INFORMATICĂ · CLASA A IX-A · ELEMENTELE DE BAZĂ ALE LIMBAJULUI C++ · LECȚIA 3

Operatori și expresii în C++

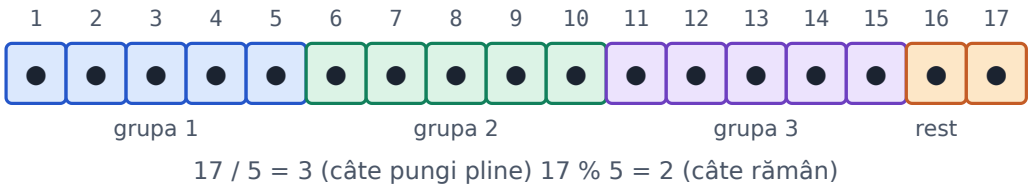
Durată: 60 min Nivel: toate profilurile Înainte: lecțiile ix-02

CE VEI ȘTI DUPĂ LECȚIE

- Folosești corect / și % pe numere întregi și știi de ce extrag cifre.
- Eviți capcana împărțirii întregi la medii și la rapoarte (a / 2.0, cast la double).
- Scrii condiții compuse cu &&, ||, ! și aplici legile lui De Morgan.
- Lucrezi cu caractere ca numere (coduri ASCII): literă mică/mare, caracter-cifră.
- Evaluezi pas cu pas o expresie respectând prioritatea operatorilor, ca la grilele de BAC.

1. De la împărțirea cu rest la `/` și `%`

Gândește-te la o problemă de clasa a III-a: ai 17 bomboane și le pui câte 5 în pungi. Câte pungi umpli? Câte bomboane rămân?



17 bomboane puse câte 5: trei pungi pline, două bomboane rămase

Ai făcut o **împărțire cu rest**: 17 = 5 · 3 + 2. În C++, cele două rezultate se obțin separat:

- `17 / 5` este **câtul**, 3 (când ambii operanzi sunt întregi, zecimalele se taie);
- `17 % 5` este **restul**, 2.

Restul e mereu mai mic decât împărțitorul: `n % 5` poate fi doar 0, 1, 2, 3 sau 4. De aici vin cele mai folosite trucuri:

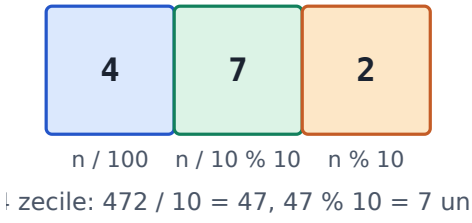
- `n % 2 == 0` verifică dacă n e **par**;
- `a % b == 0` verifică dacă **b divide pe a**;
- `n % 10` este **ultima cifră**, iar `n / 10` este numărul **fără ultima cifră**.

Operatorii aritmetici

Operator	Exemplu	Rezultat	Observație
`+` `-` `*`	`7 * 3`	`21`	
`/` între întregi	`7 / 2`	`3`	împărțire întreagă: se taie zecimalele
`/` cu un real	`7.0 / 2`	`3.5`	dacă măcar un operand e real, rezultatul e real

Operator	Exemplu	Rezultat	Observație
`%`	`7 % 2`	`1`	**doar între întregi**: restul împărțirii
`-` unar	`-x`		schimbă semnul

Cifrele unui număr de trei cifre, fără buclă



Numărul 472: sutele cu n / 100, zecile cu n / 10 % 10, unitățile cu n % 10

```
int n = 472;
int u = n % 10;          // 2 (unitățile)
int z = n / 10 % 10;     // 7 (zecile): întâi tai unitățile (47), apoi iau ultima cifră
int s = n / 100;         // 4 (sutele)
int oglindit = u * 100 + z * 10 + s;   // 274
```

DE CE AȘA?

De ce `n / 10 % 10` dă cifra zecilor? `/` și `%` au aceeași prioritate și se evaluează de la stânga la dreapta: întâi `472 / 10 = 47` (am tăiat unitățile), apoi `47 % 10 = 7` (ultima cifră a ce a rămas). În general, cifra de pe poziția 10^k este `n / 10^k % 10`.

DE ȘTIUT

La numere negative, restul are semnul deîmpărțitului: `-7 % 3` dă `-1` în C++. La BAC se lucrează aproape mereu cu numere naturale, dar e bine să știi.

2. Evaluarea pas cu pas, ca la grilele de BAC

La BAC apar grile de tipul „ce valoare are expresia `11*3/2*2/3`?". Regula: `*`, `/`, `%` au aceeași prioritate și se calculează **de la stânga la dreapta**, iar la fiecare `/` între întregi se taie imediat zecimalele.

Expresia	Pașii	Rezultat
`11*3/2*2/3`	33 → 33/2 = 16 → 32 → 32/3 = 10	**10**
`4+5%7*2`	`%` și `*` înaintea lui `+`: 5%7 = 5, 5*2 = 10, 4 + 10	**14**
`42/10*29/10`	4 → 116 → 11	**11**
`7+5/2`	5/2 = 2, 7 + 2	**9**

Expresia	Pașii	Rezultat
`3+7/4+3`	7/4 = 1, 3 + 1 + 3	**7**

CAPCANĂ

Capcana clasică: la matematică, 11 · 3 / 2 · 2 / 3 = 11. În C++ dă 10, pentru că la `33 / 2` s-a pierdut 0,5, iar pierderea nu se mai recuperează.

3. Atribuirea și prescurtările ei

```
int x = 10;
x += 5;     // x = x + 5  → 15
x -= 3;     // x = x - 3  → 12
x *= 2;     // x = x * 2  → 24
x /= 5;     // x = x / 5  → 4
x %= 3;     // x = x % 3  → 1
x++;        // x = x + 1  → 2
x--;        // x = x - 1  → 1
```

Pre- și post-incrementare (apar la grile):

```
int a = 5, b, c;
b = a++;    // b primește 5, APOI a devine 6
c = ++a;    // a devine 7, APOI c primește 7
```

După instrucțiunea	a	b	c
`int a = 5;`	5	–	–
`b = a++;`	6	5	–
`c = ++a;`	7	5	7

ATENȚIE — GREȘELI FRECVENTE

Nu scrie niciodată `a++ + ++a` sau `a = a++` într-o singură expresie: în C++ rezultatul e nedefinit (poate diferi de la un compilator la altul). Pe linii separate, `a++;` și `++a;` fac exact același lucru.

4. Conversii de tip: capcana mediei

```
int a = 7, b = 8;
double m1 = (a + b) / 2;          // 7   – împărțirea s-a făcut întreagă, apoi rezultatul s-a convertit
double m2 = (a + b) / 2.0;        // 7.5 – corect
double m3 = (double)(a + b) / 2;  // 7.5 – corect, cu „cast”
int k = 3.99;                      // k = 3: la atribuirea unui real într-un int se taie zecimalele
```

DE CE AȘA?

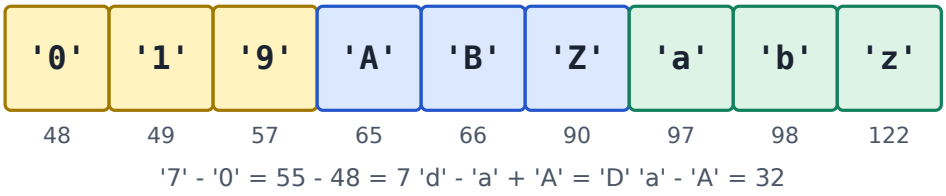
De ce `m1` e 7, deși e `double`? Ca la overflow: expresia din dreapta se calculează **înainte** de atribuire, cu tipurile operanzilor (toate `int`). Rezultatul 7 e deja întreg când ajunge în `m1`. Variabila din stânga nu schimbă modul în care se calculează dreapta.

Reguli de conversie:

- automate: `int` cu `double` dă `double`; `char` cu `int` dă `int` (se folosește codul caracterului);
- explicite (cast): `(double)a / b` sau `double(a) / b` forțează împărțirea reală.

Caracterele sunt numere

Fiecare caracter are un **cod ASCII**. Cifrele, literele mari și literele mici sunt **consecutive**:



Coduri ASCII: '0'...'9' sunt 48...57, 'A'...'Z' sunt 65...90, 'a'...'z' sunt 97...122

```
char c = 'd';
char mare = c - 'a' + 'A';   // 'D': poziția literei în alfabet, mutată în zona literelor mari
char cif = '7';
int valoare = cif - '0';     // 7: din caracter-cifră obținem numărul
```

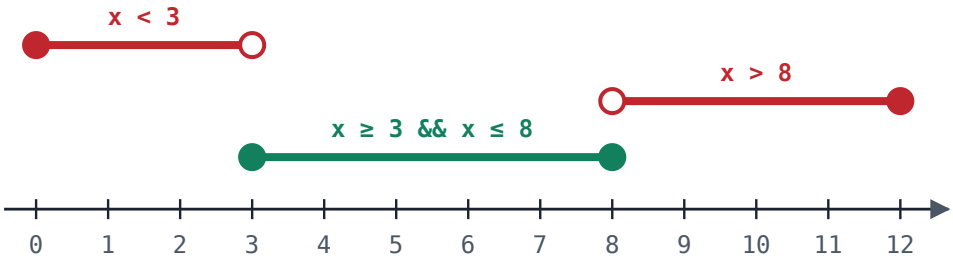
DE CE AȘA?

De ce `c - 'a' + 'A'` și nu `c - 32`? Ambele merg, dar prima variantă se citește singură: „cât de departe e c de 'a', aceeași distanță de la 'A'". Nu trebuie să ții minte niciun cod.

5. Operatori relaționali și logici

Pseudocod	C++
`=`	`==` (comparație!)
`≠`	`!=`
`≤` `≥`	`<=` `>=`
`și`	`&&`
`sau`	`||`
`not`	`!`

```
bool areDouaCifre = (n >= 10 && n <= 99);
bool divizibilCu3sau5 = (n % 3 == 0 || n % 5 == 0);
bool inAfara = (x < 3 || x > 8);     // x NU e în [3, 8]
```



:rde: în interval (&&). Roșu: în afara intervalului (||): negația lui ≥ este <, a lui ≤ este

Pe axă: intervalul [3, 8] (cu &&) și în afara lui (cu ||)

CAPCANĂ

> - `if (x = 5)` **atribuie** 5 lui x, iar condiția e mereu adevărată. Comparația se scrie `x == 5`. - `10 <= n <= 99` compilează, dar nu face ce crezi: întâi se calculează `10 <= n` (0 sau 1), apoi se compară acel 0 sau 1 cu 99, deci expresia e **mereu adevărată**. Scrie `n >= 10 && n <= 99`.

Legile lui De Morgan, în C++: `!(a && b)` este `!a || !b`, iar `!(a || b)` este `!a && !b`. Exemplu: `!(x >= 3 && x <= 8)` este `x < 3 || x > 8`.

Operatorul condițional `?:`

```
int maxim = (a > b) ? a : b;   // dacă a > b, maxim = a; altfel maxim = b
```

6. Funcții matematice (`#include <cmath>`)

Funcție	Ce face	Exemplu
`sqrt(x)`	radical	`sqrt(16)` → 4
`abs(x)`	modul (valoare absolută)	`abs(-7)` → 7
`pow(a, b)`	a^b (rezultat real)	`pow(2, 10)` → 1024
`floor(x)` / `ceil(x)`	rotunjire în jos / în sus	`floor(-2.5)` → −3, `ceil(-2.5)` → −2
`round(x)`	rotunjire la cel mai apropiat întreg	`round(2.5)` → 3

ATENȚIE — GREȘELI FRECVENTE

`pow` lucrează cu numere reale și poate da `99.9999…` în loc de `100`; convertit la `int`, devine 99. Pentru puteri întregi folosește înmulțiri repetate (le facem la structurile repetitive).

Afișarea numerelor reale cu un număr fix de zecimale

```
#include <iomanip>
// ...
cout << fixed << setprecision(2) << 3.14159;   // 3.14
```

7. Prioritatea operatorilor

1. `()` paranteze
2. `!`, `++`, `--`, `-` unar, cast
3. `*`, `/`, `%`
4. `+`, `-`
5. `<`, `<=`, `>`, `>=`
6. `==`, `!=`
7. `&&`
8. `||`
9. `?:`
10. `=`, `+=`, `-=` ...

Când nu ești sigur, **pune paranteze**: nu costă nimic și face codul mai ușor de citit.

8. Greșeli frecvente

ATENȚIE — GREȘELI FRECVENTE

> - Media a două întregi scrisă `(a + b) / 2`: rezultat întreg chiar dacă variabila e `double`. - `=` în loc de `==` în condiții. - `10 <= n <= 99` în loc de `n >= 10 && n <= 99`. - Negarea greșită a unei condiții compuse (uiți să schimbi `&&` în `||`). - `%` aplicat pe numere reale: nu compilează. Pentru partea întreagă a unui real folosește cast: `(int)x % 10`.

9. La BAC

LA BAC

Subiectul I aproape întotdeauna începe cu o grilă de expresii: „indicați valoarea expresiei...” sau „indicați expresia care are valoarea 1 dacă și numai dacă...”. Pentru prima, evaluează pas cu pas, de la stânga la dreapta, tăind zecimalele la fiecare `/` întreg. Pentru a doua, testează fiecare variantă pe 2–3 valori alese inteligent: o valoare din interval, una din afară și **capetele** intervalului.

10. Exersează pe hârtie

Răspunsurile sunt la finalul lecției, după probleme: încearcă întâi singur.

1. Ce valoare are `17 / 5 * 5 + 17 % 5`? Dar `n / 5 * 5 + n % 5`, pentru orice n natural?

2. Ce afișează `cout << 7 / 2 * 2.0;`? Dar `cout << 7 / 2.0 * 2;`?

3. Scrie expresia C++ care verifică dacă `x` **nu** aparține intervalului `[a, b]`.

4. După secvența `int a = 3; int b = a++; int c = ++a;`, ce valori au `a`, `b` și `c`?

5. Variabila `c` de tip `char` conține o cifră (`'0'` ... `'9'`). Scrie o expresie care are ca valoare numărul corespunzător.

6. Care expresie are valoarea 1 dacă și numai dacă numărul natural `n` are **exact trei cifre**?

a) `n / 100 > 0`
b) `n >= 100 && n <= 999`
c) `n / 100 > 0 && n / 1000 == 0`
d) `100 <= n <= 999`

(Atenție: pot fi mai multe corecte.)

7. Ce valoare are expresia `(x % 10 + x / 10 % 10) % 2 == 0` pentru `x = 384` ? Ce verifică, în cuvinte?

8. Variabila `x` este de tip `double` și are valoarea 7.8. Ce afișează `cout << (int)x % 5 << ' ' << round(x);` ?

11. Mai departe (pentru cine vrea mai mult)

Numărul multiplilor dintr-un interval, fără buclă. Câte numere din `[1, b]` se divid cu `k` ? Exact `b / k` . Atunci în `[a, b]` sunt `b / k - (a - 1) / k` . Aceasta e o idee care transformă o buclă de un miliard de pași într-un singur calcul: exact ce cer problemele de concurs.

Rotunjirea împărțirii în sus. Câte autobuze de câte 50 de locuri trebuie pentru `n` elevi? `(n + 49) / 50` , adică `(n + k - 1) / k` . Nu ai nevoie de `ceil` și nici de numere reale.

Operatorii pe biți. `&` , `|` , `^` , `<<` , `>>` lucrează direct pe scrierea în baza 2 a numerelor. De exemplu `n & 1` este 1 pentru numerele impare, iar `1 << k` este 2^k. Sunt foarte rapizi și apar des la concursuri și în programarea de sisteme (drivere, rețele, jocuri).

Unde contează în practică. O greșeală de împărțire întreagă (sau o rotunjire greșită) într-un program financiar înseamnă bani pierduți la fiecare tranzacție. De aceea, aplicațiile bancare lucrează cu sume în **bani întregi** (cenți), nu cu `double` : 10,99 lei se păstrează ca 1099 de bani.

Rezumat

- `/` între întregi = câtul (zecimalele se taie); `%` = restul; împreună extrag cifrele.
- `*` , `/` , `%` se evaluează de la stânga la dreapta, cu tăiere la fiecare `/` întreg.
- Un singur operand real face împărțirea reală: `(a + b) / 2.0` .
- `==` compară, `=` atribuie; `&&` = și, `||` = sau, `!` = not; De Morgan schimbă `&&` cu `||` și neagă fiecare relație.
- `char` se comportă ca un număr (codul ASCII): `c - '0'` , `c - 'a' + 'A'` .

Probleme de lucru (pbinfo)

Începe cu acestea, în ordine: sunt alese să urmeze pașii din lecție.

- [ ] **#102** **cifre** — ușoară
  Ultimele două cifre cu % și /.
  https://www.pbinfo.ro/probleme/102/cifre

- [ ] #467 **numar** — ușoară
  Extragi cifrele unui număr de trei cifre.
  https://www.pbinfo.ro/probleme/467/numar
- [ ] #468 **numar1** — ușoară
  Construiești un număr nou din cifre: atenție la tipul pătratului.
  https://www.pbinfo.ro/probleme/468/numar1
- [ ] #169 **sumcif** — ușoară
  Suma cifrelor unui număr de exact trei cifre, fără buclă.
  https://www.pbinfo.ro/probleme/169/sumcif
- [ ] #2200 **uppercase** — ușoară
  Literă mică → literă mare cu codul ASCII.
  https://www.pbinfo.ro/probleme/2200/uppercase
- [ ] #2199 **afcar** — ușoară
  Codul ASCII al unui caracter.
  https://www.pbinfo.ro/probleme/2199/afcar

Temă (pbinfo)

Trimite soluțiile pe pbinfo și notează punctajul obținut. Bifează ce ai terminat.

- [ ] #2264 **sumacifre2** — ușoară
  https://www.pbinfo.ro/probleme/2264/sumacifre2
- [ ] #2601 **sumapatratecifre** — ușoară
  https://www.pbinfo.ro/probleme/2601/sumapatratecifre
- [ ] #2604 **schimbarecifre** — ușoară
  https://www.pbinfo.ro/probleme/2604/schimbarecifre
- [ ] #815 **lazi** — ușoară
  Câte cutii încap: împărțire întreagă.
  https://www.pbinfo.ro/probleme/815/lazi
- [ ] #3210 **cumparaturi** — ușoară
  Cât și rest, ca în exemplul cu caietele.
  https://www.pbinfo.ro/probleme/3210/cumparaturi
- [ ] #3609 **countmod10** — ușoară
  Câți multipli de 10 sunt între a și b: o formulă, fără buclă.
  https://www.pbinfo.ro/probleme/3609/countmod10
- [ ] #1282 **radical** — ușoară
  sqrt din <cmath>.
  https://www.pbinfo.ro/probleme/1282/radical

- [ ] #1334 **romb** — ușoară
  Teorema lui Pitagora + sqrt.
  https://www.pbinfo.ro/probleme/1334/romb
- [ ] #2602 **ultimacifraapartiiintregi** — ușoară
  Real → întreg prin cast, apoi % 10.
  https://www.pbinfo.ro/probleme/2602/ultimacifraapartiiintregi
- [ ] #2603 **primacifraapartiizecimale** — ușoară
  https://www.pbinfo.ro/probleme/2603/primacifraapartiizecimale

Pentru cei care vor mai mult

Probleme mai grele sau de concurs, pentru cine vrea să meargă dincolo de BAC.

- [ ] #209 **elimcif** — medie
  https://www.pbinfo.ro/probleme/209/elimcif
- [ ] #3611 **sumacifperm** — medie
  Găsește o formulă în loc să scrii toate permutările.
  https://www.pbinfo.ro/probleme/3611/sumacifperm
- [ ] #460 **timp** — medie
  Ore și minute: transformă totul în minute, apoi înapoi cu / și %.
  https://www.pbinfo.ro/probleme/460/timp
- [ ] #461 **timp1** — medie
  https://www.pbinfo.ro/probleme/461/timp1
- [ ] #3928 **abcd** — medie
  Câte numere divizibile cu ceva într-un interval: formula cu / .
  https://www.pbinfo.ro/probleme/3928/abcd

Grile din subiectele de BAC (pbinfo)

Exerciții de tip grilă din variantele de Bacalaureat, rezolvate online pe pbinfo, cu verificare automată a răspunsului.

- [ ] #10 **Exercițiul #10**
  11*3/2*2/3: ordinea operațiilor la împărțirea întreagă (Bac 2009)
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=10
- [ ] #488 **Exercițiul #488**
  4+5%7*2 (Bac 2015)
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=488
- [ ] #498 **Exercițiul #498**
  42/10*29/10 (Bac 2014)
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=498

- [ ] #805 **Exercițiul #805**
  3+7/4+3 (Bac 2017)
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=805

- [ ] #1054 **Exercițiul #1054**
  împărțire cu un operand real (Bac 2020)
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=1054

- [ ] #1377 **Exercițiul #1377**
  22/20*20/2 (Bac 2022)
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=1377

- [ ] #1738 **Exercițiul #1738**
  20/25*20/2 (Bac 2026)
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=1738

- [ ] #7 **Exercițiul #7**
  x aparține unui interval (Bac 2009)
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=7

- [ ] #11 **Exercițiul #11**
  x nu aparține unui interval (Bac 2009)
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=11

- [ ] #15 **Exercițiul #15**
  expresii echivalente, De Morgan
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=15

- [ ] #61 **Exercițiul #61**
  negația unei expresii compuse
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=61

- [ ] #1195 **Exercițiul #1195**
  floor, ceil (Bac 2021)
  https://www.pbinfo.ro/?pagina=itemi-evaluare&id=1195

Citește și exersează în plus

- Operatori C++ https://www.pbinfo.ro/articole/16/operatorii-cpp
- Expresii C++ https://www.pbinfo.ro/articole/15/expresii-cpp
- cppreference — prioritatea operatorilor — tabelul complet al priorităților (în engleză)
  https://en.cppreference.com/w/cpp/language/operator_precedence
- Python Tutor pentru C++ — vezi pas cu pas ce valori primesc variabilele după ++, +=, cast
  https://pythontutor.com/cpp.html

Răspunsuri

Exersează pe hârtie

1. 17. În general `n / 5 * 5 + n % 5 = n`: este chiar teorema împărțirii cu rest, n = 5 · cât + rest.

2. `6` și `7`. În primul caz, `7 / 2` se face întâi, între întregi (3), apoi 3 · 2.0 = 6. În al doilea, `7 / 2.0 = 3.5`, apoi 3.5 · 2 = 7.

3. `x < a || x > b` (sau `!(x >= a && x <= b)`).

4. `a = 5`, `b = 3`, `c = 5`. `b = a++` îi dă lui b valoarea veche (3), apoi a devine 4; `c = ++a` crește întâi a la 5, apoi c primește 5.

5. `c - '0'`.

6. b) și **c)**. Varianta a) e adevărată și pentru numerele de patru cifre sau mai multe; d) e mereu adevărată (vezi capcana din secțiunea 5).

7. Pentru 384: 4 + 8 = 12, iar 12 % 2 = 0, deci expresia are valoarea 1 (adevărat). Verifică dacă **suma ultimelor două cifre** ale lui x este pară.

8. `2 8`. `(int)7.8` este 7, iar `7 % 5 = 2`; `round(7.8)` este 8.