FIȘĂ DE LUCRU · CLASA A X-A · FIȘA 3 · ACOPERĂ LECȚIA X.4

Sortări

Durată: 90 min Nivel: toate profilurile

CE VEI ȘTI DUPĂ LECȚIE

- Urmărești pas cu pas bulele, selecția și inserția și numeri treceri, comparații, interschimbări.
- Alegi metoda potrivită pentru date aproape sortate, valori mici sau vectori uriași.
- Sortezi după un criteriu, ții împreună doi vectori paralели și sortezi doar o parte.

Ce trebuie să știi (o pagină)

Metoda	Inima codului	Ce e sigur după pasul `k`
bule (cu indicator)	`if (a[i] > a[i+1]) { swap(a[i], a[i+1]); sortat = false; }` repetat cât timp `!sortat`	ultimele `k` elemente sunt la locul lor
selecție (clasică)	`pmin = i;` caut `a[j] < a[pmin]`, apoi un singur `swap(a[i], a[pmin])`	primele `k` elemente sunt cele mai mici, în ordine
inserție	`x = a[i];` cât timp `a[j] > x` mut `a[j]` la dreapta; `a[j+1] = x;`	`a[1..k+1]` e sortat (dar se mai poate schimba)
criteriu propriu	schimbi **doar** condiția din `if`	descrescător: `a[i] < a[i+1]`
doi vectori paralели	interschimbi **ambii** vectori în același `if`	nota rămâne lângă numele ei
STL	`sort(a + 1, a + n + 1);` / `sort(a + 1, a + n + 1, greater<int>());`	O(n log n)

Metoda	Comparații	Interschimbări / mutări	Bună când
bule	până la ~n^2	= numărul de perechi inversate	vectorul e aproape sortat (cu indicator)
selecție	mereu $n(n-1)/2$	cel mult $n - 1$	interschimbarea e scumpă
inserție	până la ~$n^2/2$	= numărul de perechi inversate	vectorul e aproape sortat

ATENȚIE — GREȘELI FRECVENTE

O sortare e **stabilă** dacă elementele egale își păstrează ordinea. Bulele și inserția sunt stabile, selecția cu interschimbări nu. Contează când sortezi elevi după notă și vrei ca, la egalitate, să rămână în ordinea din catalog.

Nivelul 1 · Prezice

★ citește și prezice

P1. Metoda bulelor cu indicator pe `(4, 9, 2, 7, 1)`. Completează tabelul. Câte treceri face în total? Câte interschimbări?

trecere	vectorul după trecere	interschimbări
1		
2		
…		

P2. Selecția **clasică** (cu `pmin`, o singură interschimbare pe pas) pe `(7, 3, 9, 3, 1)`. Scrie vectorul după fiecare valoare a lui `i`. Câte interschimbări face? Câte ar face varianta „interschimb de fiecare dată când `a[j] < a[i]`"?

P3. Inserția pe `(3, 8, 5, 1, 6)`. Pentru fiecare `i` de la 2 la 5, scrie vectorul după inserare și câte elemente au fost **mutate la dreapta**. Cât fac mutările în total?

P4. Ce afișează? Spune **în cuvinte** după ce criteriu sortează. De ce `42` rămâne înaintea lui `22`?

```
int a[] = {0, 25, 13, 42, 31, 17, 22}, n = 6;
bool s;
do {
    s = true;
    for (int i = 1; i < n; i++)
        if (a[i] % 10 > a[i + 1] % 10) { swap(a[i], a[i + 1]); s = false; }
} while (!s);
for (int i = 1; i <= n; i++) cout << a[i] << ' ';
```

P5. Iepurele și broasca țestoasă. Adaugă un contor `cmp++` înaintea fiecărui `if` din bulele cu indicator. Completează pentru fiecare vector:

vector	treceri	comparații
`(1, 2, 3, 4, 5)`		
`(2, 3, 4, 5, 1)`		
`(5, 1, 2, 3, 4)`		

Ultimii doi vectori sunt „aproape sortați" la fel de mult. De ce costă atât de diferit?

P6. Vectorul global `a[1..6] = (9, 6, 1, 8, 3, 5)`, deci `a[0] = 0`. Ce conține `a[1..6]` după fiecare instrucțiune, executate separat pe vectorul inițial? Pentru b), cât este `a[3]`?

a) `sort(a + 2, a + 5);` **b)** `sort(a + 1, a + 7, greater<int>());` **c)** `sort(a, a + 6);`

P7. Ce afișează pentru `a = (4, 7, 2, 9, 6, 1)` ? Ce face secvența, în cuvinte? Elementele de același fel își păstrează ordinea?

```
for (int i = 1; i < n; i++)
    for (int j = i + 1; j <= n; j++)
        if (a[i] % 2 > a[j] % 2) swap(a[i], a[j]);
```

P8. Catalogul. Cinci elevi, în ordinea din catalog: `A 8, B 9, C 8, D 7, E 9`. Se sortează crescător după notă, cu selecția „cu interschimb de fiecare dată" (interschimbi atât nota, cât și litera). În ce ordine apar literele? Dar cu bulele? Care dintre rezultate păstrează ordinea din catalog la note egale?

Nivelul 2 · Găsește și repară

★★ găsește și repară

P9. Vectorul `int a[101]` este global, `n = 4`, `a = (5, 1, 4, 2)`. Ce afișează programul? Unde a ajuns `5` ? Repară.

```
do {
    sortat = true;
    for (int i = 1; i <= n; i++)
        if (a[i] > a[i + 1]) { swap(a[i], a[i + 1]); sortat = false; }
} while (!sortat);
for (int i = 1; i <= n; i++) cout << a[i] << ' ';
```

P10. Inserție „fără variabilă în plus". Ce afișează pentru `(5, 2, 4)` ? Dar pentru `(1, 3, 2)` ? Explică ce se pierde și repară.

```
for (int i = 2; i <= n; i++)
{
    int j = i - 1;
    while (j >= 1 && a[j] > a[i]) { a[j + 1] = a[j]; j--; }
    a[j + 1] = a[i];
}
```

P11. *Puzzle cu rânduri amestecate.* Bule **descrescătoare**, care la fiecare trecere scurtează zona verificată. Pune rândurile în ordine; un rând este în plus. De ce e corect `m--` ?

```
A)  } while (!sortat);
B)  int m = n;  bool sortat;
C)      m--;
D)  do {
E)      sortat = true;
F)      for (int i = 1; i < m; i++)
G)          if (a[i] < a[i + 1]) { swap(a[i], a[i + 1]); sortat = false; }
H)      for (int i = 1; i <= m; i++)
```

P12. Completează condiția ca bulele să sorteze **crescător după ultima cifră**, iar la aceeași ultimă cifră **descrescător după valoare**. Test: `(34, 12, 54, 7, 92, 17)` → `92 12 54 34 17 7`.

```
if (_____________________________________________)
    swap(a[i], a[i + 1]);
```

P13. Trebuie sortate crescător **doar numerele pare**, iar cele impare rămân pe loc. Pentru `(8, 3, 2, 6, 4)` corect este `2 3 4 6 8`. Elevul a scris bulele cu condiția de mai jos și obține altceva. Ce obține și de ce? Repară schimbând bucla, nu condiția.

```
if (a[i] % 2 == 0 && a[i + 1] % 2 == 0 && a[i] > a[i + 1]) ...
```

P14. Ce greșeală a făcut elevul? Selecția lui afișează `4 1 3 5` pentru `(5, 3, 1, 4)`. Urmărește pașii și găsește caracterul greșit.

```
for (int i = 1; i < n; i++)
{
    int pm = i;
    for (int j = i + 1; j <= n; j++)
        if (a[j] < a[i]) pm = j;
    swap(a[i], a[pm]);
}
```

Nivelul 3 · Scrie cod

★★ scrie cod

La P15–P19 poți folosi `sort`, cu excepția celor în care îți cer altfel. Verifică **toate** testele.

P15. Coșul de cumpărături. Ai bugetul `B` și prețurile a `n` produse. Câte produse poți cumpăra, cel mult? (Indiciu: ce cumperi întâi?)

intrare	ieșire
`5 100` / `40 15 70 25 30`	`3`
`3 10` / `20 30 40`	`0`
`4 100` / `25 25 25 25`	`4`

P16. Clasamentul la cros. Concurentul `i` a alergat în `t[i]` secunde. Afișează numerele concurenților în ordinea clasamentului (timp crescător; la timp egal, numărul mai mic întâi). Scrie sortarea de mână, cu doi vectori paraleli.

intrare	ieșire
`5` / `62 58 70 58 61`	`2 4 5 1 3`
`3` / `10 10 10`	`1 2 3`
`1` / `99`	`1`

P17. Mediana. Afișează mediana a `n` temperaturi, cu o zecimală: elementul din mijloc după sortare sau, dacă `n` e par, media celor două din mijloc.

intrare	ieșire
`5` / `7 1 9 3 5`	`5.0`
`4` / `10 2 8 4`	`6.0`
`2` / `1 2`	`1.5`

P18. Raftul cu goluri. Sortează crescător doar elementele de pe pozițiile **impare** (1, 3, 5, ...); cele de pe poziții pare rămân pe loc. Fără vector auxiliar.

intrare	ieșire
`7` / `9 1 4 2 8 3 1`	`1 1 4 2 8 3 9`
`2` / `5 0`	`5 0`
`3` / `3 7 1`	`1 7 3`

P19. Gemenii de la baschet. Se dau înălțimile a `n ≥ 2` jucători. Afișează cea mai mică diferență dintre doi jucători și cele două înălțimi (crescător; prima pereche, în ordinea sortată, la egalitate). Fă-o în O(n log n), nu cu toate perechile.

intrare	ieșire
`6` / `190 175 201 182 196 186`	`4 182 186`
`2` / `180 180`	`0 180 180`
`3` / `150 200 170`	`20 150 170`

P20. Cel mai mic număr. Se citește `n ≥ 1` (până la 18 cifre). Afișează cel mai mic număr care are exact aceleași cifre, **fără zero în față**.

intrare	ieșire
`30210`	`10023`
`5`	`5`
`1000`	`1000`
`987`	`789`

Nivelul 4 · Pune mintea la contribuție

★★★ pune mintea la contribuție

P21. Problema inversă. Un vector cu valorile `1, 2, 3, 4, 5` (fiecare o dată) a devenit `(2, 1, 4, 3, 5)` după **prima** trecere a bulelor. Găsește toți vectorii inițiali posibili. Câți sunt? Unde putea fi `5` la început?

P22. Perechile inversate. O pereche de poziții `i < j` cu `a[i] > a[j]` se numește *inversiune*. Câte inversiuni are `(3, 1, 4, 2)`? Câte interschimbări fac bulele pe el? Câte mutări face inserția? Explică de ce cele trei numere sunt mereu egale. Câte interschimbări fac bulele, cel mult, pe un vector cu 6 elemente?

P23. Doar 0 și 1. Vectorul conține doar `0` și `1`. Sortează-l **într-o singură parcurgere**, fără vector de frecvență și fără să numeri, doar cu interschimbări (indiciu: un indice pleacă din stânga, unul din dreapta). Test: `(1, 0, 1, 1, 0, 0, 1)` → `0 0 0 1 1 1 1`. Ce schimbi dacă apare și valoarea `2`?

P24. Adevărat sau fals? a) „După prima trecere a bulelor, ultimul element este maximul." **b)** „Dacă la prima trecere a bulelor primul element nu s-a mutat, atunci el este minimul." Pentru cea falsă, dă un contraexemplu cu 3 elemente; pentru cea adevărată, o explicație.

P25. Alege unealta. Pentru fiecare situație, alege metoda și justifică în 1–2 propoziții: **a)** vectorul `(2, 3, …, 1000)` sortat, la care s-a adăugat la final `1`; compară inserția și bulele (câte comparații face fiecare, aproximativ?); **b)** 10^6 note de la 1 la 10; **c)** același vector ca la a), dar cu `1000` mutat la început, `(1000, 1, 2, …, 999)`: mai sunt bulele lente?

Probleme de lucru în clasă (pbinfo)

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

- [ ] `#509` **ordonare** — ușoară
  https://www.pbinfo.ro/probleme/509/ordonare
- [ ] `#129` **sortare** — ușoară
  https://www.pbinfo.ro/probleme/129/sortare
- [ ] `#510` **sortprime** — ușoară
  https://www.pbinfo.ro/probleme/510/sortprime
- [ ] `#130` **sortcif** — ușoară
  https://www.pbinfo.ro/probleme/130/sortcif

Temă pentru acasă (pbinfo)

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

- [ ] `#511` **ksort** — ușoară
  https://www.pbinfo.ro/probleme/511/ksort
- [ ] `#4146` **sort10** — ușoară
  https://www.pbinfo.ro/probleme/4146/sort10
- [ ] `#183` **sortpp** — medie
  https://www.pbinfo.ro/probleme/183/sortpp
- [ ] `#515` **sortsd** — medie
  https://www.pbinfo.ro/probleme/515/sortsd