

PRINCIPIILE PROGRAMARII STRUCTURATE

Orice algoritm are o structură bine stabilită. Astfel transpunerea lui într-un limbaj de programare devine clară și ușor de controlat.

Principalele tipuri de structuri sunt:

- Structura secvențială
- Structura decizională
- Structura repetitivă

1. **Structura secvențială** – în acest caz instrucțiunile se derulează una după cealaltă
2. **Structura decizională** – instrucțiunile se derulează după un criteriu de selecție. Se va selecta execuția unei instrucțiuni în funcție de o condiție dată
3. **Structura repetitivă** – instrucțiunile din cadrul structurii se repetă în funcție de un anumit criteriu. Apare aici un test care stabilește numărul de repetiții. Acest test poate fi la începutul structurii (**structură repetitivă cu test inițial**) sau la sfârșit (**structură repetitivă cu test final**).

Înainte de a fi implementat într-un anumit limbaj de programare un algoritm poate fi descris folosind **schemele logice** sau folosind un limbaj de tip **pseudocod** (vom folosi descrierea algoritmilor folosind pseudocodul).

REPREZENTAREA ALGORITMILOR ÎN PSEUDOCOD

Pseudocodul folosește anumite convenții sau codificări pentru descrierea algoritmului.

Vom stabili următoarele convenții de scriere a algoritmilor în pseudocod:

a. Datele (constantele și variabilele) se vor descrie astfel:

- constantele numerice – așa cum se utilizează în matematică cu observația că virgula zecimală va fi înlocuită cu punctul
- constantele de tip caracter se vor scrie între apostrofuri
- constantele logice vor fi *adevarat* și *fals*
- variabilele – se vor specifica numele și tipul (*natural, intreg, real, caracter, sir de caractere, logic*) despărțite de ":" astfel: *nume:tip*;

b. Operațiile numerice, logice, între șiruri de caracter și între caractere sunt cele prezentate în capitolul anterior

c. Instrucțiunile vor avea cuvinte rezervate:

citește, scrie, ← (pentru atribuire), dacă, atunci, altfel, cât timp, execută, până când, pentru, reia, stop.

A. ALGORITMI CU STRUCTURĂ SECVENȚIALĂ (LINIARĂ)

Sintaxa unui astfel de algoritm este:

```
DECLARARE VARIBILE  
INSTRUCȚIUNE1  
INSTRUCȚIUNE1  
...  
INSTRUCȚIUNE1  
STOP
```

În acest tip de structură instrucțiunile se execută una după alta până la sfârșitul algoritmului.

Vom porni de la următorul exemplu:

Fiind date două numere întregi notate a și b . Să se afișeze suma lor.

Algoritmul în pseudocod este:

```
 $a, b, \text{suma}$ : întreg  
citește  $a$   
citește  $b$   
 $\text{suma} \leftarrow a + b$   
scrie  $\text{suma}$   
stop
```

Exemplul 2: *Se citesc valorile a două variabile a și b naturale. Să se interschimbe conținuturile lor.*

Algoritmul în pseudocod este:

```
 $a, b, \text{suma}$ : natural  
citește  $a$   
citește  $b$   
 $\text{aux} \leftarrow a$   
 $a \leftarrow b$   
 $b \leftarrow \text{aux}$   
scrie  $a$   
scrie  $b$   
stop
```

Observație: Pentru a înțelege acest algoritm întâlnit destul de des care face interschimbarea conținutului a două variabile ne pute imagina două pahare, unul cu apă (a) iar celalalt cu suc(b). Pentru a schimba conținutul este nevoie de un al treilea pahar (aux).

2. ALGORITMI CU STRUCTURĂ DECIZIONALĂ

Acest tip de algoritm poate lua o decizie în funcție de o condiție dată. Decizia poate fi **asupra unui singur caz** sau **între două cazuri posibile**.

a. Algoritmi cu decizie asupra unui singur caz

În acest caz dacă este îndeplinită condiția se execută o instrucțiune iar dacă nu este îndeplinită nu se acționează deloc.

Sintaxa unui astfel de algoritm este:

*DACĂ expresie_logică ATUNCI
EXECUTĂ instrucțiune*

Pașii algoritmului:

- se evaluează expresia logică.
- dacă aceasta este adevărată atunci se execută instrucțiunea
- dacă aceasta nu este adevărată atunci nu se execută nimic

Exemplu:

Se citește valoarea unei variabile întregi a. Să se afișeze mesajul "Numarul este pozitiv" dacă valoarea acestei variabile este mai mare decât 0.

Algoritmul este:

*a: întreg
citește a
dacă $a \geq 0$ atunci
 scrie 'Numarul este pozitiv'
stop*

b. Algoritmi cu decizie între două cazuri

În acest caz dacă este îndeplinită condiția se execută o instrucțiune iar dacă nu este îndeplinită se execută o altă instrucțiune.

Sintaxa unui astfel de algoritm este:

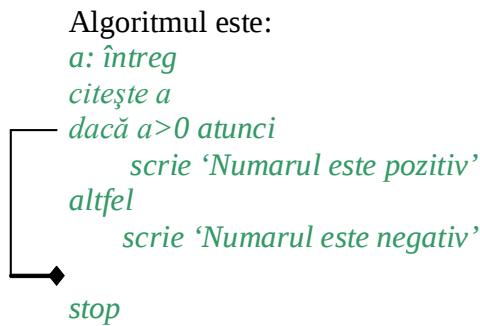
*DACĂ expresie_logică ATUNCI
 instrucțiune1
ALTFEL
 instrucțiune2*

Pașii algoritmului:

- se evaluează expresia logică.
- dacă aceasta este adevărată atunci se execută instrucțiunea1
- dacă aceasta nu este adevărată atunci se execută instrucțiunea2

Exemplu:

Se citește valoarea unei variabile întregi a. Să se afișeze mesajul "Numărul este pozitiv" dacă valoarea acestei variabile este mai mare sau egal cu 0 sau "Numărul este negativ".



4. ALGORITMI CU STRUCTURĂ REPETITIVĂ

Acest tip de structură execută repetarea unei sau mai multor instrucțiuni în funcție de o condiție dată. În funcție de condiția de repetare, structurile repetitive se împart în:

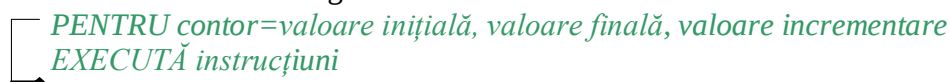
- structuri cu număr cunoscut de repetiții
- structuri cu număr necunoscut de repetiții

A. Structuri cu număr cunoscut de repetiții

O astfel de structură folosește o variabilă de tip **contor** care va fi inițializată cu o valoare de început și va fi incrementată (adunată) sau decrementată (scăzută) până va ajunge la o valoare de sfârșit. Pentru a se stabili dacă s-a ajuns la valoarea de sfârșit se va face un test de margine.

Dacă valoarea de incrementare este 1 atunci aceasta nu se mai scrie în structură.

Sintaxa unui astfel de algoritm este:



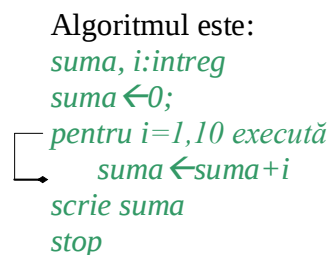
Pașii algoritmului:

1. - se inițializează contorul (în general acesta pornește de la 0 sau 1)
2. - se execută instrucțiunile
3. - se incrementează (decrementează) contorul
4. - se testează dacă contorul nu a ajuns la valoarea finală, caz în care se iese din structură. Dacă nu s-a ajuns la valoarea finală se trece la pasul 2.
5. algoritmul continuă până s-a ajuns la valoarea finală

Observație: Variabila contor va fi întotdeauna de tip întreg sau natural

Exemplu:

Să se afișeze suma primelor 10 numere naturale.



B. Structuri cu număr necunoscut de repetiții

Acest tip de structură este de două feluri:

- cu test inițial
- cu test final

B.1. Structuri repetitive cu test inițial

Au următoarea sintaxă:

┌ CÂT TIMP expresie_logică EXECUTĂ
└ INTRUCȚIUNI

Pașii algoritmului:

1. – se evaluează condiția logică
2. – dacă aceasta este adevărată se trece la pasul 3. Dacă este falsă se trece la pasul 4
3. – se execută instrucțiunile
4. – se iese din structură

Exemplu:

Se citesc numere întregi până la întâlnirea lui 0. Să se afișeze suma lor.

suma ← 0
citeste nr
┌ cât timp nr <> 0 executa
└ citeste nr
 suma ← suma + nr
scrie suma
stop

B.2. Structuri repetitive cu test final

Au următoarea sintaxă:

┌ REPETĂ
└ INTRUCȚIUNI
 PÂNĂ CÂND expresie_logică

Pașii algoritmului:

1. – se execută instrucțiunile
2. – se evaluează expresia. Dacă aceasta este adevărată se trece la pasul 3. Dacă este falsă se trece la pasul 4
3. – se execută instrucțiunile
4. – se iese din structură

Exemplu:

Se citesc numere intregi până la întâlnirea lui 0. Să se afișeze câte numere s-au citit.

```
cate=0  
repetă  
citeste nr  
cate ← cate+1  
pana cand nr=0  
scrie cate  
stop
```