Aplicații pentru scheme logice și pseudocod

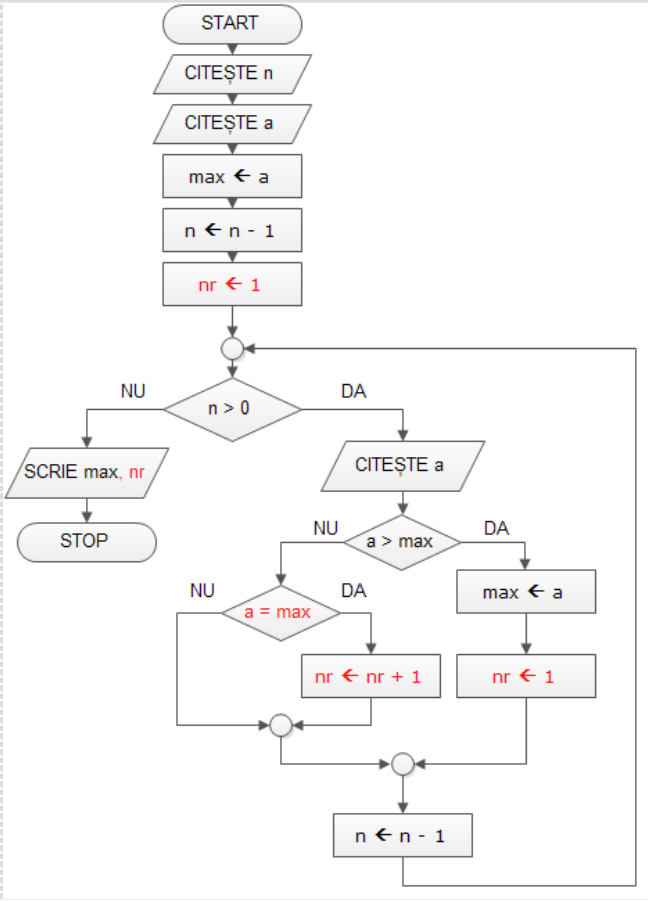
1) Maximul și numărul aparițiilor sale

- Scrieți un program care să rezolve următoarea problemă:
  - Se citesc **n** (număr natural nenul, **n<100**) și apoi **n** numere naturale **nenule** având câte **cel mult 9 cifre** fiecare (numerele nu sunt neapărat distincte);
  - Se cere să se afișeze cel mai mare dintre cele n numere citite și numărul aparițiilor sale.

1) Maximul și numărul aparițiilor sale

- Algoritmul pseudocod de rezolvare a problemei are în pseudocod forma de mai jos, iar schema logică se află în figura alăturată:

```
citește n
citește a
max ← a
n ← n – 1
nr ← 1
cât timp n > 0
{
        citește a
        dacă a > max
        {
                max ← a
                nr ← 1
        }
        altfel
        {
                dacă a= max
                {
                        nr ← nr + 1
                }
        }
        n ← n -1
}
scrie max, nr
```



1) Maximul și numărul aparițiilor sale

- Algoritmul anterior este construit pornind de la cel în care se determină maximul dintre n numere, fără a lua în calcul numărul aparițiilor acestuia;
- Instrucțiunile scrise cu roșu sunt cele necesare pentru extinderea algoritmului.

1) Maximul și numărul aparițiilor sale

- Programul C alăturat rezolvă problema determinării simultane a maximului și a numărului aparițiilor sale;
- Datele se citesc din fișierul text ”date.in”, iar rezultatele sunt scrise în fișierul ”date.out”.

```c
#include <stdio.h>

int main()
{
    int n, a, max, nr;
    FILE *in, *out;
    in = fopen( "date.in", "r" );//voi citi date din fisierul date.in
    out = fopen( "date.out", "w" );//voi scrie rezultatele in date.out
    fscanf( in, "%d%d", &n, &a );//pot citi mai mult variabile
    max = a;
    n--;//echivalent cu n = n - 1
    nr = 1;
    while( n > 0 )
    {
        fscanf( in, "%d", &a );
        if( a > max )
        {
            max = a;
            nr = 1;
        }
        else
        {
            if( a == max )
            {
                nr++;
            }
        }
        n--;
    }
    fprintf( out, "max este %d\nnr. de aparitii este %d\n", max, nr );
    return 0;
}
```

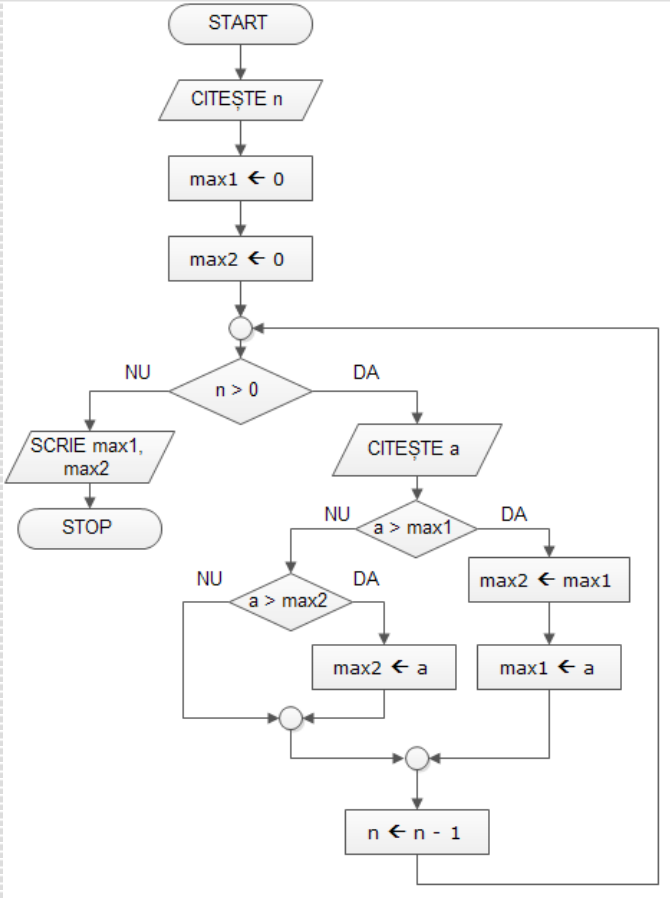
2) Cele mai mari două numere dintr-un șir

- Scrieți un program care să rezolve următoarea problemă:
  - Se citesc **n** (număr natural nenul, **n<100**) și apoi **n** numere naturale **nenule** având câte **cel mult 9 cifre** fiecare (numerele nu sunt neapărat distincte);
  - Se cere să se afișeze cele mai mari două valori (nu neapărat diferite) dintre cele **n** numere citite.

2) Cele mai mari două numere dintr-un șir

- Algoritmul pseudocod de rezolvare a problemei are în pseudocod forma de mai jos, iar schema logică se află în figura alăturată:

```
citește n
max1 ← 0
max2 ← 0
cât timp n > 0
{
        citește a
        dacă a > max1
        {
                max2 ← max1
                max1 ← a
        }
        altfel
        {
                dacă a > max2
                {
                        max2 ← max1
                }
        }
        n ← n -1
}
scrie max1, max2
```



2) Cele mai mari două numere dintr-un șir

- Algoritmul anterior se bazează pe faptul că numerele citite sunt **strict** pozitive;
- Folosind acest lucru, putem inițializa ambele variabile care vor conține rezultatele finale cu 0;
- În general inițializarea trebuie făcută astfel încât să nu afecteze rezultatul final: o sumă sau un contor vor fi inițializate cu 0 (element neutru la adunare), un produs cu 1 (element neutru la înmulțire), un minim cu o valoare mai mare deât toate cele care intră în discuție, pentru ca după prima comparare acesta să ia valoarea numărului cu care a fost comparat;
- Un maxim poate fi inițializat cu o valoare mai mică decât toate cele care sunt luate în discuție;
- Fiecare dintre aceste variabile pot fi inițializate cu primul element al șirului;

2) Cele mai mari două numere dintr-un șir

- Algoritmul folosește variabilele max1, pentru a reține cea mai mare valoare din șir și max2 pentru a doua cea mai mare valoare;
- La citirea fiecărui număr a, acesta se compară mai întâi cu max1;
- În cazul în care a>max1, trebuie modificate atât max1, cât și max2;
- Variabila max2 va prelua conținutul lui max1, iar max1 va deveni a;
- Dacă a este mai mare doar ca max2, atunci valoarea sa va fi reținută în max2.

2) Cele mai mari două numere dintr-un șir

- Programul C alăturat rezolvă problema determinării simultane a celor mai mari 2 valori dintr-un șir de n numere strict pozitive;
- Putem inițializa variabilele max1 și max2 cu 0, pentru că știm că numerele citite sunt naturale nenule;
- Datele se citesc din fișierul text ”date.in”, iar rezultatele sunt scrise în fișierul ”date.out”.

```c
#include <stdio.h>

int main()
{
    int n, a, max1, max2;
    FILE *in, *out;
    in = fopen( "date.in", "r" );//voi citi date din fisierul date.in
    out = fopen( "date.out", "w" );//voi scrie rezultatele in date.out
    fscanf( in, "%d", &n );//nu citesc un prim element al sirului
    max1 = 0;//initializarea se bazeaza pe faptul ca termenii sirului sunt strict pozititvi
    max2 = 0;
    while( n > 0 )
    {
        fscanf( in, "%d", &a );
        if( a > max1 )
        {
            max2 = max1;
            max1 = a;
        }
        else
        {
            if( a > max2 )
            {
                max2 = a;
            }
        }
        n--;
    }
    fprintf( out, "max1 = %d\nmax2 = %d\n", max1, max2 );
    fclose( in );
    fclose( out );
    return 0;
}
```