FIȘĂ DE LUCRU · CLASA A X-A · FIȘA 5 · ACOPERĂ LECȚIA X.6

Vectori de frecvență și ciurul lui Eratostene

Durată: 90 min Nivel: toate profilurile

CE VEI ȘTI DUPĂ LECȚIE

- Folosești valoarea ca indice: numeri, marchezi și sortezi valori mici fără comparații.
- Construiești ciurul lui Eratostene și îl folosești pentru multe întrebări rapide.
- Recunoști capcanele: vector prea mic, n = 0, egalitate la maxim, oprirea ciurului.

Ce trebuie să știi (o pagină)

Ce vrei	Cod	Observație
de câte ori apare fiecare valoare	`fr[x]++;`	valori în `[0, V]` → `int fr[V + 1]`, global
doar „a apărut sau nu”	`ap[x] = true;`	vector caracteristic, `bool`
valorile distincte, crescător	`for (v = 0; v <= V; v++) if (fr[v]) cout << v;`	fără nicio sortare
toate valorile, crescător	`for (k = 1; k <= fr[v]; k++) cout << v;` în bucla de mai sus	sortare prin numărare, O(n + V)
valoarea cea mai frecventă	`if (fr[v] > fr[best]) best = v;`	`>` strict → cea mai mică la egalitate
cifrele lui `n`	`while (n > 0) { fr[n % 10]++; n /= 10; }`	pentru `n = 0` nu numără nimic!
valori negative în `[-50, 50]`	`fr[x + 50]++;`	„deplasare”; la afișare scazi 50
ciurul până la `N`	`for (i = 2; i * i <= N; i++) if (!c[i]) for (j = i * i; j <= N; j += i) c[j] = true;`	`c[x] == false` → `x` prim
nu uita	`c[0] = c[1] = true;`	0 și 1 nu sunt prime

ATENȚIE — GREȘELI FRECVENTE

Vectorul de frecvență merge doar pentru valori **mici** (până la câteva milioane). Pentru valori până la 10^9, `fr[1000000001]` ar ocupa 4 GB: atunci sortezi.

Nivelul 1 · Prezice

★ citește și prezice

P1. Se citesc, pe rând, numerele `5 2 8 2 5 5 0`, cu `fr[x]++`. Completează tabelul, apoi spune ce afișează fiecare secvență (toate cu `v` de la 0 la 9).

v	0	1	2	3	4	5	6	7	8	9
fr[v]										

secvența (în bucla cu `v`)	ce afișează
A) `if (fr[v] > 0) cout << v << ' ';`	
B) `for (k = 1; k <= fr[v]; k++) cout << v << ' ';`	
C) `if (fr[v] == 1) cout << v << ' ';`	

P2. Cât este `fr[0]` după secvența de mai jos, pentru `n = 1020300`? Dar pentru `n = 0`? Este corect al doilea rezultat?

```
int fr[10] = {0};
while (n > 0) { fr[n % 10]++; n /= 10; }
```

P3. Pseudocod. `f` are inițial toate elementele 0.

```
citește n (număr natural)
┌cât timp n ≠ 0 execută
│ f[n%10] ← f[n%10] + 1
│ n ← [n/10]
└■
m ← 0
┌pentru c ← 9,0,-1 execută
│┌pentru k ← 1,f[c] execută
│| m ← m*10 + c
│└■
└■
scrie m
```

a) Ce se afișează pentru `n = 30712`? **b)** Scrie toate valorile de 4 cifre pentru care se afișează `9900`. **c)** Ce se afișează pentru `n = 30712` dacă a doua buclă devine `pentru c ← 0,9`? De ce „dispare" o cifră?

P4. Rulăm ciurul pentru `N = 50`. Completează: pentru ce valori ale lui `i` se intră în bucla cu `j` și câte atribuiri `c[j] = true` face fiecare? Câte atribuiri sunt în total? Câte numere compuse sunt între 2 și 50? De ce cele două numere diferă?

i				
atribuiri				

P5. Ce se întâmplă dacă... În ciur, elevul pornește `j` de la `i` în loc de `i * i`. Ce numere „prime" afișează programul pentru `N = 30`? Care prime au dispărut și de ce tocmai acestea?

P6. Ce afișează pentru intrarea `5` / `3 8 1 8 3`? Spune în cuvinte ce face.

```cpp
bool ap[101] = {false};
cin >> n;
for (int i = 1; i <= n; i++)
{
    cin >> x;
    if (ap[x]) { cout << x << ' ' << i; break; }
    ap[x] = true;
}
```

P7. Pseudocod. `f` are inițial toate elementele 0.

```
citește n
nr ← 0
┌pentru i ← 1,n execută
│ citește x
│┌dacă f[x] = 0 atunci nr ← nr + 1
│└■
│ f[x] ← f[x] + 1
└■
scrie nr
```

a) Ce se afișează pentru `n = 6` și `4 9 4 4 1 9`? **b)** Ce calculează algoritmul, în cuvinte? **c)** Pentru `n = 5`, dă un șir pentru care se afișează `1` și unul pentru care se afișează `5`.

P8. Valorile citite sunt în `[-50, 50]`. Ce afișează programul pentru `-3 2 -3 0 2 -3`?

```cpp
int fr[101] = {0};
// pentru fiecare x citit:  fr[x + 50]++;
for (int v = 0; v <= 100; v++)
    if (fr[v]) cout << v - 50 << ':' << fr[v] << ' ';
```

Nivelul 2 · Găsește și repară

★★ găsește și repară

P9. Se citesc notele a 30 de elevi (de la 1 la 10) și se numără câte note de fiecare fel există. Programul se comportă ciudat doar când cineva ia nota 10. De ce? Repară.

```cpp
int fr[10];   // global
// pentru fiecare notă citită:  fr[nota]++;
```

P10. Programul trebuie să afișeze valoarea cea mai frecventă, iar **la egalitate pe cea mai mică**. Ce afișează pentru `2 5 2 5`? Repară un singur caracter.

```cpp
int best = 0;
for (int v = 1; v <= 1000; v++)
    if (fr[v] >= fr[best]) best = v;
```

P11. „Au `a` și `b` o cifră comună?" Elevul folosește un singur vector pentru ambele numere și spune DA dacă vreo cifră are frecvența cel puțin 2. Ce răspunde pentru `122` și `345`? Dar pentru `11` și `22`? Repară ideea (nu doar codul).

```
while (a > 0) { fr[a % 10]++; a /= 10; }
while (b > 0) { fr[b % 10]++; b /= 10; }
for (int c = 0; c <= 9; c++)
    if (fr[c] >= 2) comun = true;
```

P12. *Puzzle cu rânduri amestecate.* Pune rândurile în ordine ca programul să afișeze toate numerele prime ≤ N (`N` deja citit). Un rând este în plus.

```
A)  if (!ciur[i])
B)  for (int j = i * i; j <= N; j += i)
C)  ciur[0] = ciur[1] = true;
D)  for (int i = 2; i * i <= N; i++)
E)  ciur[j] = true;
F)  for (int j = i; j <= N; j += i)
G)  for (int x = 2; x <= N; x++) if (!ciur[x]) cout << x << ' ';
```

P13. Completează golurile ca programul să afișeze valorile (≤ 1000) care apar **exact o dată**, în ordine **descrescătoare**. Test: `4 9 4 1 7 9 0` → `7 1 0`.

```
for (int v = ______; v ______; v--)
    if (_______________)
        cout << v << ' ';
```

P14. Elevul a scris ciurul cu `for (int i = 2; i * i < N; i++)`. Pentru `N = 49`, programul afișează și 49 printre numerele prime. De ce? Pentru ce valori ale lui `N` din `[2, 100]` greșește programul?

Nivelul 3 · Scrie cod

★★ scrie cod

P15. Zarul. Un zar a fost aruncat de `n` ori. Afișează pe prima linie fața care a ieșit cel mai des (cea mai mică, la egalitate), iar pe a doua fețele care nu au ieșit deloc, sau `toate` dacă au ieșit toate.

intrare	ieșire
`8` / `3 6 3 1 6 3 2 2`	`3` / `4 5`
`6` / `1 2 3 4 5 6`	`1` / `toate`
`3` / `5 5 5`	`5` / `1 2 3 4 6`

P16. Cifrele care lipsesc. Se citește `n` (`long long`). Afișează cifrele care **nu** apar în `n`, crescător, sau `complet` dacă apar toate cele 10. Atenție la ultimul test!

intrare	ieșire
`1234567890`	`complet`
`7007`	`1 2 3 4 5 6 8 9`
`0`	`1 2 3 4 5 6 7 8 9`

P17. Prime gemene. Afișează toate perechile `p, p + 2` de numere prime cu `p + 2 ≤ N`, câte una pe linie, sau `nu exista`. `N ≤ 10⁶`.

intrare	ieșire
`20`	`3 5` / `5 7` / `11 13` / `17 19`
`7`	`3 5` / `5 7`
`4`	`nu exista`

P18. Magazinul de pantofi. Se citesc mărimile (35–46) a `n` perechi vândute. Afișează, crescător, fiecare mărime vândută și câte o steluță pentru fiecare pereche (o histogramă).

intrare	ieșire
`6` / `38 40 38 42 40 38`	`38: ***` / `40: **` / `42: *`
`1` / `46`	`46: *`

P19. Cel mai apropiat prim. Se dau $q \le 10^5$ numere x, cu $1 \le x \le 10^6$. Pentru fiecare, afișează cel mai mic număr prim $\ge x$. Construiește ciurul **o singură dată**. Gândește-te până unde trebuie să meargă!

intrare	ieșire
`4` / `14 2 1 1000000`	`17 2 2 1000003`

P20. Alegerile din clasă. Sunt $k \le 1000$ candidați, numerotați de la 1, și n voturi. Afișează câștigătorul sau `balotaj` dacă mai mulți candidați au numărul maxim de voturi.

intrare	ieșire
`5 3` / `1 2 2 3 2`	`2`
`4 3` / `1 2 1 2`	`balotaj`
`1 5` / `4`	`4`

Nivelul 4 · Pune mintea la contribuție

★★★ pune mintea la contribuție

P21. Detectivul invers. Algoritmul din P3 a afișat `99510`. Câte valori de **5 cifre** putea avea `n`? Calculează fără să le scrii pe toate.

P22. Numărul lipsă, fără vector. Se citesc `n - 1` numere distincte din `{1, 2, …, n}`; unul lipsește. Rezolvă întâi cu un vector caracteristic. Apoi rezolvă cu **memorie O(1)** (fără niciun vector). Test: `n = 5`, `3 1 5 2` → `4`. Ce tip de date îți trebuie pentru $n = 10^6$?

P23. Ciurul care descompune. Modifică ciurul ca să rețină în `cmf[x]` cel mai mic factor prim al lui `x` (pentru numerele compuse; un număr prim rămâne cu 0): când tai `j`, scrii `cmf[j] = i` doar dacă `cmf[j]` este încă 0. **a)** Descompune 360 folosind repetat `x = x / cmf[x]`. **b)** De ce primul `i` care „lovește" un număr compus este chiar cel mai mic factor prim al lui, deși `j` pornește de la `i * i`?

P24. Contraexemplu. Elevul afirmă: „Două numere de 3 cifre au aceleași cifre dacă au aceeași **sumă** a cifrelor și același **produs** al cifrelor." Găsește un contraexemplu cu o cifră 0, apoi unul **fără** cifra 0. De ce vectorul de frecvență nu poate fi „păcălit"?

P25. Mini-proiect: prime pe interval. Se dau 10^5 întrebări (a, b), cu $1 \le a \le b \le 10^6$: câte numere prime sunt în $[a, b]$? Parcurgerea intervalului la fiecare întrebare e prea lentă. Construiește, după ciur, un vector $cnt[x]$ = câte prime sunt ≤ x, astfel încât fiecare răspuns să coste un singur calcul. Verifică: $[10, 30]$ → 6, $[1, 100]$ → 25.

Probleme de lucru în clasă (pbinfo)

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

- [ ] **#244 cifreord** — ușoară
  https://www.pbinfo.ro/probleme/244/cifreord
- [ ] **#267 unice** — ușoară
  https://www.pbinfo.ro/probleme/267/unice
- [ ] **#301 frecventa1** — medie
  https://www.pbinfo.ro/probleme/301/frecventa1
- [ ] **#303 eratostene** — medie
  https://www.pbinfo.ro/probleme/303/eratostene

Temă pentru acasă (pbinfo)

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

- [ ] **#187 ciffrecv** — ușoară
  https://www.pbinfo.ro/probleme/187/ciffrecv
- [ ] **#239 nrlipsa** — medie
  https://www.pbinfo.ro/probleme/239/nrlipsa
- [ ] **#302 cautaprim** — medie
  https://www.pbinfo.ro/probleme/302/cautaprim
- [ ] **#529 cifre4** — medie
  https://www.pbinfo.ro/probleme/529/cifre4