FIȘĂ DE LUCRU · CLASA A XI-A · FIȘA 10 · ACOPERĂ LECȚIILE XI.17–XI.18

Stivă, coadă și liste înlănțuite

Durată: 120 min Nivel: intensiv

CE VEI ȘTI DUPĂ LECȚIE

- Simulezi pe hârtie stive, cozi și liste înlănțuite, inclusiv secvențe de tip BAC cu pointeri.
- Găsești greșelile clasice: vârful greșit, ordinea legăturilor la inserare, pointer transmis fără &.
- Alegi structura potrivită (stivă, coadă, listă) pentru o problemă nouă și o implementezi.

Ce trebuie să știi (o pagină)

Ce	Cu vector	Cu STL / observație
stivă (LIFO): push / pop / vârf / vidă	`st[++vf] = x;` · `vf--;` · `st[vf]` · `vf == 0`	`s.push(x)`, `s.pop()`, `s.top()`, `s.empty()`
coadă (FIFO): push / pop / față / vidă	`q[++ultim] = x;` · `prim++;` · `q[prim]` · `prim > ultim`	`q.push(x)`, `q.pop()`, `q.front()`, `q.empty()`
paranteze corecte	`(` → push; `)` → vârful trebuie să fie `(`, pop	la final stiva **goală**
Lee (BFS pe matrice)	`d = -1` = nevizitat; distanța se pune la **adăugare**	`di = {-1,0,1,0}`, `dj = {0,1,0,-1}`
pointer	`int *p = &x;` · `*p = 10;`	`p->info` = `(*p).info`
alocare / eliberare	`nod *p = new nod;` · `delete p;`	`new int[n]` ↔ `delete[] v`
parcurgere listă	`for (nod *p = prim; p != NULL; p = p->urm)`	
inserare după `p`	`q->urm = p->urm;` **apoi** `p->urm = q;`	invers pierzi restul listei
ștergere după `p`	`q = p->urm; p->urm = q->urm; delete q;`	primul nod: modifici `prim`
funcție care schimbă `prim`	`void f(nod *&prim, …)`	fără `&` lucrezi pe o copie

SFAT

La orice exercițiu cu liste, **desenează**: cutii pentru noduri, săgeți pentru `urm`, iar numele pointerilor sub nodurile pe care arată. Execută instrucțiunile una câte una, mutând săgețile.

Nivelul 1 · Prezice

★ citește și prezice

P1. S → C. În stiva S se introduc, în ordine, 1, 2, 3, 4, 5. Apoi, de 3 ori: scoți vârful din S și îl pui în coada C. Apoi, de 2 ori: scoți primul element din C și îl pui în S. Ce conține S (bază → vârf) și ce conține C (față → spate)?

P2. Ce afișează?

```
stack<int> s; queue<int> q;
for (int i = 1; i <= 6; i++)
    if (i % 2) s.push(i); else q.push(i);
while (!s.empty()) {
    int x = s.top(); s.pop();
    if (x > q.front()) q.push(x);
    cout << q.front() << ' '; q.pop();
}
cout << "| " << q.size();
```

P3. Ce afișează? Câte elemente ies din coadă în total (`pasi`)?

```
queue<int> q; int v[] = {3, 8, 5, 6, 1}, pasi = 0;
for (int i = 0; i < 5; i++) q.push(v[i]);
while (!q.empty()) {
    int x = q.front(); q.pop(); pasi++;
    if (x % 2 == 0) cout << x << ' ';
    else if (x > 1) q.push(x / 2);
}
cout << "| " << pasi;
```

P4. Paranteze de trei feluri. (a) Scrie conținutul stivei după fiecare caracter pentru `(([]){})` . Cât este înălțimea maximă? (b) Este corect `({[]})[(` ? Ce rămâne în stivă? (c) La al câtelea caracter își dă seama programul că `{[}]` este greșit?

P5. Postfixat. Calculează cu o stivă: `6 2 3 + * 4 -` , `2 3 4 * + 5 -` , `8 2 - 3 -` și `8 2 3 - -` . De ce ultimele două dau rezultate diferite?

P6. Unda lui Lee. Completează distanța minimă (număr de pași, sus/jos/stânga/dreapta) de la S la fiecare celulă liberă. `#` = zid. Cât este distanța până la F?

	1	2	3	4	5
1	S	·	·	#	·
2	·	#	·	·	·
3	·	#	·	#	·
4	·	·	·	#	F

P7. Pointeri. Ce afișează?

```
int x = 5, y = 8;
int *p = &x, *q = &y;
*p = *q + 1;
p = q;
*p = 2 * x;
q = &x;
cout << x << ' ' << y << ' ' << *p + *q;
```

P8. Grilă cu listă. Lista este `3 → 1 → 4 → 1 → 5 → 9`. Ce se afișează la parcurgere după secvența de mai jos? Ce s-a întâmplat cu nodul care conținea 4?

```
nod *p = prim->urm;
nod *q = p->urm->urm;
p->urm = q;
q->info = p->info + prim->info;
```

Nivelul 2 · Găsește și repară

★★ găsește și repară

P9. Ce afișează `push(3); push(7); cout << top() << ' '; pop(); cout << top();`? Ce ar trebui să afișeze? Repară modificând **o singură** funcție.

```
int st[100], vf = 0;                  // global
void push(int x) { st[vf++] = x; }
int top()        { return st[vf]; }
void pop()       { vf--; }
```

P10. În codul lui Lee din lecție, un elev inițializează `d` cu 0 și testează „nevizitat” cu `d[ni][nj] == 0`. Distanța până la destinație iese corect, dar afișarea drumului (`afisDrum`) nu se mai oprește. Explică. Ce valoare are `d` în celula de start, la final, pe harta `S . F` (o linie, trei celule libere)?

P11. Lista este `3 → 5 → 8`, iar `p` arată spre nodul 5. Se inserează 7 după `p` așa. Ce afișează parcurgerea (primele 8 valori)? De ce? Repară.

```
nod *q = new nod; q->info = 7;
p->urm = q;
q->urm = p->urm;
```

P12. Rânduri amestecate. Pune în ordine corpul funcției `void stergeUltim(nod *&prim)`, care șterge ultimul nod (merge și pentru lista goală și pentru lista cu un singur nod). Un rând este în plus.

```
A)  if (prim->urm == NULL) { delete prim; prim = NULL; return; }
B)  nod *p = prim;
C)  while (p->urm->urm != NULL)
D)      p = p->urm;
E)  delete p->urm;
F)  p->urm = NULL;
G)  if (prim == NULL) return;
H)  while (p->urm != NULL)
```

P13. Programul de mai jos ar trebui să afișeze `3 2 1`. Ce afișează? Ce caracter lipsește?

```cpp
void adaugaInceput(nod *prim, int x) {
    nod *p = new nod; p->info = x; p->urm = prim; prim = p;
}
// în main:
nod *prim = NULL;
for (int x = 1; x <= 3; x++) adaugaInceput(prim, x);
for (nod *p = prim; p != NULL; p = p->urm) cout << p->info << ' ';
```

P14. Coada circulară. Coada folosește doar 5 poziții, reluate „în cerc". Completează golurile. Apoi: după `push` 1, 2, 3, 4, două `pop`, `push` 5, 6, 7, ce conține vectorul `q[0..4]` și cât este `front()`?

```cpp
const int N = 5;
int q[N], prim = 0, nr = 0;        // nr = câte elemente are coada
void push(int x) { q[_____________] = x; nr++; }
int front()      { return q[prim]; }
void pop()       { prim = _____________; nr--; }
```

Nivelul 3 · Scrie cod

★★ scrie cod

Spune întâi ce structură folosești și de ce. Verifică **toate** testele.

P15. Editorul cu Undo. Comenzi: `T x` (scrie caracterul x la sfârșit), `B` (șterge ultimul caracter; pe text gol nu face nimic și nu se ține minte), `U` (anulează ultima comandă `T` sau `B` încă neanulată). Afișează textul final sau `(gol)`.

intrare	ieșire
`5` · `T a` · `T b` · `B` · `T c` · `U`	`a`
`4` · `T x` · `B` · `U` · `U`	`(gol)`
`6` · `T a` · `T b` · `T c` · `B` · `B` · `U`	`ab`

P16. Imprimanta corectă. `n` documente așteaptă la imprimantă; documentul i are `p[i]` pagini. Imprimanta ia primul document din coadă și tipărește cel mult `K` pagini; dacă documentul nu e gata, trece la sfârșitul cozii. Afișează documentele în ordinea în care se termină.

intrare (`n K`, apoi paginile)	ieșire
`3 2` · `3 1 4`	`2 1 3`
`4 5` · `5 2 7 1`	`1 2 4 3`
`2 1` · `2 2`	`1 2`

P17. Calul. Pe o tablă `n × n`, câte mutări de cal trebuie, minimum, de la `(si, sj)` la `(fi, fj)`? `-1` dacă nu se poate. (Lee cu 8 direcții.)

intrare	`8 1 1 8 8`	`8 1 1 1 1`	`3 1 1 2 2`
ieșire	`6`	`0`	`-1`

P18. Playlist scurt. Duratele melodiilor sunt într-o listă înlănțuită. Șterge **toate** melodiile mai lungi de `X` minute (atenție la cele de la început și la cele consecutive) și afișează lista sau `gol`.

intrare (`n` , duratele, `X`)	ieșire
`5` · `4 3 5 2 6` · `4`	`4 3 2`
`3` · `7 1 9` · `5`	`1`
`2` · `6 7` · `5`	`gol`

P19. Clasamentul. Citești punctajele pe rând și inserezi fiecare **direct la locul lui** într-o listă ordonată descrescător (fără sortare la final). Afișează lista.

intrare	`5` · `70 90 80 90 60`	`1` · `50`	`3` · `10 20 30`
ieșire	`90 90 80 70 60`	`50`	`30 20 10`

P20. Când se încălzește? Pentru fiecare zi, după câte zile vine prima zi **mai caldă** (0 dacă nu vine). O singură parcurgere, cu o stivă de indici.

intrare	ieșire
`8` · `73 74 75 71 69 72 76 73`	`1 1 4 2 1 1 0 0`
`5` · `30 40 50 60 70`	`1 1 1 1 0`
`3` · `50 50 50`	`0 0 0`

Nivelul 4 · Pune mintea la contribuție

★★★ pune mintea la contribuție

P21. Ce poate ieși dintr-o stivă? Numerele 1, 2, 3, 4, 5 intră în stivă în această ordine, dar poți scoate oricând vârful. Care dintre ieșirile următoare sunt posibile? Pentru cele posibile, scrie secvența de `push` / `pop`. a) `3 2 1 5 4` b) `4 5 3 1 2` c) `1 2 3 4 5` d) `2 4 3 5 1` e) `3 1 2 4 5`

P22. Coadă din două stive. Ai doar stivele `in` și `out`. `push` pune în `in`. Pentru `front` / `pop`: dacă `out` e goală, muți **tot** din `in` în `out`, apoi lucrezi pe vârful lui `out`. Ce afișează `push 1, push 2, front, pop, push 3, front, pop, front`? De ce fiecare element este mutat cel mult o dată din `in` în `out`?

P23. Mijlocul, dintr-o singură trecere. Fără să numeri nodurile, găsește nodul din mijlocul listei (pentru un număr par de noduri, al doilea dintre cele două din mijloc). Indiciu: doi pointeri, unul face un pas, celălalt doi. Teste: `1 2 3 4 5` → `3`, `1 2 3 4 5 6` → `4`, `7` → `7`.

P24. Ce se întâmplă dacă... în Lee folosești o **stivă** în loc de coadă (restul codului identic, direcțiile sus, dreapta, jos, stânga)? Pe o matrice 3 × 3 fără ziduri, cu startul în (1,1), ce valoare se scrie în (1,3)? Dar în (3,3)? Ce concluzie tragi?

P25. Mini-proiect: butoanele browserului. Pornești de pe pagina `acasa`. Comenzi: `viz u` (deschizi pagina u), `inapoi`, `inainte` (când nu se poate, rămâi pe loc). După fiecare comandă afișezi pagina curentă. Folosește două stive. Ce se întâmplă cu „înainte" după un `viz`?

intrare	ieșire
`6` · `viz a` · `viz b` · `inapoi` · `inapoi` · `inapoi` · `inainte`	`a b a acasa acasa a`

intrare	ieșire
`8` · `viz a` · `viz b` · `viz c` · `inapoi` · `inapoi` · `inainte` · `viz d` · `inainte`	`a b c b a b d d`

Probleme de lucru în clasă (pbinfo)

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

- [ ] #875 **stiva** — ușoară
  https://www.pbinfo.ro/probleme/875/stiva
- [ ] #876 **coada** — ușoară
  https://www.pbinfo.ro/probleme/876/coada
- [ ] #848 **paranteze1** — ușoară
  https://www.pbinfo.ro/probleme/848/paranteze1
- [ ] #1168 **fadaugarelalista** — ușoară
  https://www.pbinfo.ro/probleme/1168/fadaugarelalista
- [ ] #1169 **fafisarelista** — ușoară
  https://www.pbinfo.ro/probleme/1169/fafisarelista

Temă pentru acasă (pbinfo)

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

- [ ] #852 **paranteze3** — ușoară
  https://www.pbinfo.ro/probleme/852/paranteze3
- [ ] #3277 **lee** — medie
  https://www.pbinfo.ro/probleme/3277/lee
- [ ] #1884 **uemm1** — medie
  https://www.pbinfo.ro/probleme/1884/uemm1
- [ ] #1176 **flistastergere** — ușoară
  https://www.pbinfo.ro/probleme/1176/flistastergere
- [ ] #3789 **flistaoglindire** — ușoară
  https://www.pbinfo.ro/probleme/3789/flistaoglindire