FIȘĂ DE LUCRU · CLASA A IX-A · FIȘA 7 · ACOPERĂ LECȚIILE IX.12–IX.13

Șiruri recurente, secvențe și baze de numerație

Durată: 100 min Nivel: toate profilurile

CE VEI ȘTI DUPĂ LECȚIE

- Generezi termenii unui șir recurent cu o fereastră de 2–3 variabile, în ordinea corectă a atribuirilor.
- Prelucrezi secvențe citite comparând cu elementul anterior: platouri, monotonie, recorduri.
- Treci un număr dintr-o bază în alta (împărțiri repetate, Horner, grupe de biți) și lucrezi cu cifrele binare.

Ce trebuie să știi (o pagină)

Ce vrei	Cod / regulă	Reține
recurență cu doi termeni	`c = a + b; a = b; b = c;`	ordinea contează: întâi `a = b`
termenii până la o limită	`while (a <= n) { …a…; c = a + b; a = b; b = c; }`	nu știi câți sunt → `while`
Fibonacci mare	f_{47} depășește `int`, f_{93} depășește `long long`	pentru ultima cifră: `% 10` la fiecare pas
secvență (platou, crescătoare…)	`if (cond(ant, x)) lung++; else lung = 1; lmax = max(lmax, lung); ant = x;`	`lmax` se actualizează la **fiecare** pas
fereastră de trei	`if (b > a && b > c) …; a = b; b = c;`	vârfuri, văi
baza 10 → baza b	resturile lui `n % b`, citite invers: `rez += n % b * p; p *= 10; n /= b;`	merge doar pentru b ≤ 10 și numere mici
baza b → baza 10	Horner: `val = val * b + c;` (cifrele de la stânga)	1101_2 → 1, 3, 6, 13
cifră validă în baza b	`0 ≤ c < b`	în baza 8 nu există cifra 8
2 ↔ 8 ↔ 16	grupe de 3, respectiv 4 biți, de la dreapta	$1010\ 1111_2 = AF_{16}$
cifre hexa	10 = A, …, 15 = F	`"0123456789ABCDEF"[r]`
biții lui n	`n % 2` ultimul bit, `n / 2` îl taie; `1 << k` = 2^k	`n` are k biți ⇔ $2^{k-1} \le n < 2^k$

SFAT

La orice recurență scrisă cu atribuiri (ca la BAC), fă **tabelul de valori** pe primele 3–4 iterații înainte să tragi o concluzie. Două rânduri inversate schimbă tot șirul.

Nivelul 1 · Prezice

★ citește și prezice

P1. Scrie primii 8 termeni ai fiecărui șir. Ce observi la al treilea și la al patrulea?

definiția	primii 8 termeni
$x_1 = 2,\ x_2 = 1,\ x_n = x_{n-1} + x_{n-2}$	
$x_1 = 3,\ x_2 = 5,\ x_n = 2 \cdot x_{n-1} - x_{n-2}$	
$x_1 = 1,\ x_2 = 2,\ x_n = x_{n-1} - x_{n-2}$	
$x_1 = 1,\ x_2 = 2,\ x_n = x_{n-1} \cdot x_{n-2}$	

P2. *Ca la BAC.*

```
citește n                    (număr natural nenul)
a ← 1
b ← 1
k ← 0
┌cât timp b ≤ n execută
│   c ← a + b
│   a ← b
│   b ← c
│   k ← k + 1
└■
scrie a, " ", k
```

a. Ce se afișează pentru `n = 100`? Dar pentru `n = 12`? Ce reprezintă `a`?

b. Scrie **toate** valorile lui `n` pentru care prima valoare afișată este `13`.

P3. Pentru șirul `3 3 5 7 7 7 2 4 6 9 1`, care este lungimea celei mai lungi secvențe de elemente: a) egale; b) strict crescătoare; c) pare?

P4. *Grilă.* Fiecare secvență pornește de la `a = 1, b = 1` și se repetă de 6 ori: `cout << a << ' ';` urmat de atribuirile din tabel. Care afișează `1 1 2 3 5 8`? Ce afișează celelalte?

V1	V2	V3	V4	V5
`c = a + b; a = b; b = c;`	`b = a + b; a = b - a;`	`a = b; b = a + b;`	`a = a + b; b = a - b;`	`b = a + b; a = b;`

P5. Completează tabelul.

în baza 10	baza 2	baza 3	baza 8	baza 16
77				
2026				

P6. Cu Horner, scrie valorile succesive ale lui `val` când citești, de la stânga, cifrele lui $2102_{(3)}$, apoi pe cele ale lui $666_{(7)}$.

P7. *Ca la BAC.*

```
citește n                    (număr natural)
p ← 1
m ← 0
┌cât timp n > 0 execută
│   ┌dacă n % 2 = 0 atunci
│   │   m ← m + p
│   └■
│   p ← p * 2
│   n ← [n / 2]
└■
scrie m
```

a. Ce se afișează pentru `n = 10`? Dar pentru `n = 13` și `n = 32`? Spune în cuvinte ce face (scrie `n` și `m` în baza 2).

b. Scrie toate valorile nenule `n ≤ 100` pentru care se afișează `0`.

P8. Câte cifre are `2026` în bazele 2, 3, 8 și 16? Verifică cu secvența `k = 0; while (n > 0) { n /= b; k++; }`. Ce afișează ea pentru `n = 0`? Este corect?

Nivelul 2 · Găsește și repară

★★ găsește și repară

P9. Programul caută cel mai lung platou, dar pentru `1 2 2 2` afișează `1`. Pentru `5 5 1 2` merge. De ce? Repară mutând un singur rând.

```
cin >> n >> ant;
int lung = 1, lmax = 1;
for (int i = 2; i <= n; i++)
{
    cin >> x;
    if (x == ant) lung++;
    else { if (lung > lmax) lmax = lung; lung = 1; }
    ant = x;
}
cout << lmax;
```

P10. Programul trebuia să afișeze termenii Fibonacci `≤ n`. Ce afișează pentru `n = 10`? Dar pentru `n = 1`? Ce termen lipsește mereu? Repară.

```
long long a = 1, b = 1;
while (b <= n) { cout << a << ' '; long long c = a + b; a = b; b = c; }
```

P11. Elevul a transformat în baza 2 cu `rez = rez * 10 + n % 2;` (ca la oglindit). Ce obține pentru `6`, `13` și `8`? De ce `8` iese atât de prost? Repară.

P12. *Rânduri amestecate.* Programul afișează lungimea celei mai lungi secvențe **strict crescătoare** (pentru `5` / `1 3 2 4 6` afișează `3`). Ordonează rândurile; unul este în plus.

```
A)  ant = x;
B)  int lung = 1, lmax = 1;
C)  else lung = 0;
D)  cin >> n >> ant;
E)  if (lung > lmax) lmax = lung;
F)  else lung = 1;
```

```
G)  int n, x, ant;
H)  cout << lmax;
I)  if (x > ant) lung++;
J)  for (int i = 2; i <= n; i++) {
K)  cin >> x;
L)  }
```

P13. *Completează golurile.* `x` conține cifrele unui număr scris în baza `b` (de exemplu `x = 1201`, `b = 3`). Secvența afișează valoarea lui în baza 10 sau `invalid` dacă `x` are o cifră nepermisă. Teste: `1201 3 → 46`, `129 8 → invalid`, `777 8 → 511`.

```
long long val = 0, p = 1;  bool ok = true;
while (x > 0)
{
    int c = ________;
    if (________) ok = false;
    val = val + ________;
    p = ________;
    x = x / 10;
}
```

P14. Pentru un număr `n < 256`, elevul afișează cele două cifre hexazecimale cu `cout << char('0' + n / 16) << char('0' + n % 16);`. Pentru `200` apare `<8`, iar pentru `255` apare `??`. De unde vin semnele acestea? Ce ar trebui să apară? Repară.

Nivelul 3 · Scrie cod

★★ scrie cod

Pentru fiecare program, verifică **toate** testele date înainte să spui că e gata.

P15. Ultima cifră Fibonacci. Afișează ultima cifră a lui f_n ($f_1 = f_2 = 1$) pentru `n ≤ 10⁶`. Termenii nu încap în niciun tip, dar ultima cifră a unei sume depinde doar de ultimele cifre ale termenilor.

`n`	10	20	1	60	1 000 000
ieșire	`5`	`5`	`1`	`0`	`5`

P16. Seria de victorii. Se citesc `n` și rezultatele a `n` meciuri (`1` = victorie, `0` = înfrângere). Afișează lungimea celei mai lungi serii de victorii și numărul meciului cu care începe (prima, dacă sunt mai multe); dacă nu există victorii, afișează `0`.

intrare	`10` / `1 1 0 1 1 1 0 1 1 1`	`3` / `0 0 0`	`4` / `1 1 1 1`	`5` / `0 1 0 1 1`
ieșire	`3 4`	`0`	`4 1`	`2 4`

P17. Bursa. Se citesc `n` și prețul unei acțiuni în `n` zile. Afișează în câte zile s-a stabilit un **record** (prețul a fost mai mare decât în toate zilele anterioare; prima zi contează) și lungimea celei mai lungi perioade de scădere continuă (zile consecutive cu preț strict mai mic decât în ziua dinainte, numărând și ziua de început).

intrare	`7 / 10 12 11 9 15 14 20`	`4 / 5 4 3 2`	`1 / 7`	`5 / 3 3 3 3 3`
ieșire	`4 3`	`1 4`	`1 1`	`1 1`

P18. Culoarea web. Se citesc componentele `r g b` (0–255). Afișează culoarea în formatul `#RRGGBB`. Teste: `255 136 0 → #FF8800`, `0 0 0 → #000000`, `18 52 86 → #123456`, `171 205 239 → #ABCDEF`.

P19. Ceasul binar. Se citește ora `h m`. Afișează ora pe exact 5 biți și minutul pe exact 6 biți (cu zerouri în față). Teste: `13 45 → 01101 101101`, `0 0 → 00000 000000`, `23 59 → 10111 111011`, `8 1 → 01000 000001`.

P20. Palindrom binar. Pentru `n ≥ 1`, afișează `DA` dacă scrierea lui `n` în baza 2 este palindrom (`NU` altfel) și numărul de biți. Nu construi scrierea binară ca număr zecimal. Teste: `9 → DA 4`, `10 → NU 4`, `1 → DA 1`, `585 → DA 10`, `6 → NU 3`.

Nivelul 4 · Pune mintea la contribuție

★★★ pune mintea la contribuție

P21. Fără să generezi tot. Cât este x_{2026} pentru șirul al treilea din P1 ($x_1 = 1$, $x_2 = 2$, $x_n = x_{n-1} - x_{n-2}$)? Explică de ce nu trebuie calculați toți termenii.

P22. De ce funcționează? Arată că numărul scris `121` este pătrat perfect în **orice** bază `b ≥ 3`. În ce bază valorează `121` exact 49? Dar 100?

P23. Fără baza 10. Transformă $1011011101_{(2)}$ în baza 8 și în baza 16 **doar** grupând biți. Verifică apoi trecând prin baza 10. De ce grupezi de la dreapta, nu de la stânga?

P24. Mini-proiect: suma de Fibonacci. Orice număr natural nenul se poate scrie ca sumă de termeni Fibonacci diferiți (1, 2, 3, 5, 8, ...), alegând mereu cel mai mare termen care încape. Scrie programul și verifică: `100 = 89 + 8 + 3`, `2026 = 1597 + 377 + 34 + 13 + 5`. Observă că nu apar niciodată doi termeni **consecutivi** ai șirului. De ce?

P25. Ce se întâmplă dacă... folosești metoda din lecție (`rez += n % 2 * p; p *= 10;`, cu `long long`) pentru `n = 1 000 000`? Câte cifre binare are 10^6 și cât de mare ar fi `rez`? Care este cel mai mare `n` pentru care metoda mai funcționează? Cum ai afișa corect cifrele binare ale oricărui `int`?

Probleme de lucru în clasă (pbinfo)

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

- [ ] **#423** fibonacci1 — ușoară
  https://www.pbinfo.ro/probleme/423/fibonacci1
- [ ] **#1879** platou2 — ușoară
  https://www.pbinfo.ro/probleme/1879/platou2
- [ ] **#426** bazab — ușoară
  https://www.pbinfo.ro/probleme/426/bazab
- [ ] **#428** transfb — ușoară
  https://www.pbinfo.ro/probleme/428/transfb

Temă pentru acasă (pbinfo)

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

- [ ] #256 **fiboverif** — medie
  https://www.pbinfo.ro/probleme/256/fiboverif
- [ ] #277 **varfuri** — medie
  https://www.pbinfo.ro/probleme/277/varfuri
- [ ] #947 **baza16** — ușoară
  https://www.pbinfo.ro/probleme/947/baza16
- [ ] #427 **bazaminima** — ușoară
  https://www.pbinfo.ro/probleme/427/bazaminima
- [ ] #945 **baze** — ușoară
  https://www.pbinfo.ro/probleme/945/baze