# INFORMATICĂ

**Suport de lucru pentru anul I de studiu - gimnaziu**



(C) 2007 - Mircea Sabău

# I. INTRODUCERE

**Calculatorul**, căruia datorită complexității sale i se mai spune **sistem de calcul** este alcătuit din dispozitive care realizează:

- **introducerea datelor** – cu tastatură, mouse, scaner şi altele;
- **prelucrarea datelor pe baza unui program** – cu microprocesorul;
- **extragerea (vizualizarea) rezultatelor -** folosind imprimanta, monitorul şi nu numai;
- **memorarea datelor** – cu ajutorul memoriei interne, discului hard, discului flexibil (floppy), compact discului (CD), discului digital versatil (DVD) şi altele.

**Sistemul de calcul** este un ansamblu format din **trei componente**:

- hardware (componente fizice);
- software (componente logice);
- peopleware (personalul necesar pentru programarea, operarea şi întreținerea calculatorului).

*Definiție:* **BIT**
Unitatea de măsură utilizată pentru a măsura cantitatea de informație se numeşte **BIT** (**BI**nary digi**T** – cifră binară). Un bit exprimă o cifră binară (**0** sau **1**) sau o condiție logică (**adevărat** sau **fals**).

*Definiție:* **OCTET** (BYTE)
Un grup de **8 biți** se numeşte **octet** sau **byte** şi poate reprezenta diferite tipuri de informații, cum ar fi o literă din alfabet, o cifră zecimală sau altceva.

Pentru cantități mari de informație se utilizează **multipli octetului**:

- **1 kilobyte (1 KB) = 1024 bytes**
- **1 megabyte (1 MB) = 1024 KB**
- **1 gigabyte (1 GB) = 1024 MB**
- **1 terrabyte (1 TB) = 1024 GB**

*Definiție:* **INFORMATICA**
Informatica este ştiința care se ocupă cu studiul prelucrării automate a datelor în scopul obținerii informațiilor.

Deseori se confundă termenul informație cu cel de dată. Datele sunt nişte cunoştințe care au fost deja folosite, pe când informațiile sunt obținute în urma prelucrării datelor.

## SISTEMUL DE NUMERAȚIE ZECIMAL ŞI BINAR

### Numere zecimale

Metoda noastră pentru scrierea numerelor se bazează pe **puterile numărului 10**. Vom vedea imediat ce înseamnă putere ...

*Observație*: cu semnul * notăm operația de **înmulțire**. De ce? Ca să nu confundăm litera **x** cu semnul de înmulțire.



(C) 2007 - Mircea Sabău

*Definiție:* **PUTERE**
Dacă am două numere, să zicem a şi b, atunci prin "a la puterea b" înțeleg operația notată cu $a^b$ care este egală cu a * a * ... * a, unde pe a îl înmulțesc cu el însuşi de (b-1) ori.
Exemplu: zece la puterea a patra este $10^4$ = 10*10*10*10 = 10000 (adică 10 înmulțit cu el însuşi de trei ori)

Fie numărul 2468. Atunci, 2 reprezintă 2 mii, 4 reprezintă 4 sute, 6 reprezintă 6 zeci, iar 8 reprezintă 8 unități:

2468 = 2*1000 + 4*100 + 6*10 + 8*1

O mie este 10*10*10, ceea ce poate fi scris $10^3$, sau 10 la puterea a treia.
Utilizând această notație, putem scrie relația precedentă astfel:
$2468 = 2*10^3 + 4*10^2 + 6*10^1 + 8*10^0$ – aceasta este **forma notației în baza 10**

*Observații*:
- $10^0 = 1$ (zece la puterea zero este egal cu 1)
- În general, orice **număr diferit de zero** ridicat la **puterea zero** este egal cu 1

Deoarece notația este bazată pe puterile lui 10, o denumim notație **în baza 10** sau **zecimală**.
Oricine poate alege un alt număr ca bază. Pentru a marca faptul că numărul 2468 este scris în baza 10, **îl vom nota 2468 $_{(10)}$**.

*Exercițiu*
Să se scrie următoarele numere zecimale sub forma notației **în baza 10 (zecimală)**.
$1209_{(10)}$ =
$804_{(10)}$ =
$7180_{(10)}$ =

**Numere binare**

Chiar dacă folosiți notații zecimale pentru a scrie un număr întreg, calculatorul îl va stoca (memora) în **valoare binară**, sau **în baza 2**.
În notația binară se folosesc doar două cifre, **0** şi **1**.

De exemplu, 10011011 este un număr binar.
Ca să îl deosebim de numerele reprezentate în alte sisteme de numerație, cum ar fi sistemul în baza zece, **îl vom nota: 10011011 $_{(2)}$**

Numerele binare se bazează pe **puterile lui 2**:

**10011011 $_{(2)}$** $= 1*2^7+0*2^6+0*2^5+1*2^4+1*2^3+0*2^2+1*2^1+1*2^0 =$
$128_{(10)}+0_{(10)}+0_{(10)}+16_{(10)}+8_{(10)}+0_{(10)}+2_{(10)}+1_{(10)}$ = **155 $_{(10)}$**

*Observație*: calculul de mai sus reprezintă modalitatea de a trece un număr scris în baza 2 în numărul corespunzător, scris în baza 10.


(C) 2007 - Mircea Sabău

*Exercițiu*
Să se scrie următoarele numere binare în baza zece.
$101001_{(2)}$ =
$110010_{(2)}$ =
$011001_{(2)}$ =

Notația binară se potrivește memoriei unui calculator, în care fiecare element, care se numește **bit**, poate fi **activ** (egal cu 1) sau **inactiv** (egal cu 0).

Memoria este organizată de obicei în **grupuri de biți**, care se numesc **octeți**, fiecare octet având **8 biți**. Biții într-un octet sunt numerotați după puterile lui 2 corespunzătoare.
Astfel, cel mai din dreapta este bitul 0, următorul este bitul 1 și așa mai departe. Mai jos este prezentat un număr întreg pe doi octeți.

*Observație*: cel mai din **dreapta** bit se numește **bitul cel mai puțin semnificativ** (bitul zero), iar cel mai din stânga bit se numește **bitul cel mai semnificativ** (bitul 15, în exemplul care urmează).

Exemplu de număr întreg reprezentat pe 16 biți (doi octeți):

| număr bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **valoare bit** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **1** | **0** | **0** | **1** | **0** | **0** | **0** | **1** | **0** |

Valoarea numărului binar de mai sus în baza 10 este:
$1*2^{11} + 1*2^{8} + 1*2^{5} + 1*2^{1} = 2048 + 256 + 32 + 2 = 2338$

*Observație*: în suma de mai sus lipsesc termenii corespunzătorii biților cu valoarea zero, deoarece aceștia sunt egali cu zero.
*Atenție:* în calculator un număr întreg poate fi reprezentat pe 32 de biți (4 octeți), iar mai nou, pe 64 de bți, nu pe 16 biți (doi octeți). Există însă și excepții, în care nu lucrăm cu numere întregi mari și ajung doi octeți pentru reprezentarea lor.

**Corespondențe zecimal – binare**

| **Număr zecimal** | **Număr binar** |
|---|---|
| $0_{(10)}$ | $0_{(2)}$ |
| $1_{(10)}$ | $1_{(2)}$ |
| $2_{(10)}$ | $10_{(2)}$ |
| $3_{(10)}$ | $11_{(2)}$ |
| $4_{(10)}$ | $100_{(2)}$ |
| $5_{(10)}$ | $101_{(2)}$ |
| $6_{(10)}$ | $110_{(2)}$ |
| $7_{(10)}$ | $111_{(2)}$ |
| $8_{(10)}$ | $1000_{(2)}$ |

| **Număr zecimal** | **Număr binar** |
|---|---|
| $9_{(10)}$ | $1001_{(2)}$ |
| $10_{(10)}$ | $1010_{(2)}$ |
| $11_{(10)}$ | $1011_{(2)}$ |
| $12_{(10)}$ | $1100_{(2)}$ |
| $13_{(10)}$ | $1101_{(2)}$ |
| $14_{(10)}$ | $1110_{(2)}$ |
| $15_{(10)}$ | $1111_{(2)}$ |
| $16_{(10)}$ | $10000_{(2)}$ |
| $17_{(10)}$ | $10001_{(2)}$ |


(C) 2007 - Mircea Sabău

**Trecerea unui număr din baza 10 în baza 2**

Luând spre exemplu numărul 57 şi folosind **teorema împărţirii cu rest**, obţinem :

```
57 |2 .
56 |28  |2 .
 1 |28  |14   |2 .
    0  14     |7 |2 .
       0       6 |3 |2 .
               1 |2 .|1
                  1
```

**$57_{(10)} = 111001_{(2)}$**

*Observaţie*: Cifrele obţinute pentru scrierea în baza 2 sunt luate începând cu ultima împărţire.

**De reţinut:** Pentru a trece un număr din baza 10 în baza 2 se împarte numărul la 2 şi se reţine restul, apoi câtul obţinut se împarte la 2 şi se reţine restul; se continuă procedeul până se obţine câtul 0. Numărul scris în baza 2 se obţine scriind toate resturile de la ultimul la primul.

Exemple:
$2 = 2_{(10)} = 10_{(2)}$; $62_{(10)} = 111110_{(2)}$; $1995_{(10)} = 11111001011_{(2)}$; $1024_{(10)} = 10000000000_{(2)}$ ;

*Exerciţiu*
Să se scrie următoarele numere zecimale în baza doi.
$39_{(10)}$ =
$47_{(10)}$ =
$64_{(10)}$ =

Datele se păstrează în memoria internă, codificate în sistemul de numeraţie binar. Folosind codul **ASCII** (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange – cod standard american pentru schimbul de informaţii) fiecărui caracter de pe tastatură (literă, cifră, semn special) i se ataşează în mod unic un număr din **mulţimea numerelor întregi {0 ... 255}**.

De exemplu, litera mare **A**, are codul în zecimal 65, litera **B** are codul 66 şi aşa mai departe, iar caracterul **1** are codul 49, caracterul **2** are codul 50, caracterul **3** are codul 51 etc.

Memoria calculatorului este alcătuită din componente fizice care au două stări: fals, codificat prin 0 şi adevărat, codificat prin 1. Aşadar, datele vor fi reprezentate în calculator printr-o succesiune de biţi de lungime fixă.

*De exemplu*,
numărul zecimal întreg **92** va fi reprezentat în calculator pe **4 octeţi**:

| octetul 3 | octetul 2 | octetul 1 | octetul 0 |
|---|---|---|---|
| 00000000 | 00000000 | 00000000 | 01011100 |

iar litera mare **A** (care are codul ASCII 65), va fi reprezentată în calculator pe **1 octet**:

| octetul 0 |
|---|
| 01000001 |



(C) 2007 - Mircea Sabău

**TEMĂ**

**1.** Să se scrie următoarele numere din baza 10 în baza 2:
$2_{(10)}$ =
$4_{(10)}$ =
$8_{(10)}$ =
$16_{(10)}$ =
$31_{(10)}$ =
$120_{(10)}$ =
$256_{(10)}$ =

**2.** Să se scrie următoarele numere din baza 2 în baza 10:
$1100_{(2)}$ =
$111_{(2)}$ =
$1000_{(2)}$ =
$101011_{(2)}$ =

**3.** De câți biți avem nevoie pentru a putea reprezenta în baza 2 numărul $65_{(10)}$?

**4.** Care este cel mai puțin semnificativ bit al numărului $38_{(10)}$ reprezentat în baza 2?

**5.** Se dă numărul zecimal $29_{(10)}$. Ce valoare are bitul 1 în reprezentarea binară a acestui număr?

**6.** Să se scrie în baza 2 codul ASCII al literei mari **B**.

**7.** Să se „descopere” modul în care se poate utiliza aplicația **Calculator** pentru efectuarea operațiilor de conversie a numerelor din baza 10 în baza 2 şi din baza 2 în baza 10. Aplicația Calculator este accesibilă din: Start – All Programs – Accessories – Calculator. Verificați apoi dacă ați rezolvat corect exercițiile 1, 2, 3, 4, 5 şi 6.



(C) 2007 - Mircea Sabău

## II. DE LA ALGORITM LA PROGRAMUL C

Dacă în matematică noțiunea cea mai importantă este numărul, în informatică cea mai importantă noțiune cu care se lucrează este **algoritmul**.

*Definiție*: Prin algoritm vom înțelege o metodă universală de rezolvare a problemelor de un anumit tip. Algoritmul este o succesiune finită de operații cunoscute, care se execută într-o ordine stabilită astfel încât plecând de la un set de date (numite datele problemei sau date de intrare = D.I. ) ce îndeplinesc anumite condiții, să obținem într-un interval de timp finit un set de valori (numite soluțiile problemei sau date de ieşire = D.E.).

### Proprietățile algoritmului

Nu orice descriere pas cu pas a etapelor rezolvării unei probleme reprezintă un algoritm . Pentru a fi algoritm, o metodă de rezolvare a unei probleme trebuie să aibă *trei proprietăți*:

1. generalitate (universalitatea) - un algoritm oferă o metodă generală de rezolvare a unui anumit tip de probleme;
2. finitudine - orice algoritm se încheie într-un timp după un număr finit de paşi;
3. unicitatea - pentru aceleaşi date de intrare se obțin totdeauna aceleaşi date de ieşire.

Pentru a rezolva o problemă cu ajutorul calculatorului trebuie să procedați astfel:

- să stabiliți datele de intrare şi datele de ieşire;
- să elaborați algoritmul şi să îl descrieți folosind **schema logică** sau **limbajul pseudocod**;
- să **transcrieți algoritmul** într-un **limbaj de programare** (construirea programului);
- să executați programul cu mai multe seturi de date de intrare, să corectați erorile şi să stabiliți rezultatele.

### Problemă rezolvată

Activitatea unui lucrător de la Primărie este aceea de a delimita, cu panglică, terenurile agricole, de formă dreptunghiulară, ale țăranilor. Lucrătorul vă cere să îl ajutați să calculeze câți metri de panglică îi sunt necesari pentru a împrejmui un teren, de formă dreptunghiulară, a cărui lungime şi lățime le cunoaşte. Observați că el vrea să scrieți un program care să îi rezolve problema pentru **orice** teren agricol de formă dreptunghiulară, de dimensiuni cunoscute.

1. Datele problemei
Date de intrare: lungimea şi lățimea terenului, pe care le numim **lung** şi **lat**.
Date de ieşire: perimetrul terenului, pe care il numim **perim**.

2. Se elaborează algoritmul, adică se stabileşte o succesiune finită şi bine stabilită de operații (paşi) prin care datele de intrare se transformă în rezultate (date de ieşire).

Algoritmii lucrează cu date care pot fi variabile sau constante. O variabilă corespunde unei zone de memorie, are un nume şi o valoare ataşată. De exemplu, **lung**, **lat** şi **perim** sunt variabilele ce intervin în problema lucrătorului. În cadrul algoritmului, datele sunt supuse următoarelor operații:

- operația de citire – realizează întroducerea de date în memoria calculatorului;



(C) 2007 - Mircea Sabău

- operația de scriere – realizează extragerea rezultatelor din memoria calculatorului şi, de multe ori, afişarea lor;
- operația de atribuire – permite efectuarea de calcule;
- operația de decizie – în funcție de îndeplinirea sau neîndeplinirea unei condiții, algoritmul se ramifică.

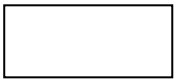
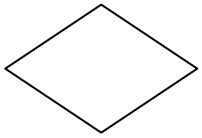
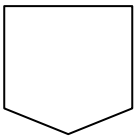
*De reținut:*
Algoritmii efectuează 3 tipuri de operații:
- intrare/ieşire (citire/scriere)
- atribuire
- decizie

*Definiție:* **Algoritm secvențial**
Un algoritm de forma:

*operație 1*
*operație 2*
.....
*operație n*

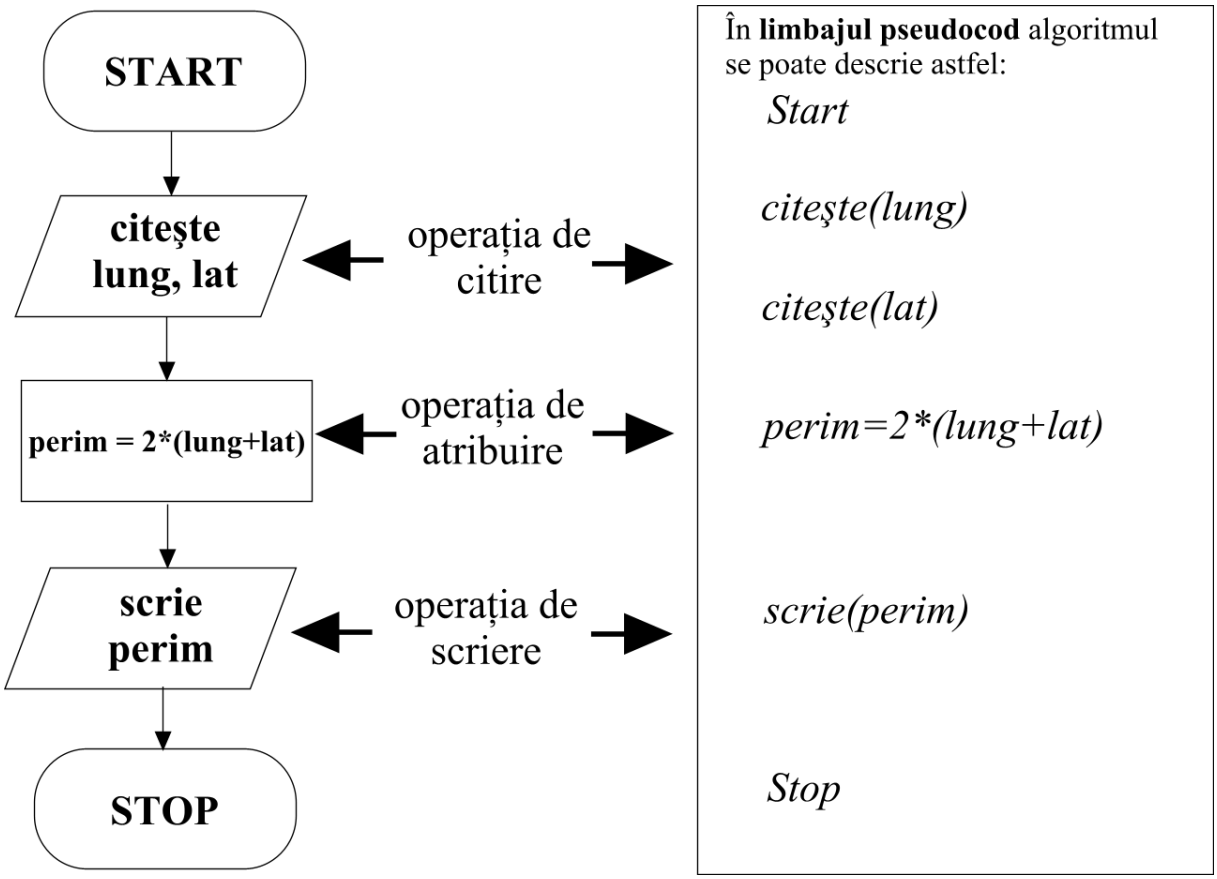
este un algoritm secvențial.

**Schema logică** are simboluri adecvate pentru fiecare operație.

| **Simboluri utilizate în schema logică** | **Semnificație** |
|---|---|
| | **bloc terminal** – indică începutul sau sfârşitul algoritmului |
| | **bloc de citire/scriere** – corespunde operației de citire sau scriere |
|  | **bloc de calcul** – corespunde operației de atribuire |
|  | **bloc de decizie** – descrie operația de decizie |
|  | **conector de pagină** – permite scrierea algoritmului pe mai multe pagini |



(C) 2007 - Mircea Sabău




**Limbajul pseudocod** reprezintă o altă modalitate foarte utilizată de reprezentare a algoritmilor. Spre deosebire de schema logică, care utilizează mijloace grafice, limbajul pseudocod foloseşte o serie de **cuvinte-cheie**. De aceea se spune că este o modalitate textuală (adică prin cuvinte) de reprezentare a algoritmilor. În exemplul anterior cuvintele-cheie utilizate au fost: **start**, **stop**, **citeşte**, **scrie**.

**Scrierea programului** conform algoritmului elaborat anterior. Un program reprezintă o succesiune de comenzi date calculatorului în scopul executării lor. Mai jos este scris programul în C care rezolvă problema lucrătorului de la Primărie. Dacă sunt lucruri pe care nu le înțelegeți, nu vă speriați. Vom reveni asupra acestuia.

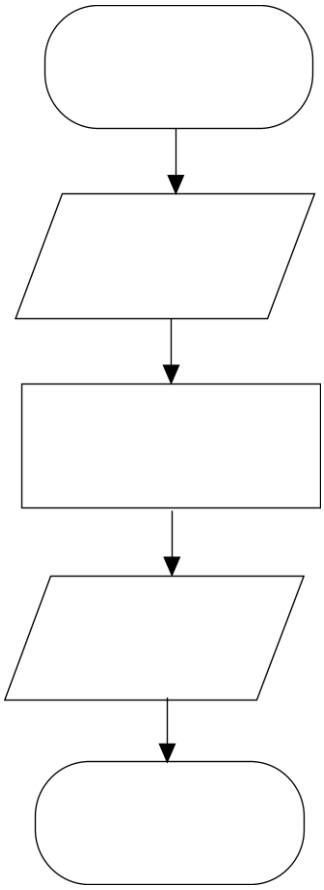
```
#include <stdio.h>

void main(void)
{
  int lung, lat, perim;
  printf("Introduceti lungimea terenului: ");
  scanf("%d", &lung);
  printf("Introduceti latimea terenului: ");
  scanf("%d", &lat);
  perim = 2*(lung+lat);
  printf("Lungimea panglicii este de %d metri.", perim);
}
```



(C) 2007 - Mircea Sabău

*Exercițiu*

Completați blocurile schemei logice şi scrieți algoritmul în limbajul pseudocod pentru următoarea prolemă:

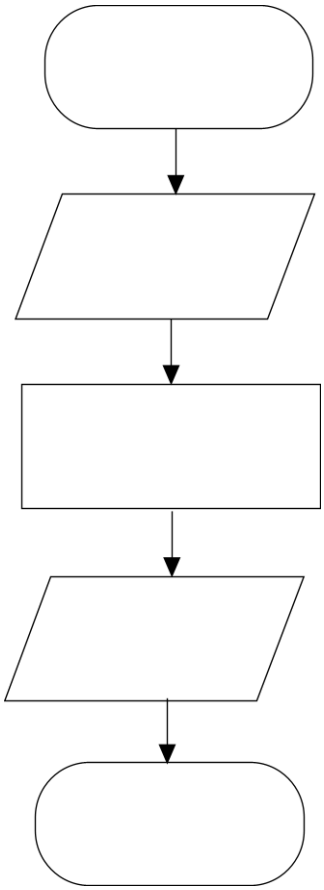
| | |
|---|---|
| Se cunosc lungimile laturilor unui teren agricol de formă triunghiulară (în metri). Determinați câți metri de sârmă sunt necesari pentru împrejmuirea lui.<br><br>**Date de intrare:**<br><br>**Date de ieşire:**<br><br>**Pseudocod:** | **Schema logică**<br>  |



(C) 2007 - Mircea Sabău

**TEMĂ**

**1.** Completați blocurile schemei logice și scrieți algoritmul în limbajul pseudocod pentru următoarea prolemă:

| | |
|---|---|
| Pentru echipa de fotbal FC Steaua calculați "golaverajul" știind că acesta se calculează ca diferența dintre numărul de goluri marcate și numărul golurilor primite.<br><br>**Date de intrare:**<br><br>**Date de ieşire:**<br><br>**Pseudocod:** | **Schema logică**<br>  |



(C) 2007 - Mircea Sabău

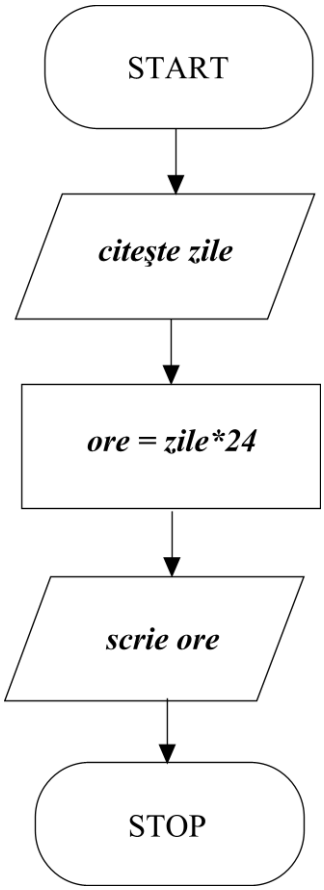
**2.** Reprezentați algoritmul sub formă de schemă logică şi limbaj pseudocod pentru următoarea problemă:
Un alergător parcurge un traseu a cărui lungime o cunoaşte, fiind exprimată în km. Câți metri are traseul alergătorului?

| **Date de intrare:**<br><br>**Date de ieşire:** | |
|---|---|
| **Schema logică** | **Limbaj pseudocod** |
| | |



(C) 2007 - Mircea Sabău

**3.** Concepeți un enunț pentru problema care are ca algoritm schema logică reprezentată mai jos.

| | |
|---|---|
|  | **Enunțul problemei:** |



(C) 2007 - Mircea Sabău

## III. DECLARAREA VARIABILELOR ŞI CONSTANTELOR

Datele cu care lucrează algoritmii pot fi **variabile** sau **constante**. În program, ele se declară astfel:

```
const char a[]=”elev”;
int nota1, nota2;
char adresa[80];
```

La declarare se precizează **numele** variabilelor şi constantelor (în cazul de mai sus: **a**, **nota1**, **nota2, adresa**) şi **tipul** lor (în cazul de mai sus: **char**, **int, char**).

Variabilele şi constantele pot fi:

- întregi (**int**)
- reale (**float** sau **double**)
- caracter (**char**)
- şir de caractere (**char[]**) şi altele ...

*Se observă* că în cazul declarației *const char a[]=”elev”;* în interiorul parantezei drepte nu apare niciun număr, iar în cazul declarației *char adresa[80];* în interior este numărul 80, adică adresa este o variabilă de tip şir de caractere, cu lungimea de 80 de caractere.

*Precizări:* în primul caz, nu este nevoie să spunem care este lungimea şirului de caractere deoarece chiar în momentul declarației, constanta a primeşte şi valoarea sa (elev). Aşadar, se poate cunoaşte, din declarație, că a va avea lungimea de 4 caractere.
În cel de al doilea caz, trebuie să precizăm care va fi lungimea variabilei adresa, deoarece nu se face nicio atribuire în acel moment şi nu se poate deduce numărul caracterelor variabilei adresa. De aceea, în declarație, se specifică această lungime.

În cadrul algoritmului, asupra variabilelor se execută operații de citire şi scriere. Corespunzător, în program se utilizează funcțiile **scanf** pentru citire şi **printf** pentru scriere.

*Exemplu*

```
#include <stdio.h>
void main(void)
{
    char nume[80];
    int varsta;
    printf(“Tasteaza numele tau:”);
    scanf(”%s”, nume);
    printf(“Tasteaza varsta:”);
    scanf(”%d”, &varsta);
    printf(“%s ai varsta de %d ani !!!”, nume, varsta);
}
```

Efectul execuției programului de mai sus va fi:

```
Tasteaza numele tau: Stefan
Tasteaza varsta: 10
Stefan ai varsta de 10 ani !!!
```



(C) 2007 - Mircea Sabău

Să analizăm programul de mai sus:

**Prima linie:**

```
#include <stdio.h>
```

Această declarație există în aproape orice program C. Prin ea programul este informat că, funcțiile de citire **scanf** şi de scriere **printf** sunt definite în fişierul **stdio.h**. Dacă această declarație nu ar exista, programul ne-ar da un mesaj de eroare prin care ne-ar informa că "nu ştie ce e aia scanf şi printf".

**A doua şi a treia linie:**

```
void main(void)
{
```

Aşa începe un program.

**Ultima linie:**

```
}
```

Aşa se termină programul.

**Liniile 4 şi 5:**

```
char nume[80];
int varsta;
```

Aici se declară variabila **nume** care este de tipul şir de caractere şi este lungă de 80 de caractere. Apoi se declară variabila **varsta** care este de tipul întreg.

**Linia 6:**

```
printf("Tasteaza numele tau:");
```

Cu ajutorul funcției printf, se scrie pe ecran mesajul **Tasteaza numele tau:**
Se observă că mesajul este pus între ghilimele.

**Linia 7:**

```
scanf("%s", nume);
```

Cu ajutorul funcției scanf, se citeşte de la tastatura un şir de caractere. Utilizatorul scrie un nume şi apasă apoi tasta ENTER. Numele scris la tastatura va fi preluat şi atribuit vaiabilei **nume**. Se observă aici că, între ghilimele apare **%s**. Acesta indică funcției scanf că va citi de la tastatură un şir de caractere (**s** vine de la **string** – care înseamnă **şir**).

**Linia 8:**

```
printf("Tasteaza varsta:");
```

Cu ajutorul funcției printf, se scrie pe ecran mesajul **Tasteaza varsta:**
Se observă că mesajul este pus între ghilimele.

**Linia 9:**

```
scanf("%d", &varsta);
```

Cu ajutorul funcției scanf, se citeşte de la tastatura un varsta. Utilizatorul scrie un număr şi apasă apoi tasta ENTER. Numărul scris la tastatura va fi preluat şi atribuit vaiabilei **varsta**. Se observă aici că, între ghilimele apare **%d**. Acesta indică funcției scanf că va citi de la tastatură un număr întreg (**d** vine de la **decimal** – care înseamnă număr **zecimal**, adică număr întreg scris în baza 10). Se mai observă aici că, înainte de numele variabilei **varsta**, apare caracterul &. Acesta de regulă nu trebuie să lipsească atunci când citim numere. Excepție este cazul în care citim un şir de caractere (vezi linia 7).



(C) 2007 - Mircea Sabău

**Linia 10:**

```
printf("%s ai varsta de %d ani !!!", nume, varsta);
```

Cu ajutorul funcției printf, scriem pe ecran mesajul dintre ghilimele. Mesajul începe cu **%s**, care va fi înlocuit de conținutul primei variabile (**nume**), apoi urmează **ai varsta de**, **%d** va fi înlocuit de conținutul celei de a doua variabile (**varsta**) şi în final mesajul se încheie cu **!!!**

Despre **ultima linie**, am discutat deja. Aceasta încheie programul.

*Observație:*
Numele de variabile şi de constante se formează numai cu:

- litere
- cifre
- caracterul **_**

şi trebuie să înceapă obligatoriu cu o literă sau cu caracterul _

Spre exemplu, a1, nr, v32, a_2, x, xy sunt nume de variabile şi constante corecte.

*Exercițiu:*
Subliniați numele corecte pentru variabile din următoarea listă:
32_a, a_32, x1, x2, s$, $b, xy, lung camera, lung_camera

*Exercițiu:*
Completați în coloana liberă de tip de date veți folosi pentru a lucra cu:

| | |
|---|---|
| alocația unui elev | |
| prețul unei cărți | |
| numărul de elevi ai unei clase | |
| numele profesorilor | |
| numărul cărților dintr-o bibliotecă | |
| autorii cărților | |

**Citirea de la tastatură – scanf**
Pentru a citi numere (întregi, reale), texte de la tastatură, se foloseşte funcția **scanf**. Pentru a putea folosi **scanf**, trebuie inclus, înainte de începerea programului, fişierul **stdio.h** astfel:

```
#include <stdio.h>
```

**Citirea unui număr întreg**
Pentru a citi un număr întreg de la tastatură, se declară o variabilă de tip număr întreg, după care, cu ajutorul scanf se poate citi de la tastatură numărul şi plasa în acea variabilă:

```
int numar;
scanf("%d", &numar);
```

**%d** – înseamnă că se va citi un număr **întreg**
**&** - înseamnă că numărul scris la tastatură va fi plasat **în** variabila număr.



(C) 2007 - Mircea Sabău

**Citirea unui număr real**
Pentru a citi un număr real de la tastatură, se declară o variabilă de tip număr real, după care, cu ajutorul scanf se poate citi de la tastatură numărul şi plasa în acea variabilă:

```
float numar_real;
scanf("%f", &numar_real);
```

**%f** – înseamnă că se va citi un număr **real (flotant, în simplă precizie)**
**&** - înseamnă că numărul scris la tastatură va fi plasat **în** variabila număr_real.
... sau, dacă doresc să folosesc numere reale în precizie dublă declar variabila:

```
double numar_real;
```

**Citirea unui şir de caractere**
Pentru a citi un şir de caractere de la tastatură, se declară o variabilă de tip şir de caratere, după care, cu ajutorul scanf se poate citi de la tastatură şirul de caractere şi plasa în acea variabilă:

```
char mesaj[60];
scanf("%s", mesaj);
```

**%s** – înseamnă că se va citi un **şir de caractere**
**În cazul citirii unui şir de caractere, nu se foloseşte semnul &**, înainte de variabila mesaj.

**Scrierea pe ecran – printf**
Pentru a scrie pe ecran numere (întregi, reale), texte, se foloseşte funcția **printf**. Pentru a putea folosi **printf**, trebuie inclus, înainte de începerea programului, fişierul **stdio.h** astfel:

```
#include <stdio.h>
```

**Scrierea unui număr întreg**
Pentru a scrie un număr întreg pe ecran, se foloseşte funcția **printf** astfel:

```
printf("%d", numar);
```

**%d** – înseamnă că se va scrie un număr **întreg**

**Scrierea unui număr real**
Pentru a scrie un număr real pe ecran, se foloseşte funcția **printf** astfel:

```
printf("%f", numar_real);
```

**%f** – înseamnă că se va scrie un număr **real**

**Scrierea unui şir de caractere din variabilă**
Pentru a scrie pe ecran un şir de caractere dintr-o variabilă, se foloseşte funcția **printf** astfel:

```
printf("%s", mesaj);
```

**%s** – înseamnă că se va scrie un **şir de caractere**


(C) 2007 - Mircea Sabău

**Scrierea unui şir de caractere direct din program**
Pentru a scrie pe ecran un şir de caractere direct din program, se foloseşte funcția **printf** astfel:

```
printf("Acesta este mesajul pe care vreau sa il scriu !");
```

**Trecerea la rândul următor, când se scrie pe ecran**
Pentru a trece la rândul următor când scriem pe ecran, putem scrie un mesaj special (**n**)care face acest lucru:

```
printf("\n");
```

*Exemplu*:

```
printf("Acesta este mesajul din primul rand\n");
printf("Acesta este mesajul din al doilea rand");
```

Programul va afişa:

**Acesta este mesajul din primul rand**
**Acesta este mesajul din al doilea rand**

*Observație*: vom mai reveni asupra funcțiilor **printf** şi **scanf**.

*Problemă rezolvată*
Se citesc titlul unei cărți şi prețul acesteia. Să se afişeze titlul cărții şi prețul cărții scumpită cu 24 lei folosind limbajul pseudocod şi limbajul C.

| **Date de intrare:**<br>titlu, pret<br><br>**Date de ieşire:**<br>titlu, pret_nou | |
|---|---|
| **Limbaj pseudocod**<br><br>**Start**<br>**citeste(titlu)**<br>**citeste(pret)**<br>**pret_nou = pret + 24**<br>**scrie (titlu)**<br>**scrie(pret_nou)**<br>**Stop** | **Limbaj C**<br><br>#include <stdio.h><br>void main(void)<br>{<br>char titlu[60];<br>int pret;<br>int pret_nou;<br>scanf("%s", titlu);<br>scanf("%d", &pret);<br>printf("%s", titlu);<br>printf(" costa acum ");<br>printf("%d", pret_nou);<br>} |



(C) 2007 - Mircea Sabău

*Exercițiu*:
Precizați ce se afișează în urma executării următoarelor două programe C:

| | |
|---|---|
| #include <stdio.h><br>void main(void){<br>  printf(“Sunt un elev\n”);<br>  printf(“pasionat de informatica\n”);<br>  printf(“de la Gimnaziul George Cosbuc.\n”);<br>} | #include <stdio.h><br>void main(void){<br>  printf(“1 iunie”);<br>  printf(“2007”);<br>  printf(“Tirgu Mures”);<br>} |
| Se afișează: | Se afișează: |

*Exercițiu*:
Programele următoare citesc trei note la informatică ale unui elev și trebuie să le afișeze:

| a) unele după altele | b) unele sub altele |
|---|---|
| Completați programele astfel încât să răspundă cerințelor: | |
| #include <stdio.h><br>void main(void)<br>{<br>  int n1, n2, n3;<br>  printf(“Introduceti notele\n”);<br>  scanf(“%d”, &n1);<br>  scanf(“%d”, &n2);<br>  scanf(“%d”, &n3);<br><br>} | #include <stdio.h><br>void main(void)<br>{<br>  int nota1, nota2, nota3;<br>  printf(“Introduceti notele\n”);<br>  scanf(“%d”, &nota1);<br>  scanf(“%d”, &nota2);<br>  scanf(“%d”, &nota3);<br><br>} |



(C) 2007 - Mircea Sabău

**TEMĂ**

**1.** Se citesc numele şi greutatea (în grame) a doi copii. Să se scrie programul C, care citeşte aceste date de intrare şi apoi afişează numele copiilor şi greutatea lor în kilograme, ca în exemplul de mai jos:
*Exemplu:*
**Programul citeşte**
Adina
35000
Dan
41000
**Programul afişează**
Adina – 35kg
Dan – 41kg


(C) 2007 - Mircea Sabău

## IV. CALCULE CU DATE DE TIP ÎNTREG

Pentru a realiza calcule în cadrul algoritmilor se utilizează operația de **atribuire**. Ea are următoarea formă:

```
variabila = expresie;
```

Expresia din partea dreaptă a semnului egal poate fi formată din:

- constante;
- variabile;
- apeluri de funcții;
- operatori.

De regulă, **variabila şi expresia** trebuie să aibă **acelaşi tip**. În expresiile numerice pot apărea operatori aritmetici: +. -, *, /, %, ++, --. Mai sunt şi alții ...

Operatorul **%** (**modulo**) calculează restul împărțirii întregi a două numere. De exemplu 7%4 va fi egal cu restul împărțirii întregi a lui 7 cu 4, adică **3**.
Explicație: 7 / 4 = 1 rest **3**.

**Operatorul de împărțire**
Împărțirea a două numere întregi se face cu ajutorul operatorului: /
Rezultatul împărțirii a două numere întregi este restul împărțirii.
*Exemplu:*

```
int a, b, c;    /* declaram variabilele a, b si c ca fiind numere intregi */
a=20;           /* lui a ii dam valoarea 20 */
b=3;            /* lui b ii dam valoarea 3 */
c=a / b;        /* c va fi egal cu catul impartirii lui a la b */
printf("%d\n",c);
c=a % b;        /* acum c va fi egal cu restul impartirii lui a la b */
printf("%d",c);
```

Acest cod va afişa:

```
6
2
```

Explicație: 20 / 3 = 6 rest 2
*Observație:* textul cuprins între **/* şi */** este un **comentariu**. Acesta nu face parte din program.

Operatorul ++ este unul de **incrementare** (mărire cu 1). De exemplu, secvența:

```
int i;
i = 7;
printf("%d\n",i);
i++;                    /* incrementare */
printf("%d\n",i);
```

va afişa:

```
7
8
```

La fel, operatorul -- este unul de **decrementare** (micşorare cu 1).



(C) 2007 - Mircea Sabău

*Notă:*

în loc de
**i++;**
poate fi scris şi
**i = i + 1;**

*La fel,*

în loc de
**i--;**
poate fi scris şi
**i = i - 1;**

*Problemă rezolvată*

Să se calculeze ultima cifră a produsului a două numere naturale, mai mici ca 200, numerele fiind citite de la tastatură.

```
#include <stdio.h>
/* Calculeaza ultima cifra a produsului a*b */
void main(void)
{
    int a, b, c, p;
    printf(”Introduceti numarul a: “);
    scanf(”%d”, &a);
    printf(”Introduceti numarul b: “);
    scanf(”%d”, &b);
    p = a * b;
    c = p % 10;
    printf(“Ultima cifra a produsului este: %d”, c);
}
```

Să verificăm cu un exemplu de date de intrare: a = 7, b = 4. Atunci:
p = 7*4 = 28;
c = 28 % 10 = **8.** (28 / 10 = 2 rest **8**).

*Observaţie*: Cele două instrucţiuni:

```
p = a * b;
c = p % 10;
```

le puteam scrie şi într-una singură:

```
c = (a * b) % 10;
```

*Exerciţiu*:

Stabiliţi valorile expresiilor şi completaţi în tabelul de mai jos conform modelului:

| **Expresia** | **Valoarea expresiei** |
|---|---|
| 123+4*5 | 123+20=143 |
| 125%5 | |
| 5*24%5+25/10 | |
| 14+(7*(3+9/4)) | |
| 24/9+2*5%6 | |



(C) 2007 - Mircea Sabău

*Exercițiu*:
Alcătuiți un program care să afișeze valorile expresiilor de la exercițiul anterior. Verificați dacă valorile obținute cu ajutorul calculatorului corespund cu cele scrise de voi în tabel.

*Exercițiu*:
Care sunt valorile variabilelor a şi b după ce s-au executat instrucțiunile de atribuire?

**1.**
a=3;
b=a/3;
b=b+1;

a=.....; b=.....

**2.**
a=100;
a=a+20;
b=a*4;

a=.....; b=.....

**3.**
a=100/25;
b=100%25;
a=a+b;

a=.....; b=.....

**4.**
b=99;
a=b*3;
a=b/3;

a=.....; b=.....



(C) 2007 - Mircea Sabău

**TEMĂ**

**1.** Se cunoaşte salariul pe luna noiembrie al unui muncitor. Cât a avut muncitorul salariul în luna octombrie, ştiind că salariul pe luna noiembrie este cu 500 lei mai mare decât cel din luna octombrie? Scrieți programul C.

**2.** Ce va afişa următorul program?

```
#include <stdio.h>

void main(void)
{
  const char c[]=”Rezultatul este“;
  int a, b;
  a=100;
  b=20;
  a=a+b;
  a=a/10;
  printf(“%s %d”,c,a);
}
```



(C) 2007 - Mircea Sabău

## V. CALCULE CU DATE DE TIP REAL

Deseori, în rezolvarea problemelor, este necesar să utilizăm fracții zecimale finite. De exemplu, trebuie să calculăm:

- **media aritmetică** a două note: **m** = (7+8)/2 = 7,5
- **semiperimetrul** unui triunghi (adică jumătate din perimetrul lui): **s** = (9+9+9)/2 = 13,5

Ca urmare, o dată ce se stabilesc care sunt datele de intrare şi de ieşire, de stabilesc şi variabilele corespunzătoare şi se decide ce tip au. La scrierea programului se cere multă atenție la declararea tipului variabilelor cu care lucrăm. Revenind la exemplul de mai sus, media notelor – **m** şi semiperimeptrul – **s** vor fi toate de tip **REAL** (float sau double).

Ca o primă concluzie, când dorim să lcrăm cu fracții zecimale finite, în informatică se spune că se utilizează **numere reale**.

*Observație:* la scrierea acestor numere în program, caracterul **, (virgulă)** este înlocuit cu caracterul **. (punct)**. Astfel, în program, voi scrie **m = 7.5;** (nu 7,5) sau **s = 13.5;** (nu 13,5).

*De reținut:* când se împart două numere întregi, rezultatul este:

- întreg

Exemplu:

```
int a,b,c;
a=7;
b=4;
c=a/b; /* aici c va primi valoarea înreagă 1 */
```

*De reținut:* când se împart un număr întreg la unul real, rezultatul este:

- real

Exemplu:

```
int a;
float b,c;
a=7;
b=4;
c=a/b; /* aici c va primi valoarea reală 1.75 */
```

**Notația numerelor reale**

Un număr real se poate reprezenta folosind **notația comună** (de exemplu 6.2) sau folosind **notația ştiințifică** (de exemplu, 6.200E+00). Nu insistăm asupra notației ştiințifice. Vom folosi doar notația comună.

*Problemă rezolvată*

Se cunosc cele două note ale Andrei la informatică, obținute în urma unor lucrări de control. Să se calculeze media notelor sale.

Datele de intrare sunt cele două note – cărora le ataşăm variabilele nota1 şi nota2, de tip **intreg (int)**.

Datelor de ieşire – media aritmetică – le corespunde variabila media, care va fi de tip **real (float sau double)** (deoarece prin împărțirea la 2 putem obține un rezultat real).

Algoritmul în limbaj pseudocod este:



(C) 2007 - Mircea Sabău

**Start**
**citeste(nota1)**
**citeste(nota2)**
**media=(nota1+nota2)/2**
**scrie(media)**
**Stop**

Programul se scrie cu uşurinţă:

```
#include <stdio.h>

void main(void)
{
    int nota1, nota2;
    float media;
    printf("Introduceti nota 1: ");
    scanf("%d",&nota1);
    printf("Introduceti nota 2: ");
    scanf("%d",&nota2);
   /* impart suma notelor la 2.0, pt. ca media sa fie un real */
    media=(nota1+nota2)/2.0;
   printf("Media notelor este %f", media);
}
```

*Exerciţiu:*
Se citesc a, b şi c, lungimile laturilor unui triunghi. Să se calculeze **semiperimetrul** triunghiului, scriind programul C.

| **Programul C** |
| --- |
| |

*Exerciţiu:* **Regula celor trei pahare**
Care este efectul execuţiei următorului program ştiind că a, b şi aux sunt numere reale.



(C) 2007 - Mircea Sabău

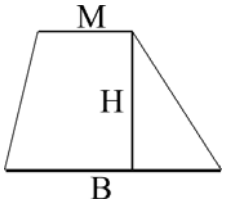
```
/* Regula celor trei pahare */
#include <stdio.h>
void main(void)
{
        float a, b, aux;
        printf("Introduceti a= ");
        scanf("%f",&a);
        printf("Introduceti b= ");
        scanf("%f",&b);
        aux=a;
        a=b;
        b=aux;
        printf("a=%f si b=%f", a,b);
}
```

a) dacă a=10.2 şi b=9.3 programul afişează _____
b) dacă a=100 şi b=200 programul afişează _____
c) dacă a=-60.53 şi b=10 programul afişează _____
d) dacă a=10 şi b=9.27 programul afişează _____
e) dacă a=-60 şi b=1000.1 programul afişează _____

**DE REȚINUT**
Algoritmul implementat de programul de mai sus realizează interschimbarea valorilor a două variabile folosind **regula celor trei pahare**.

**TEMĂ**

**1.** Se cunosc baza mare, baza mică şi înălțimea unui trapez. Să se calculeze aria lui. Scrieți programul C.




(Formula de calcul a ariei este **A = (B+M)*H/2**, unde am notat cu B baza mare, cu M baza mică, cu H înălțimea şi cu A aria trapezului).

Exemplu: pentru B=16, M=8 şi H=3, se va afişa 36.

| **Programul C** |
| --- |
| |



(C) 2007 - Mircea Sabău

## VI. OPERAȚIA DE DECIZIE ŞI INSTRUCȚIUNEA if

De multe ori, în rezolvarea unei probleme apar operații de decizie care conduc la ramificații în cadrul algoritmului.

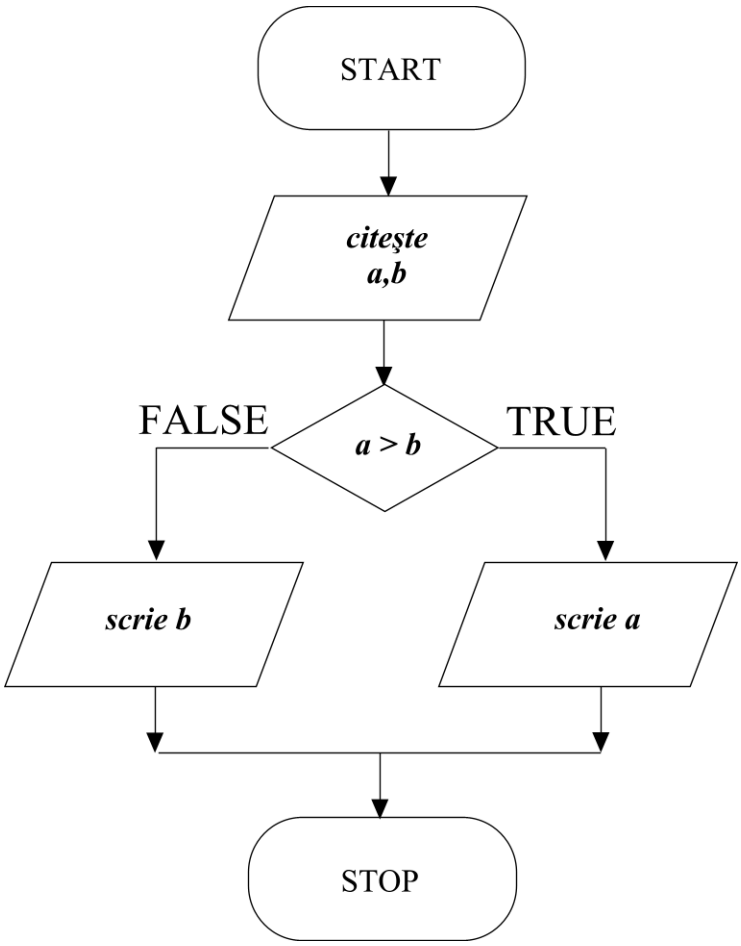
*Definiție:* **Algoritmi cu ramificații**
Algoritmii în care se folosesc operații de decizie se numesc algoritmi cu ramificații.

*Problemă rezolvată*
Se citesc salariile a două persoane – **a** şi **b**. Să se afişeze cel mai mare salariu.

Algorimul prezentat în **limbajul pseudocod** este:

```
Start
    citeste(a,b)
    daca a>b atunci
        scrie(a)
    altfel
        scrie(b)
Stop
```

**Schema logică** a algoritmului este:






(C) 2007 - Mircea Sabău

Decizia se ia în funcție de o **condiție** sau o **expresie logică**. Expresia logică poate lua una din valorile **TRUE** (adevărat) sau **FALSE** (fals). Spre exemplu, 9>4 are valoarea TRUE, iar 10<7 are valoare FALSE.

**Sintaxa instrucțiunii if**

```
if(expresie logică)
        {
                set de instrucțiuni 1;
        }
else
        {
                set de instrucțiuni 2;
        }
```

**Operatorii relaționali sunt:**

- mai mic: <
- mai mare: >
- mai mic sau egal: <=
- mai mare sau egal: >=
- egal: ==
- diferit: !=

Instrucțiunea if transcrie în limbajul C operația de decizie, aşa cum rezultă din exemplul următor:

```
#include <stdio.h>

void main(void)
{
 long int a,b; /* un intreg lung, adica pana la 2147483647 */
/* daca era int, valoarea maxima era doar 65535 */
 printf("Introduceti primul salariu: ");
 scanf("%ld",&a);
 printf("Introduceti al doilea salariu: ");
 scanf("%ld",&b);
 if(a>b)
        {
         printf("Salariul maxim este %ld",a);
        }
 else
        {
         printf("Salariul maxim este %ld",b);
        }
}
```

**Operatori logici**

În expresiile logice pot apărea şi operatorii logici **&&** (operatorul ŞI), **||** (operatorul SAU) sau **!** (operatorul de negare, NU).



(C) 2007 - Mircea Sabău

*Exemple:*
if(x<9 && x>7) – se traduce prin: dacă x este mai mic decat 1 **şi** x este mai mare decât 7
if(x<5 || y>8) – se traduce prin: dacă x este mai mic decat 5 **sau** y este mai mare decât 8
if(!(x<13)) – se traduce prin: dacă x **nu** este mai mic decât 13

**Tabele de adevăr ale operatorilor logici ŞI, SAU, NU**

| **operatorul ŞI** | | |
|---|---|---|
| **a** | **b** | **&&** |
| FALSE | FALSE | FALSE |
| FALSE | TRUE | FALSE |
| TRUE | FALSE | FALSE |
| **TRUE** | **TRUE** | **TRUE** |

| **operatorul SAU** | | |
|---|---|---|
| **a** | **b** | **\|\|** |
| **FALSE** | **FALSE** | **FALSE** |
| FALSE | TRUE | TRUE |
| TRUE | FALSE | TRUE |
| TRUE | TRUE | TRUE |

| **operatorul NU** | |
|---|---|
| **a** | **!** |
| FALSE | TRUE |
| TRUE | FALSE |

*Exercițiu:*
Completați următoarele enunțuri:
1. O expresie logică poate lua valoarea _____ sau _____.
2. În expresiile logice pot apărea şi operatorii logici _____, _____ şi ____.
3. În expresiile logice se pot utiliza şi operatorii relaționali <, _____.

*Exercițiu:*
Se citesc două numere naturale a şi b, mai mici decât 65535. Dacă a este mai mic decât b, se interschimbă valorile celor două variabile. Să se calculeze câtul şi restul împărțirii celor două numere.
Exemplu: pentru a=5000 şi b=5001 se vor afişa câtul 1 şi restul 1.

| **Programul C** |
|---|
| |



(C) 2007 - Mircea Sabău

**TEMĂ**

**1.** Andrei este elev în clasa a V-a şi tatăl său i-a promis o excursie la munte dacă la teza la matematică ia cel puțin nota 6, iar la teza la română ia nota 10. Se citesc cele două note obținute de Andrei la teze şi se afişează dacă băiatul pleacă sau nu în excursie.

| **Programul C** |
| --- |
| |



(C) 2007 - Mircea Sabău

## VII. PROBLEME DE DIVIZIBILITATE

**Problemă rezolvată**

Ionel a participat la un concurs de cultură generală şi are de primit un premiu în bani. Acest premiu are o valoare cunoscută x (număr întreg mai mic decât 65535). Aflați dacă suma poate fi plătită în bancnote de 50 lei.

Ca **date de intrare** se consideră **x** – de tipul **int**, deoarece suma este mai mică decât 65535).

Rezultatul sau **datele de ieşire** îl reprezintă unul din mesajele:

- suma poate fi platita in bancnote de 50 lei
- suma nu poate fi platita in bancnote de 50 lei

Completați următorul tabel:

| **x** | **Rezultat** |
|---|---|
| 5000 | 100 bancnote de 50 lei |
| 1500 | ..... bancnote de 50 lei |
| 2250 | ..... bancnote de 50 lei |

Algoritmul va determina dacă numărul **x** este divizibil (se împarte exact) cu 50:

**Start**
  **citeşte(x)**
  **dacă x%50 == 0 atunci**
    **scrie("suma poate fi platita in bancnote de 50 lei")**
  **altfel**
    **scrie("suma nu poate fi platita in bancnote de 50 lei")**
**Stop**

**Verificarea parității unui număr natural x**

Se face folosind condiția:

```
if(x%2 == 0)
{
    /* numarul x este par */
}
```

**Verificarea imparității unui număr natural x**

Se face folosind condiția:

```
if(x%2 == 1)
{
    /* numarul x este impar */
}
```

**Verificarea apartenenței unui număr natural x la un interval [a, b]**

Se face folosind condiția:

```
if(x>=a && x<=b)
{
    /* numarul x apartine intervalului [a, b] */
}
```



(C) 2007 - Mircea Sabău

*Exercițiu:*
Dacă **a** reprezintă o variabilă întreagă, ce expresie se utilizează pentru a testa dacă **a** este divizibil cu 10?
a) if(a/10 == 0)  b) if(a%10 == 10)  c) if(a%10 == 0)  d) if(a%10 == 1)

*Exercițiu:*
Dacă **a** reprezintă o variabilă întreagă, ce expresie se utilizează pentru a testa dacă **a** este divizibil cu 3?
a) if(a/3)  b) if(a%3 == 2)  c) if(a/3 == 0)  d) if(a%3 == 0)

**Aflarea primei şi ultimei cifre a unui număr**
Să considerăm un număr natural **n** de trei cifre. Să se afle prima cifră şi ultima cifră a acestui număr.
Exemplu: dacă **n** = 153, atunci prima cifră este **p** = 1, iar ultima cifră este **u** = 3.
Mod de calcul:

```
p = n/100;
u = n % 10;
```

**Generalizare**
Dacă avem un număr natural **n** de **k** cifre, atunci prima cifră **p** şi ultima cifră **u** se calculează astfel:

$$p = n/10^{(k-1)}$$
$$u = n\%10$$

*Observație:* $10^{(k-1)}$ înseamnă 10 la puterea (k-1) adică 10 înmulțit cu el însuşi de (k-2) ori.
*Observație:* ultima cifră se calculează fără să conteze câte cifre **k** are numărul **n**.

*Exercițiu:*
Scrieți un program care calculează **cifra zecilor** a unui număr natural **n**.

| **Programul C** |
| --- |
| |


(C) 2007 - Mircea Sabău

**TEMĂ**

**1.** Un elev este declarat respins la examen dacă una din cele două note (**n1**, respectiv **n2**) obținute este sub 5 sau dacă media lor aritmetică este sub 5. Care dintre următoarele expresii verifică aceste condiții?

a) if(n1 || n2<5 && (n1+n2)/2)
b) if(n1 <5 || n2<5 || (n1+n2)/2)
c) if((n1 <5) || (n2<5) || ((n1+n2)/2<5)
d) if(n1 <5 || n2<5 || n1+n2/2<5)

**2.** De la matematică se cunoaşte că dacă ultimele două cifre ale unui număr formează un număr divizibil cu 4, atunci numărul respectiv este divizibil cu 4. Să se scrie programul care citeşte nu număr natural **n** şi care verifică dacă este divizibil cu 4 folosind criteriul de mai sus. Programul va citi numărul şi afişa unul din mesajele de mai jos:

- este divizibil cu 4
- nu este divizibil cu 4

| **Programul C** |
| --- |
| |



(C) 2007 - Mircea Sabău

## VIII. INSTRUCȚIUNEA switch

Există situații în rezolvarea unei probleme când sunt mai multe căi posibile de urmat, caz în care putem folosi instrucțiunea **switch**.

**Sintaxa** instrucțiunii **switch** este următoarea:

```
switch (var)
{
        case c1: set_instructiuni;
                 break;
        case c2: set_instructiuni;
                 break;
        ....................................
        case cn: set_instructiuni;
                 break;
        default:  set_instructiuni;
}
```

*Observații:*
1. **var** este o variabilă, iar **c1**, **c2**, ..., **cn** sunt constante (valori cunoscute)
2. **break** determină ca execuția programului să se facă începând cu prima instrucțiune de după terminarea blocului switch **}**
3. **default** face referire la oricare altă valoare **ck** nemenționată printre valorile **c1**, **c2**, ..., **cn**.

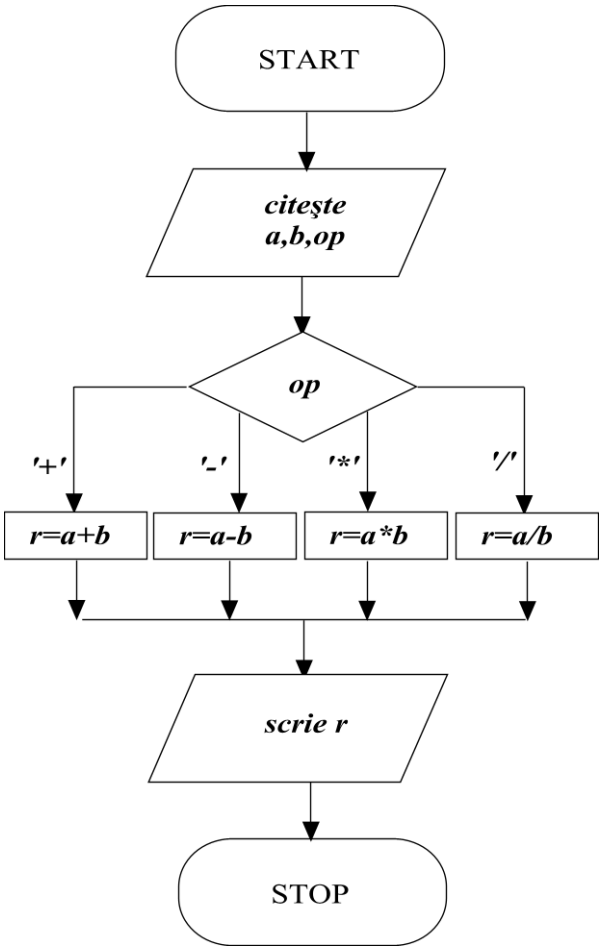
**Problemă rezolvată – un calculator simplu**
Se citesc două valori reale **a** și **b** și un caracter (un semn grafic ce reprezintă un simbol de operație), **op** din mulțimea {+, -, *, /}. Să se afișeze pe ecran rezultatul operației **a op b**.

Există 4 căi de urmat în funcție de valoarea operatorului **op**. Vor rezulta 4 ramuri pe care se vor realiza 4 operații diferite de atribuire.



(C) 2007 - Mircea Sabău

**Schema logică**




**Programul C**

```
#include <stdio.h>

void main(void)
{
 float a, b, r; /* variabile reale */
 char c; /* tip caracter */
 printf("a=");
 scanf("%f",&a);
 printf("b=");
 scanf("%f",&b);
 printf("op=");
 scanf("%c",&c);
 switch(c)
 {
        case '+': r=a+b;
                  break;
        case '-':  r=a-b;
                  break;
        case '*': r=a*b;
                  break;
        case '/':  r=a/b;
                  break;
 }
 printf("Rezultatul este: %f",r);
}
```

*Problemă:*
Se introduce de la tastatură o cifră zecimală şi se cere să se obţină denumirea acesteia. De exemplu, dacă se introduce de la tastatură **5**, programul va afişa **cinci**. Scrieţi programul C.

**Programul C**



(C) 2007 - Mircea Sabău

**TEMĂ:**

1. Se dă următorul program. Încercați să descoperiți ce anume realizează acesta.

**Program C necunoscut**

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>

void main(void)
{
 int c;
 int gata;
 gata=0;
 while(gata==0)
 {
  c=getch();
  switch(c)
  {
        case '1':  sound(262); delay(200); nosound();
                   break;
        case '2':  sound(294); delay(200); nosound();
                   break;
        case '3':  sound(330); delay(200); nosound();
                   break;
        case '4':  sound(349); delay(200); nosound();
                   break;
        case '5':  sound(392); delay(200); nosound();
                   break;
        case '6':  sound(440); delay(200); nosound();
                   break;
        case '7':  sound(494); delay(200); nosound();
                   break;
        case '8':  sound(523); delay(200); nosound();
                   break;
        case '0':   gata=1;
  }
 }
}
```

**Descrieți cu cuvintele voastre ce anume face programul:**



(C) 2007 - Mircea Sabău

**2.** Se citeşte un număr **n**, corespunzător unei luni calendaristice (de exemplu, pentru luna ianuarie, **n** este 1, pentru februarie este 2, pentru martie 3 etc.). Să se determine trimestrul calendaristic corespunzător. Se va verifica corectitudinea datei de intrare.
Exemplu: dacă se citeşte de la tastatură numărul **7**, programul va afişa **trimestrul trei**.

| **Programul C** |
| --- |
| |



(C) 2007 - Mircea Sabău

# IX. ALGORITMI CICLICI

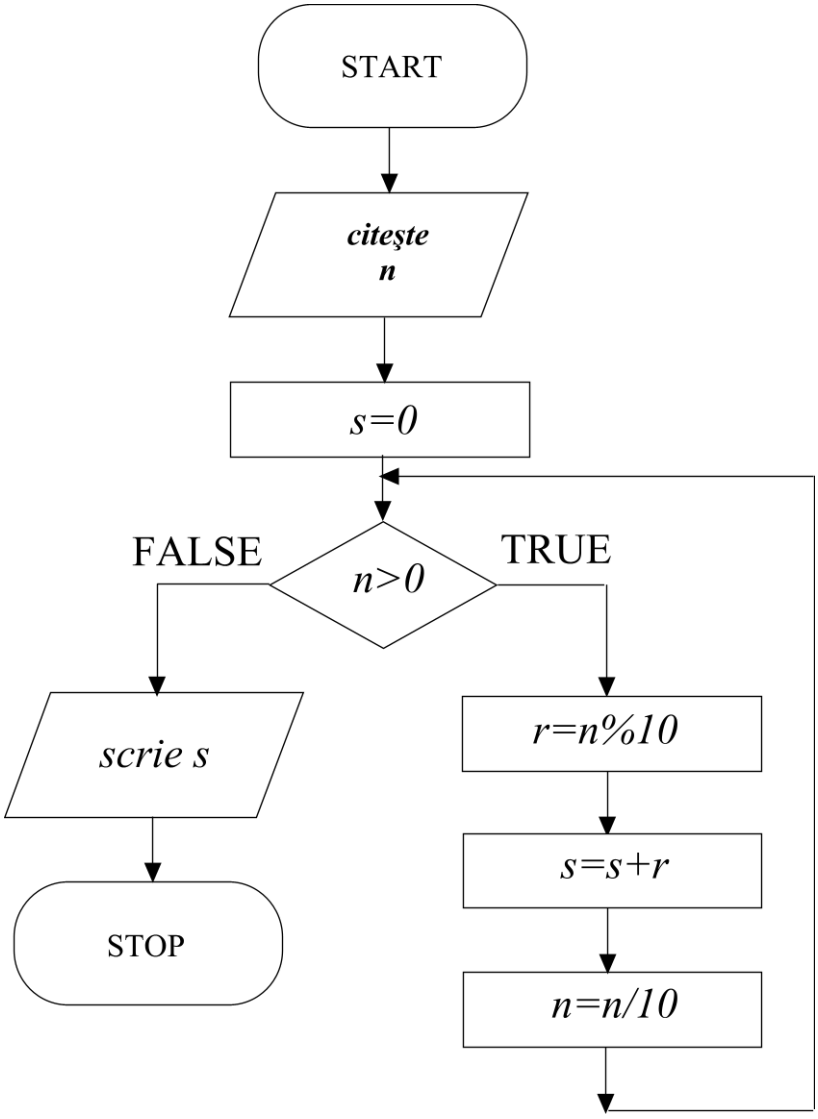
## Ciclul cu un număr necunoscut de paşi şi instrucțiunea while

**Problemă rezolvată**
Se citeşte un număr natural **n**. Să se afle suma cifrelor acestui număr.
**Exemplu:** pentru **n** = 579, rezultatul este 21 (5 + 7 + 9).
Vom efectua în mod repetat următoarele prelucrări:

- se extrage ultima cifră - restul împărțirii numărului la 10
- se face adunarea (restul + suma (inițial suma este zero))
- se obține un nou număr – se face o împărțire la 10

Repetiția acestor operații poartă numele de ciclu sau iterație. Orice ciclu are un număr finit de pași, altfel spus trebuie să se încheie. În exemplul prezentat, repetiția se încheie când numărul obținut prin împărțire la 10 devine 0.

Reprezentarea algoritmului în **schema logică**:






(C) 2007 - Mircea Sabău

*Observații:*

- Ciclul (iterația) asigură executarea în mod repetat a unor operații.
- Ciclul este controlat de una sau mai multe expresii logice.
- Ciclurile pot fi: cu număr necunoscut de paşi şi cu număr cunoscut de paşi.

*Definiție:* **ALGORITMI CICLICI**
Algoritmii care conțin cicluri se numesc **algoritmi ciclici**.

**Programul C**

```
#include <stdio.h>

void main(void)
{
 int n, s, r;
 printf("Scrieti un numar n= ");
 scanf("%d",&n);
 s=0;
 while(n>0)
 {
  r=n%10;
  s=s+r;
  n=n/10;
 }
 printf("Suma cifrelor este: %d",s);
}
```

*De reținut:*
1. Pentru a se calcula o sumă se procedează astfel:

- se inițializează o variabilă **s** cu 0 (în afara ciclului, de obicei înaintea lui)
- se aplică în mod repetat formula de calcul
  **s = s (anterior) + termenul curent**

**Instrucțiunea while** codifică în limbajul C **ciclul cu număr necunoscut de paşi**.
Are **sintaxa**:

```
while (expresie_logica)
{
  instructiune_1;
  instructiune_2;
  ......................
  instructiune_n;
}
```



(C) 2007 - Mircea Sabău

*Exerciţiu:*
Să se verifice dacă un număr natural **n** citit de la tastatură este divizibiul cu 3, conform criteriului de divizibilitate cu 3 (dacă suma cifrelor lui **n** este multiplu de 3, atunci **n** este divizibil cu 3).
Exemplu: pentru **n** = 333999, rezultatul va fi **este divizibil cu 3**, iar pentru **n** = 46, rezultatul va fi **nu este divizibil cu 3**.

| **Programul C** |
| --- |
| |



(C) 2007 - Mircea Sabău

**TEMĂ**

**1.** Se citeşte un număr natural **n**. Să se calculeze produsul cifrelor sale.
Indicație: o variabilă p se inițializează cu 1; se aplică în mod repetat formula de calcul
**p = p (anterior) * factorul curent**

| **Programul C** |
| --- |
| |



(C) 2007 - Mircea Sabău

**2.** Se citeşte un număr natural **n**. Să se calculeze produsul cifrelor sale impare. Dacă nu are asemenea cifre în componența sa, atunci se afişează **nu are cifre impare**.
Exemplu: pentru **n** = 2347, rezultatul va fi **21**, iar pentru **n** = 224466, rezultatul va fi **nu are cifre impare**.

| **Programul C** |
| --- |
| |



(C) 2007 - Mircea Sabău

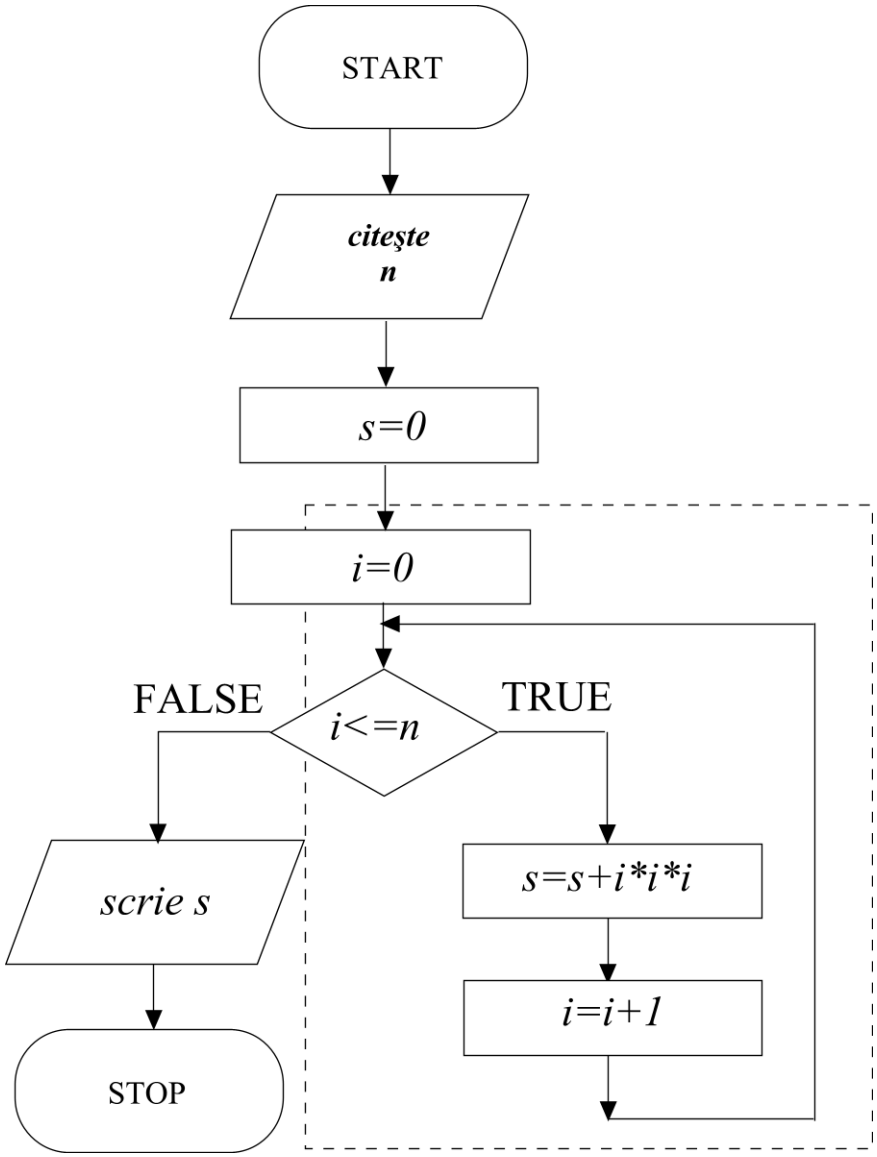
# X. ALGORITMI CICLICI

## Ciclul cu contor şi instrucțiunea for

Am văzut până acum că ciclurile se clasifică în cicluri cu un număr necunoscut de paşi şi **cicluri cu un număr cunoscut de paşi**. Pentru a le implementa pe cele din prima categorie în limbajul C se utilizează instrucțiunea while, iar pentru cealaltă categorie se foloseşte instrucțiunea **for**.

**Problemă rezolvată**
Să se calculeze suma **s** = $1^3 + 2^3 + ... + n^3$ pentru **n** natural citit de la tastatură (1<=**n**<=50).
Amintim că $1^3$ = 1*1*1 = 1, $2^3$ = 2*2*2 = 8, $3^3$=3*3*3 = 27 etc.
Precum se ştie că se procedează la calcularea unei sume, o variabilă **s** se inițializează cu 0. În continuare se repetă de **n** ori operația de atribuire **s = s + i*i*i**, unde **i** se numeşte variabila de ciclu şi ia pe rând valorile 1, 2, 3, ..., până la **n**.

Algoritmul se va reprezenta folosind **schema logică** astfel:






(C) 2007 - Mircea Sabău

*De reținut:* **CICLUL CU CONTOR sau cu număr cunoscut de paşi**
Ciclul cu contor sau cu număr cunoscut de paşi asigură repetiția uneia sau mai multor operații de un număr cunoscut de ori. Contorul ține evidența numărului de repetiții (paşi).

**Programul C** care implementează algoritmul problemei rezolvate şi care utilizează instrucțiunea **for** este:

```
#include <stdio.h>

void main(void)
{
 int n, i;
 long int s; /* intreg lung, deoarece suma poate fi mare */
 printf("Scrieti numarul n= ");
 scanf("%d",&n);
 s=0;
 for(i=1; i<=n; i++)
 {
  s=s+i*i*i;
 }
 printf("Suma este: %ld", s);
}
```

**Sintaxa instruncțiunii for** este următoarea:

```
for(v=val_i; v<=val_f; v++)
{
        instructiune;
        instructiune;
        ....................
        instructiune;
}
```

unde:

- **v** este variabila de ciclu sau contor
- **val_i** este valoarea inițială a contorului
- **val_f** este valoarea finală a contorului
- **v++** corespunde atribuirii **i=i+1** din schema logică a problemei rezolvate (valoarea contorului creşte cu o unitate)

*Exercițiu:*
Se cunosc înălțimile celor **n** băieți dintr-o clasă (1<=**n**<=30). Calculați şi afişați cea mai mare înălțime, ştiind că înălțimile sunt numere întregi, exprimate în centimetri.
**Date de intrare**: **n** (numărul de băieți) şi **h** (păstrează pe rând înălțimea fiecărui băiat din cei **n** băieți)
**Date de ieşire**: **hmax** (înălțimea maximă)
*Indicație:* variabila **hmax**, în care se va obține rezultatul se inițializează cu valoarea 0, o valoare pe care înălțimile băieților nu o pot lua niciodată.



(C) 2007 - Mircea Sabău

| **Programul C** |
| --- |
| |

**TEMĂ**

**1.** Un lift parcurge distanța dintre două etaje **a** şi **b** (**a** şi **b** se citesc de la tastatură). Să se afişeze toate etajele parcurse, în ordinea atingerii lor.
**Exemple**: pentru **a**=4 şi **b**=7 programul va scrie **4 5 6 7**; pentru **a**=10 și **b**= 8 programul va scrie **10 9 8**.

| **Programul C** |
| --- |
| |



(C) 2007 - Mircea Sabău

**2. Ce afişează următorul program?**

```
#include <stdio.h>

void main(void)
{
 int i, j, p;
 for(i=7; i<=8; i++)
 {
  for(j=1; j<=9; j++)
  {
        p=i*j;
        printf("%d x %d = %d\n", i, j, p);
  }
  printf("\n");
 }
}
```

| **Rezultatul afişat** |
| --- |
| |



(C) 2007 - Mircea Sabău

**CUPRINS**

I. Introducere în informatică. Sistemul de numerație zecimal şi binar.
..... pag. 2

II. De la algoritm la programul C. Implementarea algoritmilor în limbajul C. Algoritmi secvențiali.
..... pag. 7

III. Implementarea algoritmilor în limbajul C. Algoritmi secvențiali. Declararea variabilelor şi constantelor.
..... pag. 14

IV. Implementarea algoritmilor în limbajul C. Algoritmi secvențiali. Calcule cu date de tip întreg.
..... pag. 21

V. Implementarea algoritmilor în limbajul C. Algoritmi secvențiali. Calcule cu date de tip real.
..... pag. 25

VI. Implementarea algoritmilor în limbajul C. Algoritmi cu ramificații. Operația de decizie şi instrucțiunea if.
..... pag. 28

VII. Implementarea algoritmilor în limbajul C. Algoritmi cu ramificații. Probleme de divizibilitate.
..... pag. 32

VIII. Implementarea algoritmilor în limbajul C. Algoritmi cu ramificații. Instrucțiunea switch.
..... pag. 35

IX. Implementarea algoritmilor în limbajul C. Algoritmi ciclici. Ciclul cu număr necunoscut de paşi şi instrucțiunea while.
..... pag. 39

X. Implementarea algoritmilor în limbajul C. Algoritmi ciclici. Ciclul cu contor şi instrucțiunea for.
..... pag. 44



(C) 2007 - Mircea Sabău