikációja:

$(2^{\{0..99\}}, \{ h:=\emptyset, h:=h \cup e, s:=|h| \})$

Reprezentáljuk a típus értékeit (a halmazokat) *rec(v:vektor([0..99],𝕃); db:$\mathbb{N}_0$)* szerkezetű elemekkel: Ha a *k* szám benne van a halmazban, akkor a *v* vektor *k*-adik helyén egy igaz érték áll, a *db* pedig mindig a *v*-beli igaz értékek számát mutatja (ez utóbbi a típusinvariáns).
A típusműveleteket az alábbi programokkal implementáljuk :



Adat-absztrakciós megoldás implementálása

A modularizált megoldóprogram:

Fő program:

```
/*****************************************************************/
/* Feladat: Különböző elemek megszámolása                        */
/*          egy szöveges állományban elhelyezett mátrixban       */
/*****************************************************************/

// Fordítási direktrívák

#include <iostream>
#include <fstream>
#include "halmaz.h"

// Függvény deklaráció

void olv(int** &x, int &n, int &m);

int main()
{
      int **x, n, m;
      halmaz h;

  // Beolvasás

      olv(x,n,m);

  // Mátrix elemeinek átmásolása egy halmazba

      for(int i=0;i<n;i++)
          for(int j=0;j<m;j++)
              h.be(x[i][j]);

  // Kiirás

      cout<<"A megadott mátrixban "<<h.darab()
          <<" darab különböző szám található!"<<endl;

      char ch;
      cin>>ch;
      return 0;
}
```

```
// Mátrix létrehozása és feltöltése szöveges állományból

void olv(int** &x, int &n, int &m)
{
      ifstream f("input.txt");
      if (f.fail()){
          cerr<<"Az input file nem létezik"<<endl;
          char ch;
          cin>>ch;
          exit(2);
      }

      f>>n>>m;
      x=new int* [n];
      for (int i=0;i<n;i++){
          x[i]=new int [m];
          for (int j=0;j<m;j++){
             f>>x[i][j];
          }
      }
}
```

halmaz.h:

```
#ifndef HALMAZ_H
#define HALMAZ_H

// 0 és 99 közé eső egészeket tartalmazó halmazok típusa
// két művelettel: elem hozzáadása és elemszámlálás

class halmaz{
    bool vekt[100];
    int  db;
  public:
    halmaz();
    void be(int e);
    int darab();
};

#endif
```

halmaz.cpp:

```
#include "halmaz.h"

// Üres halmaz létrehozása

halmaz::halmaz()
{
    for (int i=0;i<100;i++)
        vekt[i]=false;
    db=0;
}

// Elemet tesz a halmazba

void halmaz::be(int e)
{
    if (!vekt[e]){
        vekt[e]=true;
        db++;
    }
}

// Halmaz elemszámának kérdezése

int halmaz::darab()
{
    return db;
}
```