

Limbajul Pseudocod

Permite descrierea într-un limbaj mai puțin strict a principalelor structuri de control ale unui algoritm.

1. Citirea

Pseudocod:

citește v_1, v_2, \dots

C++
`cin >> v1 >> v2 >> ...;`

2. Afișarea

scrie e_1, e_2, \dots

`cout << e1 << e2 << ...;`

3. Atribuire

$\text{variabilă} \leftarrow \text{expr}$

`variabilă = expr;`

4. Structura condițională (de decizie)

4A) dacă condiție atunci
secvență

```
if(cond)  
    secvență;
```

4B) dacă condiție atunci
 secv-adevar
 altfel
 secv-fals

```
if(cond)  
    secv-adevar;  
else  
    secv-fals;
```

Ex. dacă $x = 0$ atunci
 $a \leftarrow b + 1$
 $x \leftarrow a * a$
 altfel
 $a \leftarrow b / x$
 $x \leftarrow a * a$

→

```
if(x == 0)  
{  
    a = b + 1;  
    x = a * a;  
}  
else  
{  
    a = b / x;  
    x = a * a;  
}
```

5) Repetitivă cu test initial

[cât timp condiție execută
secvență]

6) Repetitivă cu test final

[repetă
secvență
până când condiție]

7) Repetitivă cu conținut

7A) [pentru $cnt \leftarrow li, ef$ execută
secvență]

while (cond)
secvență

do
{
secvență
} while (negare condiție)

for ($cnt = li; cnt \leq ef; cnt++$)
secvență;

7B)

pentru $\text{cnt} \leftarrow \text{li}, \text{lf}; -1$ execută
secvență

Obs: În pseudocod, spre deosebire de C++
NU este permisă modificarea cnt , li sau
 lf pe secvența din interiorul for-ului.

for($\text{cnt} = \text{li}; \text{cnt} \geq \text{lf}; \text{cnt} --$)
secvență;

Reguli de transformare între repetitive

\exists 4 reguli fixe, ce pot fi aplicate în orice
condiții.

Din păcate, orice regulă care NU face parte
dintre acestea, trebuie analizată atent și
se bazează pe un artificiu, formula de calcul,
caz particular, etc.

1) De la repetitivă cu test inițial pe repetitivă cu test final:



Obs: Dacă dăm putea scrie și o versiune mai non-standard a lui repetă... până când și avem
execută
cât timp condiție.

(este acceptată special pentru programatori de C++.

Evident în acest caz NU veți mai nega condiția)

2) De pe repetitivă cu test final pe repetitivă cu test inițial

repetă
secvența
până când condiție



Secvență

cât timp neg condiție există
secvență

3) De pe for pe cât timp:

pentru $cnt \leftarrow li, lf$ execută
secvență

$cnt \leftarrow li$

cât timp $cnt \leq lf$ execută
secvență
 $cnt \leftarrow cnt + 1$

4) De pe for pe repetă.. până când

pentru $cnt \leftarrow li, lf$ execută
secvență

$cnt \leftarrow li$
dacă $cnt \leq lf$ execută

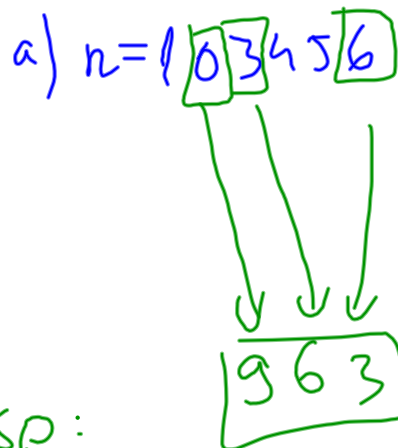
secvență
 $cnt \leftarrow cnt + 1$

cât timp $cnt \leq lf$

```

citește n (număr natural)
z ← 0
p ← 1
cât timp n > 0 execută
  c ← n % 10
  n ← [n / 10]
  dacă c % 3 = 0 atunci
    z ← z + p * (9 - c)
    p ← p * 10
scrie z

```



rsp:

Urmărim evoluția variabilelor cu tabel.

Ne dam seama că alg. formează un număr plecând de la cifre divizibile cu 3 ale celui dat, scăzute din 9 și păstrând ordinea.