Programarea calculatoarelor

#5 C++
Instrucţiunile limbajului C++

Adrian Runceanu
www.runceanu.ro/adrian

2016

Curs 5

Instrucţiunile limbajului C++

5. Instrucţiunile limbajului C++

5.1. Instrucţiunea **vidă**

Limbajul C++ are câteva instrucţiuni cu ajutorul cărora se pot construi programe. Acestea sunt:

Instrucţiunea vidă: `;`

Instrucţiunea compusă: `{ . . . }`

este delimitată de **{** şi se termină cu **}.**

5. Instrucţiunile limbajului C++

5.1. Instrucţiunea **vidă**
5.2. Instrucţiunea **compusă**
5.3. Instrucţiunea **expresie**
5.4. Instrucţiunea **if**
5.5. Instrucţiunea **while**
5.6. Instrucţiunea **do while**
5.7. Instrucţiunea **for**
5.8. Instrucţiunea **switch**
5.9. Instrucţiunea **break**
5.10. Instrucţiunea **continue**
5.11. Instrucţiunea **goto**
5.12. Instrucţiunea **return**

5.2. Instrucţiunea **compusă**

Uneori programele trebuie să efectueze una sau mai multe instrucţiuni atunci când o condiţie este îndeplinită (de exemplu într-o instrucţiune **if**) şi alte instrucţiuni când condiţia nu este îndeplinită.

Sau atunci când o condiție se evaluează într-o structură (instrucțiune) repetitivă – de tip **while**, **do while** sau **for**, iar prelucrările din acea structură pot să fie compuse din una sau mai multe instrucțiuni.

5.2. Instrucţiunea **compusă**

Limbajul **C++** consideră instrucţiunile ca fiind ***instrucţiuni simple*** şi ***instrucţiuni compuse***:

O ***instrucţiune simplă*** este de fapt o singură instrucţiune, cum ar fi aceea *de atribuire* sau *de apel al unei funcţii standard* (de exemplu funcţia ***cout***).

O ***instrucţiune compusă*** este alcătuită din două sau mai multe instrucţiuni incluse între acolade.

5. Instrucţiunile limbajului C++

5.2. Instrucţiunea **vidă**
5.2. Instrucţiunea **compusă**
5.3. Instrucţiunea **expresie**
5.4. Instrucţiunea **if**
5.5. Instrucţiunea **while**
5.6. Instrucţiunea **do while**
5.7. Instrucţiunea **for**
5.8. Instrucţiunea **switch**
5.9. Instrucţiunea **break**
5.10. Instrucţiunea **continue**
5.11. Instrucţiunea **goto**
5.12. Instrucţiunea **return**

5.3. Instrucţiunea **expresie**

Instrucţiunea expresie:

Are 3 forme:

```
expresie;
```

a) instrucţiunea de atribuire
b) instrucţiunea de apel de funcţie
c) instrucţiunea de incrementare / decrementare

5.3. Instrucţiunea **expresie**

a) instrucţiunea de atribuire

variabila = expresie;

sau

variabila operator = expresie;

Exemplu:

```
int x, y, z;
z = x + 5 * y;
x + = 10; (semnificație: x = x + 10;)
```

5.3. Instrucţiunea **expresie**

b) instrucţiunea de apel de funcţie

nume_functie(pa_1, pa_2, . . . ,pa_n);

unde pa_1, pa_2, . . ., pa_n *sunt parametrii actuali ai funcţiei* (adică valorile cu care se va lucra în funcţia respectivă la apelul funcţiei).

Exemplu:

```
maxim (int a, int b); // apelul functiei maxim care are doi parametri actuali de tip întreg
```

5.3. Instrucţiunea **expresie**

c) instrucţiunea de incrementare/decrementare

Exemplu:

```
int i, j, k;
i++;
--j;
k++ + --i;
```

variabila ++;
++ variabila;
variabila --;
-- variabila;

5. Instrucţiunile limbajului C++

5.4. Instrucţiunea if

Instrucţiunea if

(instrucţiune de decizie sau condiţională)

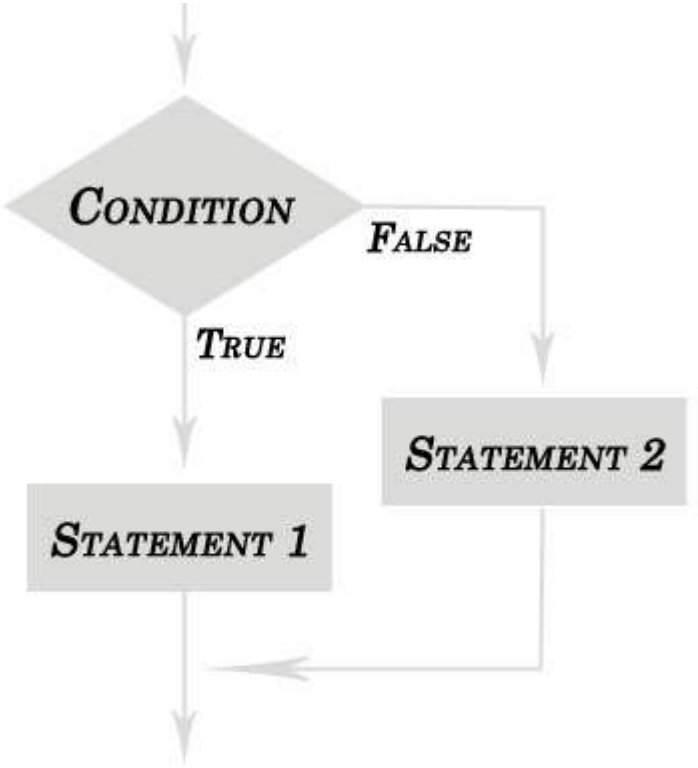
```
if (expresie) instructiune;
```

Are două forme:

```
if (expresie)
        instructiune1;
else
        instructiune2;
```

5.4. Instrucţiunea if

Instrucţiunea if



Observaţie:

În limbajul C++, spre deosebire de limbajul PASCAL, nu există cuvântul cheie **THEN**.

Exemple de programe C++

Enunț:

Să se calculeze perimetrul şi aria unui triunghi oarecare dacă se cunosc laturile triunghiului.

Pas 1: Stabilim care sunt datele de intrare, adică cele care vor fi prelucrate cu ajutorul algoritmului, împreună cu datele de ieşire.

În cazul problemei date, avem:

Date de intrare: a, b, şi c numere reale ce reprezintă laturile triunghiului.

Date de ieşire: p şi S numere reale ce reprezintă perimetrul si aria triunghiului dat.

Exemple de programe C++

Pas 2: Analiza problemei

Stabilim condiţiile pe care trebuie să le îndeplinească datele de intrare pentru a fi prelucrate în cadrul algoritmului.

În cadrul problemei pe care o avem de rezolvat, cunoaştem formula lui Heron pentru calculul ariei unui triunghi dacă se cunosc laturile sale:

$$S = \sqrt{p(p-a)(p-b)(p-c)}$$

unde p reprezintă semiperimetrul triunghiului.

Exemple de programe C++

Pas 3: Scrierea algoritmului în pseudocod:

real a, b, c, p, S
citeşte a, b, c
p <- a + b + c
scrie ‘Perimetrul triunghiului este ‘, p
p <- p / 2
$S \leftarrow \sqrt{p*(p-a)*(p-b)*(p-c)}$
scrie ‘Aria triunghiului este ’, S
stop

Exemple de programe C++

Pas 4:

Implementarea algoritmului în limbajul de programare C++:

```
#include<iostream.h>
#include<math.h>
int main(void)
{
   float a, b, c, p, S;
   cin>>a; cin>>b; cin>>c;
   p = a + b + c;
   cout<<" Perimetrul este = "<<p;
   p = p / 2;
   S = sqrt(p*(p-a)*(p-b)*(p-c));
   cout<<"Aria este = "<<S;
}
```

Exemple de programe C++

Pas 5: Testarea algoritmului pe date de intrare diferite şi verificarea rezultatelor.

Exemplul 1:

Pentru valorile a=2, b=3, c=4, obţinem următoarele rezultate:

Perimetrul este = 9

Aria este = 1.369306

Exemplul 2:

Pentru valorile a=12, b=4, c=10, obţinem următoarele rezultate:

Perimetrul este = 26

Aria este = 5.196152

Exemple de programe C++

Enunţ:

Să se calculeze valoarea funcţiei f(x), ştiind că x este un număr real introdus de la tastatură:

$$f(x)=\begin{cases} -6x+20, & daca \;\; x\in(-\infty,-7] \\ x+30, & daca \;\; x\in(-7,0] \\ \sqrt{x}+2, & daca \;\; x>0 \end{cases}$$

Pas 1:

Date de intrare: x număr real

Date de iesire: f număr real, reprezentând valoarea funcţiei date.

Exemple de programe C++

Pas 2: Analiza problemei

Stabilim condiţiile pe care trebuie să le îndeplinească datele de intrare pentru a fi prelucrate în cadrul algoritmului.

Căutăm cazurile particulare.

În cadrul problemei pe care o avem de rezolvat, verificăm condiţiile date în expresia funcţiei:

1) *Dacă x <= -7*, atunci funcţia are valoarea: -6x+20
2) *Daca x > -7 si x <= 0*, atunci funcţia are valoarea: x+30
3) *Daca x > 0*, atunci funcţia are valoarea: sqrt(x)+2

Exemple de programe C++

Pas 3:

Scrierea algoritmului în pseudocod:

```
real x, f
citeşte x
dacă x <= -7 atunci
        f  <-  -6 * x + 20
altfel
  dacă x > -7 şi x <= 0 atunci
        f <-  x + 30
   altfel
        f <-  sqrt(x) + 2
   sfârşit dacă
sfarşit dacă
scrie f
stop
```

Exemple de programe C++

Pas 4:

Implementarea algoritmului în limbajul de programare C++:

```
#include<iostream.h>
#include<math.h>
int main(void)
{
   float x, f;
   cin>>x;
   if( x <= -7 )    f = -6 * x + 20;
   else
        if( x > -7 && x <= 0 )
                f = x + 30;
        else
                f = sqrt(x) + 2;
   cout<<"f = "<<f;
}
```

Exemple de programe C++

Pas 5: Testarea algoritmului pe date de intrare diferite şi verificarea rezultatelor.

Exemplul 1:

Pentru valoarea x=2 obţinem următorul rezultat:

f= 3.414214

Exemplul 2:

Pentru valoarea x=-24 obţinem următorul rezultat:

f= 164

Exemple de programe C++

Enunţ:

Se dau trei numere întregi a,b,c. Să se afişeze în ordine crescătoare.

Exemplu: Dacă a = 12, b = 2, c = 9, atunci obţinem a = 2, b = 9, c = 12

Pas 1:

Date de intrare: a, b, c numere întregi

Date de iesire: a, b, c în ordine crescătoare

Exemple de programe C++

Pas 2: Analiza problemei

1) *Comparăm primele două numere a şi b*, dacă a este mai mare decât b atunci vom interschimba cele două valori.
2) *Comparăm următoarele două numere b şi c*, dacă b este mai mare decât c atunci vom interschimba cele două valori.
3) *Comparăm din nou cele două numere a şi b*, dacă a este mai mare decât b atunci vom interschimba cele două valori.

Exemple de programe C++

Pas 3:

Scrierea algoritmului în pseudocod:

```
întreg a, b, c, aux
citeşte a, b, c
dacă a > b atunci
        aux <- a
        a <- b
        b <- aux
sfârşit dacă
dacă b > c atunci
        aux <- b
        b <- c
        c <- aux
sfârşit dacă
dacă a > b atunci
        aux <- a
        a <- b
        b <- aux
sfârşit dacă
scrie a, b, c
stop
```

Exemple de programe C++

Pas 4: Implementarea algoritmului în limbajul de programare C++:

```
#include<iostream.h>
int main(void)
{
    int a, b, c, aux;
    cin>>a; cin>>b; cin>>c;
    if( a > b ) {
        aux=a;
        a=b;
        b=aux;
    }
    if( b > c ){
        aux=b;
        b=c;
        c=aux;
    }
    if( a > b ){
        aux=a;
        a=b;
        b=aux;
    }
    cout<<a<<" "<<b<<" "<<c;
}
```

Exemple de programe C++

Pas 5: Testarea algoritmului pe date de intrare diferite şi verificarea rezultatelor.

Exemplul 1:

Pentru valorile a=11, b=7, c=10 obţinem următorul rezultat:

7 10 11

Exemplul 2:

Pentru valorile a=2, b=17, c=5 obţinem următorul rezultat:

2 5 17

5. Instrucţiunile limbajului C++

5.1. Instrucţiunea **vidă**
5.2. Instrucţiunea **compusă**
5.3. Instrucţiunea **expresie**
5.4. Instrucţiunea **if**
5.5. Instrucţiunea **while**
5.6. Instrucţiunea **do while**
5.7. Instrucţiunea **for**
5.8. Instrucţiunea **switch**
5.9. Instrucţiunea **break**
5.10. Instrucţiunea **continue**
5.11. Instrucţiunea **goto**
5.12. Instrucţiunea **return**

5.5. Instrucţiunea **while**

Instrucţiunea while

(instrucţiune repetitivă cu test iniţial)

Are următoarea formă:

```
while (expresie)
        instructiune;
```

unde instrucţiune poate fi:

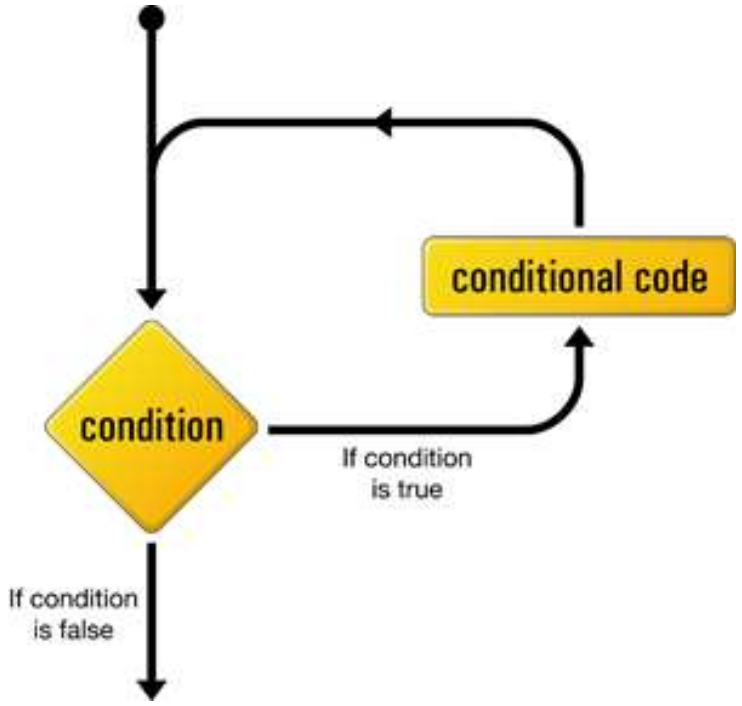
- instrucţiunea vidă
- instrucţiunea simplă
- sau instrucţiunea compusă

5.5. Instrucţiunea **while**

Instrucţiunea while

(instrucţiune repetitivă cu test iniţial)

Funcţionarea unei astfel de instrucţiuni se bazează pe evaluarea expresiei date şi executarea repetată a instrucţiunii ***cât timp*** expresia este îndeplinită.



5.5. Instrucţiunea **while**

Exemplu:

Prezentăm în continuare un program în limbajul C/C++, care calculează suma primelor n numere întregi, cu n<=10:

```
#include<iostream.h>
int main(void)
{
    int i, n, s=0;
    cout<<"Dati numarul n = ";
    cin>>n;
    i = 1;
    while( i <= n )
    {
        s = s + i;
        i++;
    }
    cout<<"Suma primelor "<<n<<"
numere intregi este "<<s;
    }
```

5.5. Instrucţiunea **while**

La execuţia acestui program se poate obţine următorul rezultat:

```
Dati numarul n = 5
Suma primelor 5 numere intregi este 15
```

5.5. Instrucţiunea **while**

Folosind facilităţile limbajului C/C++, se poate scrie aceeaşi secvenţă de program într-o formă prescurtată şi chiar mai eficientă:

```
#include<iostream.h>
int main(void)
{
    int i=1, n, s=0;
    cout<<"Dati numarul n = ";
    cin>>n;
    while(i<=n)  s+=i++;
     cout<<"Suma primelor
"<<n<<" numere intregi este
"<<s;
    }
```

5.5. Instrucţiunea **while**

Observaţie:

Pentru ca un ciclu repetitiv să se execute încontinuu (la infinit), se poate utiliza o buclă infinită prin *introducerea ca expresie a unei valori diferite de 0*.

***while** (1)* este o buclă infinită care se va executa până când de la tastatură se va întrerupe execuţia prin apăsarea tastelor **CTRL+C**.

Exemple de programe C++

Enunţ:

Să se citească un număr natural n. Să se scrie un algoritm care afişează toţi divizorii numărului dat.

Exemplu: Pentru n = 12, mulţimea divizorilor este formată din valorile 1, 2, 3, 4, 6, 12.

Pas 1: Stabilim care sunt datele de intrare, împreună cu datele de ieşire.

În cazul problemei date, avem:

Date de intrare: n număr natural

Date de ieşire: divizorii numărului n

Exemple de programe C++

Pas 2: Analiza problemei

În cadrul problemei pe care o avem de rezolvat, verificăm condiţia ca un număr să fie divizor al altui număr şi anume:

i este divizor al numărului n dacă se împarte exact la el, adică dacă este adevărată expresia n % i = 0.

Pentru a găsi toţi divizorii numărului n dat, vom da valori lui i, pornind de la valoarea 1 până la valoarea n.

Deci vom utiliza o **structură repetitivă**.

Exemple de programe C++

Pas 3:

Scrierea algoritmului în pseudocod:

```
natural n, i
citeşte n
i <- 1
cât timp i <= n execută
    dacă n % i = 0 atunci
        scrie i
    sfârşit dacă
    i <- i + 1
sfârşit cât timp
stop
```

Exemple de programe C++

Pas 4:

Implementarea algoritmului în limbajul de programare C++:

```
#include<iostream.h>
int main(void)
{
   int n, i;
   cin>>n;
   i = 1;
   while( i <= n )
   {
        if( n % i == 0 )
              cout<<i<<" ";
        i = i + 1;
   }
}
```

Exemple de programe C++

Pas 5: Testarea algoritmului pe date de intrare diferite şi verificarea rezultatelor.

Exemplul 1:

Pentru valoarea n=12 obţinem următorul rezultat:

1 2 3 4 6 12

Exemplul 2:

Pentru valoarea n=18 obţinem următorul rezultat:

1 2 3 6 9 18

Exemple de programe C++

Enunţ:

Să se citească un număr natural n. Să se scrie un algoritm care verifică dacă numărul dat este sau nu **număr prim**. Un număr n este prim dacă are ca divizori doar valorile 1 şi n.

Exemplu:

Pentru n = 7, se va afişa mesajul ‘numărul este prim’,

iar pentru n = 22, se va afişa mesajul ‘numărul NU este prim’.

Pas 1: Stabilim care sunt datele de intrare, adică cele care vor fi prelucrate cu ajutorul algoritmului, împreună cu datele de ieşire.

În cazul problemei date, avem:

Date de intrare: n număr natural

Date de ieşire: număr prim sau nu

Exemple de programe C++

Pas 2: Analiza problemei

Vom presupune, la începutul problemei, că numărul n dat este prim, şi vom specifica acest lucru cu ajutorul unei variabile de tip logic, căreia îi vom da valoarea 'adevărat'.

Apoi vom evalua, pe rând, toate valorile începând cu valoarea 2 şi până la n/2, ca să determinăm dacă sunt divizori ai numărului n dat.

Dacă găsim un singur divizor printre aceste numere, atunci vom acorda valoarea 'fals' variabilei de tip logic.

La sfârşit vom verifica care este valoarea variabilei de tip logic şi vom afişa un mesaj corespunzător.

Exemple de programe C++

Pas 3:

Scrierea algoritmului în pseudocod:

```
natural n, i
logic p
citeşte n
p <-  adevărat
i <- 2
cât timp i <= n/2 execută
    dacă n % i = 0 atunci
            p <-  fals
    sfârşit dacă
    i  <-  i + 1
sfârşit cât timp
dacă p = adevărat atunci
            scrie 'Numarul este prim'
altfel
            scrie 'Numarul NU este prim'
sfârşit dacă
stop
```

Exemple de programe C++

Pas 4:

Implementarea algoritmului în limbajul de programare C++:

```
#include<iostream.h>
int main(void)
{
    int n, i, p;
    cin>>n;
    p = 1;
    i = 2;
    while( i <= n/2 )
    {
        if( n % i == 0 )   p = 0;
        i = i + 1;
    }
    if( p == 1 ) cout<<"Numarul este PRIM";
    else cout<<"Numarul NU este PRIM";
}
```

Exemple de programe C++

Pas 5: Testarea algoritmului pe date de intrare diferite şi verificarea rezultatelor.

Exemplul 1:

Pentru valoarea n=12 obţinem următorul rezultat:

Numarul NU este PRIM

Exemplul 2:

Pentru valoarea n=7 obţinem următorul rezultat:

Numarul este PRIM

Probleme propuse spre rezolvare:

1) Pentru n cunoscut, să se calculeze f_n, termenul de rangul n din **şirul lui Fibonacci**, ştiind că:

$f_0 = 1; f_1 = 1; f_n = f_{n-1} + f_{n-2}$ pentru orice valoare n >= 2.

Exemplu:

Date de intrare: 8
Date de ieşire: 21 (1, 1, 2, 3, 5, 8, 13, 21)

Probleme propuse spre rezolvare:

2) Se dau trei numere. Determinaţi şi afişaţi cmmdc al lor (cmmdc = cel mai mare divizor comun).

Exemplu:

Date de intrare: 12 32 38
Date de ieşire: 2

Probleme propuse spre rezolvare:

3) Se dă numărul n, să se afişeze toate numerele mai mici ca el, prime cu el.
Doua numere sunt prime intre ele daca cel mai mare divizor comun al lor este 1.

Exemplu:

Date de intrare: 10
Date de ieşire: 1 3 7 9

Pentru alte informații teoretice și aplicative legate de acest capitol se recomandă următoarele referințe bibliografice:

1. Adrian Runceanu, Mihaela Runceanu, ***Noțiuni de programare în limbajul C++***, Editura Academica Brâncuși, Târgu-Jiu, 2012 (www.utgjiu.ro/editura)

2. Adrian Runceanu, **Programarea şi utilizarea calculatoarelor**, Editura Academica Brâncuși, Târgu-Jiu, 2003 (www.utgjiu.ro/editura)

3. Octavian Dogaru, **C++ - teorie şi practică**, volumul I, Editura Mirton, Timişoara, 2004 (www.utgjiu.ro/editura)

Întrebări?