

Universitatea "Constantin Brâncuși" din Târgu-Jiu
Facultatea de Inginerie
Departamentul de Automatică, Energie și Mediu



Programarea calculatoarelor

Lect.dr. Adrian Runceanu

Curs 5

Instrucțiunile limbajului C++

5. Instrucțiunile limbajului C++

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

5.1. Instrucțiunea **vidă**

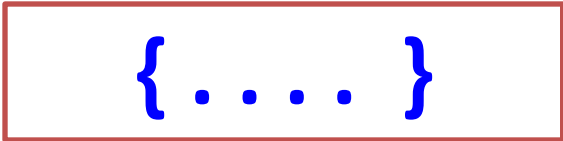
Limbajul C++ are câteva instrucțiuni cu ajutorul cărora se pot construi programe.

Acestea sunt:

Instrucțiunea vidă:



Instrucțiunea compusă:



este delimitată de { și se termină cu }.

5. Instrucțiunile limbajului C++

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

5.2. Instrucțiunea **compusă**

Uneori programele trebuie să efectueze una sau mai multe instrucțiuni atunci când o condiție este îndeplinită (de exemplu într-o instrucțiune **if**) și alte instrucțiuni când condiția 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. Instrucțiunea **compusă**

Limbajul **C++** consideră instrucțiunile ca fiind *instrucțiuni simple* și *instrucțiuni compuse*:

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

O *instrucțiune compusă* este alcătuită din două sau mai multe instrucțiuni incluse între acolade.

5. Instrucțiunile limbajului C++

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

5.3. Instrucțiunea **expresie**

Instrucțiunea expresie:

Are 3 forme:

```
expresie;
```

- a) instrucțiunea de atribuire
- b) instrucțiunea de apel de funcție
- c) instrucțiunea de incrementare / decrementare

5.3. Instrucțiunea **expresie**

a) instrucțiunea 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. Instrucțiunea **expresie**

b) instrucțiunea de apel de funcție

```
nume_funcție(pa1, pa2, . . . , pan);
```

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

Exemplu:

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

5.3. Instrucțiunea **expresie**

c) instrucțiunea de incrementare/decrementare

Exemplu:

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

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

5. Instrucțiunile limbajului C++

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

5.4. Instrucțiunea **if**

Instrucțiunea if

(instrucțiune de decizie sau condițională)

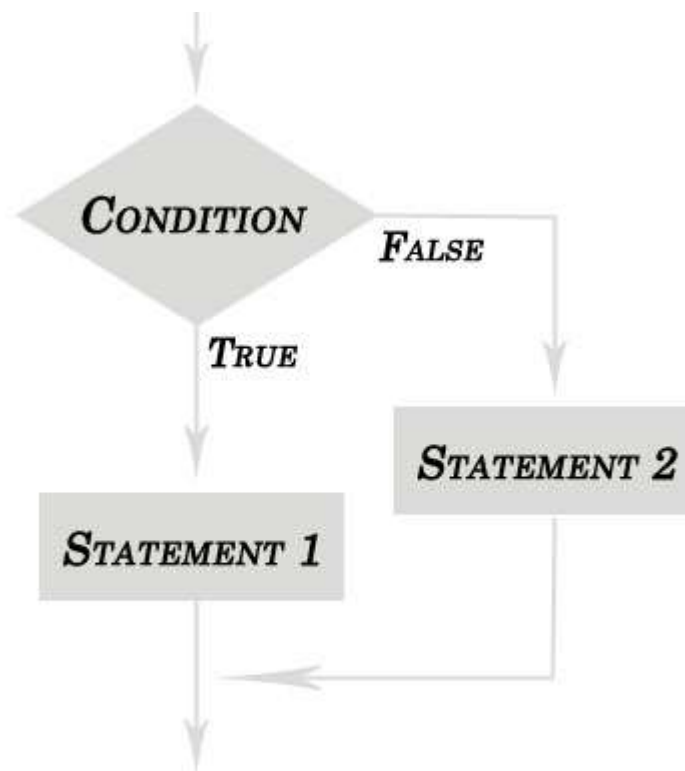
```
if (expresie) instructiune;
```

Are două forme:

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

5.4. Instrucțiunea **if**

Instrucțiunea **if**



Observație:

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

Exemple de programe C++

Enunț:

Să se calculeze perimetrul și 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 ieșire**.

În cazul problemei date, avem:

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

Date de ieșire: p și S numere reale ce reprezintă perimetrul și aria triunghiului dat.

Exemple de programe C++

Pas 2: Analiza problemei

Stabilim condițiile 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, cunoaștem **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

citește a, b, c

p ← a + b + c

scrie 'Perimetrul triunghiului este ', p

p ← p / 2

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

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 și verificarea rezultatelor.

Exemplul 1:

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

Perimetrul este = 9

Aria este = 1.369306

Exemplul 2:

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

Perimetrul este = 26

Aria este = 5.196152

Exemple de programe C++

Enunț:

Să se calculeze valoarea funcției $f(x)$, știind că x este un număr real introdus de la tastatură:

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

Pas 1:

Date de intrare: x număr real

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

Exemple de programe C++

Pas 2: Analiza problemei

Stabilim condițiile 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 condițiile date în expresia funcției:

- 1) *Dacă* $x \leq -7$, atunci funcția are valoarea: $-6x+20$
- 2) *Dacă* $x > -7$ *și* $x \leq 0$, atunci funcția are valoarea: $x+30$
- 3) *Dacă* $x > 0$, atunci funcția are valoarea: $\sqrt{x}+2$

Exemple de programe C++

Pas 3:

Scrierea
algoritmului în
pseudocod:

```
real x, f  
citește x  
dacă  $x \leq -7$  atunci  
     $f \leftarrow -6 * x + 20$   
altfel  
    dacă  $x > -7$  și  $x \leq 0$  atunci  
         $f \leftarrow x + 30$   
    altfel  
         $f \leftarrow \text{sqrt}(x) + 2$   
    sfârșit dacă  
sfârșit 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 și verificarea rezultatelor.

Exemplul 1:

Pentru valoarea $x=2$ obținem următorul rezultat:

$$f = 3.414214$$

Exemplul 2:

Pentru valoarea $x=-24$ obținem următorul rezultat:

$$f = 164$$

Exemple de programe C++

Enunț:

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

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

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 și b*, dacă a este mai mare decât b atunci vom interschimba cele două valori.
- 2) *Comparăm următoarele două numere b și c*, dacă b este mai mare decât c atunci vom interschimba cele două valori.
- 3) *Comparăm din nou cele două numere a și 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
citește a, b, c
dacă a > b atunci
    aux <- a
    a <- b
    b <- aux
sfârșit dacă
dacă b > c atunci
    aux <- b
    b <- c
    c <- aux
sfârșit dacă
dacă a > b atunci
    aux <- a
    a <- b
    b <- aux
sfârșit 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 și verificarea rezultatelor.

Exemplul 1:

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

7 10 11

Exemplul 2:

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

2 5 17

5. Instrucțiunile limbajului C++

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

5.5. Instrucțiunea **while**

Instrucțiunea **while**

(instrucțiune repetitivă cu test inițial)

Are următoarea formă:

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

unde **instrucțiune** poate fi:

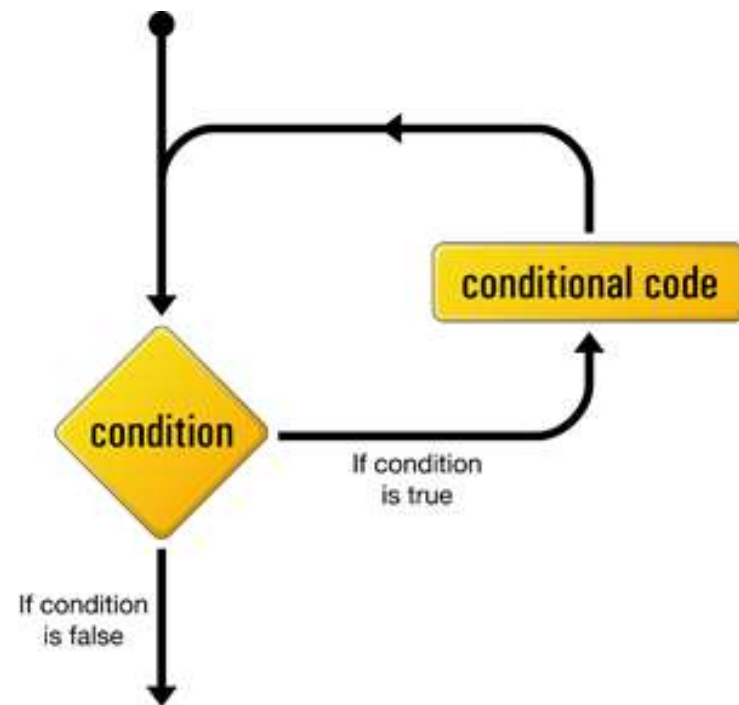
- **instrucțiunea vidă**
- **instrucțiunea simplă**
- **sau instrucțiunea compusă**

5.5. Instrucțiunea **while**

Instrucțiunea **while**

(instrucțiune repetitivă cu test inițial)

Funcționarea unei astfel de instrucțiuni se bazează pe evaluarea expresiei date și executarea repetată a instrucțiunii **cât timp expresia este îndeplinită**.



5.5. Instrucțiunea **while**

Exemplu:

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

```
#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<<endl;
}
```

5.5. Instrucțiunea **while**

La execuția acestui program se poate obține următorul rezultat:

Dati numarul $n = 5$

Suma primelor 5 numere intregi este 15

5.5. Instrucțiunea **while**

Folosind
facilitățile limbajului
C/C++, se poate
scrie aceeași
secvență de
program într-o
formă prescurtată și
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<<endl;  
}
```

5.5. Instrucțiunea **while**

Observație:

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 execuția prin apăsarea tastelor **CTRL+C**.

Exemple de programe C++

Enunț:

Să se citească un număr natural n . Să se scrie un algoritm care afișează toți divizorii numărului dat.

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

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

În cazul problemei date, avem:

Date de intrare: n număr natural

Date de ieșire: divizorii numărului n

Exemple de programe C++

Pas 2: Analiza problemei

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

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

Pentru a găsi toți 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  
citește n  
i <- 1  
cât timp i <= n execută  
  dacă n % i = 0 atunci  
    scrie i  
  sfârșit dacă  
  i <- i + 1  
sfârșit cât timp  
stop
```


Exemple de programe 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;  
    }  
}
```

Pas 4:

Implementarea
algoritmului în
limbajul de
programare C++:

Exemple de programe C++

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

Exemplul 1:

Pentru valoarea $n=12$ obținem următorul rezultat:

1 2 3 4 6 12

Exemplul 2:

Pentru valoarea $n=18$ obținem următorul rezultat:

1 2 3 6 9 18

Exemple de programe C++

Enunț:

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 și n .

Exemplu:

Pentru $n = 7$, se va afișa mesajul 'numărul este prim', iar pentru $n = 22$, se va afișa 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 ieșire.

În cazul problemei date, avem:

Date de intrare: n număr natural

Date de ieșire: număr prim sau nu

Exemple de programe C++

Pas 2: Analiza problemei

Vom presupune, la începutul problemei, că numărul n dat este prim, și 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 și până la $n-1$, 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 sfârșit vom verifica care este valoarea variabilei de tip logic și vom afișa un mesaj corespunzător.

Exemple de programe C++

Pas 3:

Scrierea
algoritmului în
pseudocod:

```
natural n, i  
logic p  
citește n  
p <- adevărat  
i <- 2  
cât timp i <= n-1 execută  
    dacă n % i = 0 atunci  
        p <- fals  
    sfârșit dacă  
    i <- i + 1  
sfârșit cât timp  
dacă p = adevărat atunci  
    scrie 'Numarul este prim'  
altfel  
    scrie 'Numarul NU este prim'  
sfârșit 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-1 )
    {
        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 și verificarea rezultatelor.

Exemplul 1:

Pentru valoarea $n=12$ obținem următorul rezultat:

Numarul NU este PRIM

Exemplul 2:

Pentru valoarea $n=7$ obținem următorul rezultat:

Numarul este PRIM

Probleme propuse spre rezolvare:

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

$f_0 = 1; f_1 = 1; f_p = f_{p-1} + f_{p-2}$ pentru orice valoare $p \geq 2$.

Exemplu:

Date de intrare: 8

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

Probleme propuse spre rezolvare:

2) Se dau trei numere. Determinați și afișați cmmdc al lor (cmmdc = cel mai mare divizor comun).

Exemplu:

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

Probleme propuse spre rezolvare:

3) Se dă numărul n , să se afișeze toate numerele mai mici ca el, prime cu el.

Doua numere sunt prime între ele dacă cel mai mare divizor comun al lor este 1.

Exemplu:

Date de intrare: 10

Date de ieșire: 1 3 7 9

Pentru alte informații teoretice și practice 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 și utilizarea calculatoarelor**, Editura Academica Brâncuși, Târgu-Jiu, 2003 (www.utgjiu.ro/editura)

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

Întrebări?