FIȘĂ DE LUCRU · CLASA A IX-A · FIȘA 8 · ACOPERĂ LECȚIA IX.14

Eficiență și recapitulare de clasa a IX-a

Durată: 110 min Nivel: toate profilurile

CE VEI ȘTI DUPĂ LECȚIE

- Numeri pașii unui program și estimezi din restricții dacă trece în timp (O(1), O(log n), O(√n), O(n), O(n²)).
- Înlocuiești o soluție lentă cu una eficientă: formulă, actualizare din mers, proprietate globală, oprire la √n, înjumătățire.
- Combini tiparele din toată clasa a IX-a (cifre, divizori, secvențe, baze) și descrii metoda ca la III.3.

Ce trebuie să știi (o pagină)

Complexitate	Cod tipic	n = 10^6	n = 10^9	Trece în ~1 s dacă
O(1)	formulă: `n * (n + 1) / 2`	1	1	oricând
O(log n)	`n /= 10`, `n /= 2`, Euclid, puterea rapidă	~20	~30	oricând
O(√n)	`for (d = 2; d * d <= n; d++)`	1 000	~31 600	n ≤ 10^{14}
O(n)	o parcurgere a datelor	10^6	10^9 (prea lent)	n ≤ 10^7–10^8
O(n log n)	`for (i…n) for (j = 1; j <= n; j *= 2)`	$2 \cdot 10^7$	—	n ≤ 10^6–10^7
O(n²)	două bucle imbricate până la n	10^{12}	—	n ≤ 10^4

Cum număr pașii	Exemplu
bucle una după alta → **aduni**; imbricate → **înmulțești**	`i = 1..n` și `j = i+1..n` → n(n − 1)/2
variabila se înmulțește / împarte la k la fiecare pas → log	`i = 1, 2, 4, …, ≤ 100` → 7 pași
contează **cazul cel mai rău**	primalitatea: numerele prime fac toți cei √n pași
~10^8 operații simple pe secundă	

SFAT

Cinci trucuri care transformă un program lent în unul rapid: **formula** (Gauss), **actualizarea din mers** (`p *= i` în loc de a recalcula), **o proprietate globală** (suma așteptată − suma citită), **oprirea la √n** și **înjumătățirea** (Euclid, puterea rapidă). Model III.3 a): „citesc numerele unul câte unul, fără să le memorez, și actualizez ...; o singură parcurgere (timp liniar) și câteva variabile (memorie constantă)".

Nivelul 1 · Prezice

★ citește și prezice

P1. De câte ori se execută `cnt++` pentru `n = 100` ? Scrie și complexitatea.

secvența	`cnt`	O(...)
`for (i = 1; i <= n; i++) for (j = 1; j <= n; j++) cnt++;`		
`for (i = 1; i <= n; i++) for (j = i + 1; j <= n; j++) cnt++;`		
`for (i = 1; i <= n; i *= 2) cnt++;`		
`for (i = 1; i * i <= n; i++) cnt++;`		
`for (i = 1; i <= n; i++) for (j = 1; j <= n; j *= 2) cnt++;`		
`for (i = n; i > 0; i /= 3) cnt++;`		

P2. Un program durează 0,1 s pentru $n = 10^5$. Cât va dura pentru $n = 10^6$ dacă are complexitatea O(n)? Dar $O(n^2)$? Dar O(√n)?

P3. Limita de timp este 1 secundă (~10^8 operații). Trece sau nu?

problema	restricția	algoritmul	trece?
primalitate	$n \le 10^9$	`d` de la 2 la n − 1	
primalitate	$n \le 10^{12}$	`d * d <= n`	
toate perechile de numere dintr-un șir	$n \le 10^5$	două bucle imbricate	
toate perechile de numere dintr-un șir	$n \le 3\,000$	două bucle imbricate	
suma cifrelor	$n \le 10^{18}$	`while (n > 0) n /= 10`	
suma 1 + 2 + ... + n	$n \le 10^{18}$	o buclă	

P4. *Ca la BAC.*

```
citește a, n                (numere naturale)
p ← 1
┌cât timp n > 0 execută
│   ┌dacă n % 2 = 1 atunci
│   │   p ← p * a
│   └■
│   a ← a * a
│   n ← [n / 2]
└■
scrie p
```

a. Ce se afișează pentru `a = 2, n = 10`? Dar pentru `a = 3, n = 5`? Ce calculează?

b. De câte ori se execută bucla pentru `n = 10`? Dar pentru `n = 1000` și `n = 10⁹`? Ce legătură are cu baza 2?

P5. Câte iterații face Euclid prin împărțiri pentru `(89, 55)`, `(100, 55)` și `(1000, 1)`? Ce au special 89 și 55? (Dintre toate perechile cu numere până la 100, perechea `(89, 55)` cere cei mai mulți pași.)

P6. De câte ori se execută corpul buclei `for (d = 2; d * d <= n && prim; d++) if (n % d == 0) prim = false;` (cu `prim = (n >= 2)` la început)?

`n`	1 000 000	999 999	1 000 003 (prim)	994 009 = 997²
execuții				

P7. *Recunoaște tiparul.* Ce calculează fiecare secvență, ce dă pentru datele indicate și ce complexitate are?

secvența	datele	rezultat	O(...)
`while (n) { s += n % 10; n /= 10; }`	`n = 2026`		
`while (b) { r = a % b; a = b; b = r; }` → `a`	`a = 2026, b = 1066`		
`for (d = 2; d * d <= n; d++) while (n % d == 0) { n /= d; k++; }` + `if (n > 1) k++;`	`n = 2026`		
`v = v * 2 + c;` pentru fiecare cifră `c`	`11111101010`		
`if (x > m1) { m2 = m1; m1 = x; } else if (x > m2) m2 = x;`	`3 8 5 8 2`		

P8. Ordonează crescător, pentru $n = 10^6$: n, $\sqrt{n} \cdot \log_2 n$, $n^2 / 1000$, 2^{20}, $n \cdot \log_2 n$, $500 \cdot \sqrt{n}$. ($\log_2 10^6 \approx 20$.)

Nivelul 2 · Găsește și repară

★★ găsește și repară

P9. Programul calculează rapid `1 + 2 + … + n` pentru `n ≤ 10⁹`. Pentru `n = 100 000` afișează un număr greșit. De ce? De la ce `n` începe să greșească? Repară.

```
int n;  cin >> n;
long long s = n * (n + 1) / 2;
```

P10. *Numărul dublat.* Se citesc `n`, apoi `n + 1` numere: toate numerele de la 1 la `n`, unul dintre ele de două ori. Elevul propune: „pentru fiecare `k` de la 1 la `n` parcurg din nou datele și număr de câte ori apare `k`". Câte numere citește în total pentru `n = 10⁵`? Rescrie metoda în O(n), cu memorie constantă. Test: `5` / `3 1 4 5 2 4` → `4`.

P11. Programul calculează `S = 1 + (1 + 2) + (1 + 2 + 3) + … + (1 + 2 + … + n)`. Câte adunări face pentru `n` dat? Rescrie-l cu **o singură** buclă, apoi fără nicio buclă (indiciu: `n(n + 1)(n + 2)/6`). Test: `n = 4 → 20`, `n = 10 → 220`.

```
long long S = 0;
for (int i = 1; i <= n; i++)
    for (int j = 1; j <= i; j++) S += j;
```

P12. Elevul a vrut să oprească bucla la primul divizor găsit. Ce numere până la 40 declară greșit prime? De ce? Repară păstrând oprirea timpurie.

```
bool prim = (n >= 2);
for (int d = 2; d * d <= n; d++)
    if (n % d == 0) prim = false;
    break;
```

P13. *Rânduri amestecate.* Se citesc `n` numere. Programul afișează câte perechi `(i < j)` au suma **pară**, în O(n), fără vectori (pentru `5` / `1 2 3 4 5` afișează `4`). Ordonează rândurile; unul este în plus. De ce funcționează formula?

```
A)  cin >> x;
B)  for (int j = i + 1; j <= n; j++)
C)  int n, x; long long p = 0, q = 0;
D)  }
```

```
E)  cout << p * (p - 1) / 2 + q * (q - 1) / 2;
F)  for (int i = 1; i <= n; i++) {
G)  cin >> n;
H)  if (x % 2 == 0) p++; else q++;
```

P14. *Descrierea de la III.3 a).* Fișierul are cel mult 10^6 numere naturale. Se cere cel mai mare număr par și de câte ori apare. Un elev scrie: „Citesc toate numerele într-un vector, îl sortez descrescător, caut primul număr par și număr de câte ori apare. Este eficient pentru că sortarea e rapidă." De ce nu primește punctajul pentru eficiență? Scrie tu descrierea.

Nivelul 3 · Scrie cod

★★ scrie cod

Pentru fiecare program, verifică **toate** testele date înainte să spui că e gata. Fiecare trebuie să meargă instant la restricțiile maxime.

P15. Suma pe interval. Afișează suma tuturor întregilor din `[a, b]`, cu `−10⁹ ≤ a ≤ b ≤ 10⁹`, fără buclă.

intrare	`1 100`	`-5 5`	`3 3`	`10 20`	`1 1000000000`	`-1000000000 1000000000`
ieșire	`5050`	`0`	`3`	`165`	`500000000500000000`	`0`

P16. Ultimele trei cifre. Afișează ultimele trei cifre ale lui `aⁿ` (cu zerouri în față), pentru `a ≤ 10⁹`, `n ≤ 10⁹`. Folosește ideea de la P4 și ține doar restul la 1000 după fiecare înmulțire.

intrare	`2 10`	`7 4`	`3 0`	`2 1000000000`	`999 999999999`
ieșire	`024`	`401`	`001`	`376`	`999`

P17. Pătratele din interval. Câte pătrate perfecte sunt în `[a, b]`, cu `1 ≤ a ≤ b ≤ 10¹²`? Atenție la rotunjirile lui `sqrt` (corectează rezultatul cu `while`).

intrare	`1 100`	`10 20`	`17 24`	`1 1000000000000`	`999999999999 1000000000000`
ieșire	`10`	`1`	`0`	`1000000`	`1`

P18. Turneul. `n` echipe joacă fiecare cu fiecare o dată; pe `k` terenuri se joacă în paralel câte un meci pe oră. Afișează numărul de meciuri și numărul minim de ore. Teste: `4 1 → 6 6`, `20 3 → 190 64`, `1 5 → 0 0`, `100000 7 → 4999950000 714278572`.

P19. Recapitulare: numărul secret. Se citesc numere naturale terminate cu 0. Adună cifrele tuturor numerelor **prime** citite și afișează suma în baza 2. Teste: `13 4 29 7 0 → 10110`, `4 6 8 0 → 0`, `2 0 → 10`, `1000003 1 0 → 100`.

P20. Recapitulare: vecini prieteni. Se citesc `n` și `n` numere naturale nenule. Afișează lungimea celei mai lungi secvențe în care oricare două numere **vecine** sunt prime între ele.

intrare	`6 / 4 9 15 8 3 6`	`3 / 2 4 6`	`4 / 1 1 1 1`	`1 / 7`
ieșire	`3`	`1`	`4`	`1`

Nivelul 4 · Pune mintea la contribuție

★★★ pune mintea la contribuție

P21. Becurile. Pe un hol sunt `n` becuri stinse. Trec pe rând `n` persoane: persoana `k` apasă întrerupătorul fiecărui bec cu numărul multiplu de `k`. Câte becuri rămân aprinse? Simulează pentru `n = 10`, apoi găsește o formulă O(1) și **explică** de ce e corectă (indiciu: de câte ori este apăsat becul 12? dar becul 16?). Ce răspuns dai pentru `n = 10¹²`?

P22. Problema inversă. Pentru ce valori ale lui `n` bucla `for (d = 1; d * d <= n; d++)` se execută de exact 1000 de ori? Pentru ce `n` a doua secvență din P1 dă `cnt = 4950`?

P23. Două numere lipsă. Se citesc `n` și `n − 2` numere distincte din `[1, n]`; lipsesc două. Găsește-le în O(n), cu memorie constantă. Indiciu: din sumă afli `a + b`, iar din suma pătratelor `a² + b²`. Teste: `5` / `1 4 5` → `2 3`; `6` / `6 1 3 5` → `2 4`; `7` / `2 3 4 5 6` → `1 7`.

P24. Optimizare în trepte. Câte numere $\leq N$ au exact 3 divizori? Varianta 1: numeri divizorii fiecărui număr. Varianta 2: folosești faptul că astfel de numere sunt p^2, cu p prim. Estimează pașii fiecărei variante pentru $N = 10^{12}$ și rulează varianta 2 pentru $N = 10^6$ și $N = 10^{12}$.

P25. Alege metoda. Pentru fiecare cerință, spune ce tipar din clasa a IX-a folosești și ce complexitate obții (fără cod): a) dacă un număr de 18 cifre este palindrom; b) cea mai lungă perioadă de zile cu temperatura în creștere, din 10^7 valori; c) câte numere din $[1, n]$ sunt prime cu n, pentru $n \leq 10^{12}$; d) ultima cifră a lui f_n (Fibonacci), pentru $n \leq 10^7$; e) cele mai mari două valori distincte dintr-un fișier uriaș; f) ora $hh:mm$ după m minute de la miezul nopții.

Probleme de lucru în clasă (pbinfo)

Le rezolvăm împreună în ședință, în ordinea de mai jos.

- [ ] **#1360** **suma-gauss** — ușoară
  https://www.pbinfo.ro/probleme/1360/suma-gauss
- [ ] **#1255** **lipsa** — ușoară
  https://www.pbinfo.ro/probleme/1255/lipsa
- [ ] **#3977** **expon** — ușoară
  https://www.pbinfo.ro/probleme/3977/expon

Temă pentru acasă (pbinfo)

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

- [ ] **#1284** **carte1** — ușoară
  https://www.pbinfo.ro/probleme/1284/carte1
- [ ] **#482** **auto** — ușoară
  https://www.pbinfo.ro/probleme/482/auto
- [ ] **#3072** **sumprodmax3** — ușoară
  https://www.pbinfo.ro/probleme/3072/sumprodmax3
- [ ] **#4493** **minnm** — medie
  https://www.pbinfo.ro/probleme/4493/minnm
- [ ] **#1798** **trepte** — medie
  https://www.pbinfo.ro/probleme/1798/trepte