FIȘĂ DE LUCRU · CLASA A XI-A · FIȘA 9 · ACOPERĂ LECȚIILE XI.15–XI.16

Greedy și programare dinamică

Durată: 120 min Nivel: intensiv

CE VEI ȘTI DUPĂ LECȚIE

- Decizi dacă un criteriu greedy este corect sau îl dărâmi cu un contraexemplu mic.
- Completezi de mână tabele de programare dinamică (scări, drumuri, SCLM, rucsac, monede).
- Scrii starea, recurența, baza și răspunsul pentru o problemă nouă, apoi codul.

Ce trebuie să știi (o pagină)

Problema	Regula	Cod-cheie
Greedy, în general	sortezi după un criteriu, iei ce se poate, **nu te mai întorci**	`sort(v + 1, v + n + 1, cmp);`
spectacole	după ora de **sfârșit**; iei dacă nu se suprapune	`if (v[i].st >= ultim) { nr++; ultim = v[i].sf; }`
rucsac fracționar	după `V/G` descrescător; ultimul obiect poate fi luat parțial	`total += v * C / g;`
clienți la o casă	durata crescător	`asteptare += curent; curent += t[i];`
PD, în general	**stare** (ce e `d[i]`, în cuvinte), **recurență**, **bază**, **răspuns**	
scări (1 sau 2 trepte)	`d[i] = d[i-1] + d[i-2]`	`d[0] = d[1] = 1`
triunghi	de jos în sus	`d[i][j] = a[i][j] + max(d[i+1][j], d[i+1][j+1])`
drumuri în matrice	din sus + din stânga; obstacol → 0	`d[i][j] = d[i-1][j] + d[i][j-1]`
cel mai lung subșir crescător	`L[i]` = cel mai lung care **se termină** în `a[i]`	răspuns: `max L[i]`, nu `L[n]`
rucsac 0/1	`d[c] = max(d[c], d[c-g] + v)`, cu `c` **descrescător**	`for (c = G; c >= g[i]; c--)`
monede (număr minim)	`d[s] = 1 + min d[s - m_k]`	`d[0] = 0`, restul `INF`

ATENȚIE — GREȘELI FRECVENTE

Un greedy „evident” poate fi greșit. Înainte să-l codezi, încearcă să-l spargi cu 3–4 obiecte. Dacă alegerea de acum depinde de ce urmează, ai nevoie de programare dinamică.

Nivelul 1 · Prezice

★ citește și prezice

P1. Festivalul. Spectacolele sunt deja sortate după sfârșit: `(1,3) (2,4) (3,5) (5,6) (4,7) (6,8) (8,9) (7,9)`. Câte alege greedy-ul din lecție (cu `>=`)? Dar dacă cineva scrie `>` în loc de `>=`? Scrie spectacolele alese în ambele cazuri.

P2. Monede ciudate. Ce afișează programul pentru `S = 15` și pentru `S = 21`? Pentru care dintre ele greedy dă numărul minim de monede?

```
int b[] = {10, 7, 1}, S, nr = 0;
cin >> S;
for (int i = 0; i < 3; i++)
    while (S >= b[i]) { cout << b[i] << ' '; S -= b[i]; nr++; }
cout << "| " << nr;
```

P3. Scara cu salturi de 1 sau 3. Poți urca 1 sau 3 trepte o dată. Scrie recurența și completează tabelul.

i	0	1	2	3	4	5	6	7	8
d[i]	1								

P4. Drumuri cu obstacole. Mergi doar la dreapta sau în jos; `X` = obstacol. Completează `d[i][j]` (numărul de drumuri până în celulă). Câte drumuri ajung în colțul dreapta-jos?

	1	2	3	4	5
1	1	·	·	·	·
2	·	X	·	·	X
3	·	·	·	X	·
4	X	·	·	·	·

P5. SCLM cu urmăritor. Pentru șirul `2 9 4 7 3 8 5 6`, completează `L[i]` și `tata[i]` ca în programul din lecție. Ce subșir afișează programul? Mai există un alt subșir de aceeași lungime?

P6. Rucsacul pe rânduri. Capacitate 7, obiecte (g, v): (3, 4), (4, 5), (2, 3). Completează vectorul `d` după fiecare obiect (bucla după `c` descrescătoare).

c	0	1	2	3	4	5	6	7
după obiectul 1								
după obiectul 2								
după obiectul 3								

P7. Doar monede de 2 și 5. Programul din lecție (număr minim de monede, `-1` = imposibil) rulează cu monedele 2 și 5. Ce afișează pentru fiecare sumă?

S	1	2	3	4	5	6	7	8	9	10
răspuns										

P8. Greedy contra PD pe triunghi. „Greedy" coboară mereu spre cel mai mare dintre cele două numere de dedesubt. Ce sumă obține? Care este suma maximă reală (calculează `d` de jos în sus) și pe ce drum?

```
    4
   8 5
  1 2 9
 6 3 2 8
```

Nivelul 2 · Găsește și repară

★★ găsește și repară

P9. Bucla rucsacului a fost scrisă crescător. Ce afișează pentru `G = 6` și obiectele (3, 5), (4, 6)? Ce ar trebui să afișeze? Ce problemă rezolvă, de fapt, codul greșit?

```
for (int i = 1; i <= n; i++)
    for (int c = g[i]; c <= G; c++)
        d[c] = max(d[c], d[c - g[i]] + v[i]);
```

P10. Un elev afișează `L[n]` ca răspuns la SCLM. Programul lui trece testul din lecție (`3 1 4 1 5 9 2 6 → 4`). De ce trece? Dă un test cu 5 numere pe care pică și spune ce afișează.

P11. La scări (1 sau 2 trepte) cineva a pus `d[0] = 0; d[1] = 1;`. Ce afișează pentru `n = 4`? Pentru ce valori ale lui `n` de la 1 la 10 răspunsul este totuși corect?

P12. Rânduri amestecate. Pune în ordine rândurile care calculează numărul minim de monede (după citire). Un rând este în plus.

```
A)  for (int s = 1; s <= S; s++) d[s] = INF;
B)  cout << (d[S] == INF ? -1 : d[S]);
C)      for (int k = 1; k <= m; k++)
D)  for (int s = 1; s <= S; s++)
E)          if (mon[k] <= s && d[s - mon[k]] != INF)
F)              d[s] = min(d[s], d[s - mon[k]] + 1);
G)  d[0] = INF;
```

P13. Pentru drumuri în matrice, un elev „pregătește marginile" așa. Ce afișează pe o matrice 3 × 3 cu un singur obstacol în (1, 2)? Care e răspunsul corect? Repară.

```
for (int j = 1; j <= m; j++) d[1][j] = 1;
for (int i = 1; i <= n; i++) d[i][1] = 1;
for (int i = 2; i <= n; i++)
    for (int j = 2; j <= m; j++)
        d[i][j] = (a[i][j] == 1) ? 0 : d[i-1][j] + d[i][j-1];
```

P14. Completează golurile. Programul calculează `d` pentru triunghi (ultima linie: `d[n][j] = a[n][j]`), apoi afișează **drumul** optim, de sus în jos. Ce afișează pe triunghiul din P8?

```
for (int i = ____; i >= 1; i--)
    for (int j = 1; j <= ____; j++)
        d[i][j] = a[i][j] + max(________, ________);
int j = 1;
for (int i = 1; i <= n; i++) {
    cout << a[i][j] << ' ';
    if (i < n && _____________________) j++;
}
```

Nivelul 3 · Scrie cod

★★ scrie cod

Pentru fiecare program, scrie întâi în cuvinte criteriul greedy sau starea `d`. Verifică **toate** testele.

P15. Udatul plantelor. Planta i trebuie udată cel puțin o dată într-o zi din intervalul `[a, b]`. Într-o zi în care uzi, uzi toate plantele. Care este numărul minim de zile de udat? (Indiciu: ce plantă e cea mai „grăbită”?)

intrare	ieșire
`3` · `1 3` · `2 5` · `4 6`	`2`
`4` · `1 2` · `3 4` · `5 6` · `7 8`	`4`
`3` · `1 10` · `2 3` · `4 5`	`2`

P16. Benzinăria. Drumul are `L` km, cu plinul faci `D` km, iar benzinăriile sunt la pozițiile date (crescător). Pleci cu plinul făcut. Câte opriri minime faci? `-1` dacă nu se poate.

intrare (`L D n`, apoi pozițiile)	ieșire
`10 4 4` · `2 4 6 8`	`2`
`10 3 3` · `2 6 8`	`-1`
`5 5 2` · `1 2`	`0`

P17. Robotul colecționar. Robotul pleacă din stânga-sus, merge doar la dreapta sau în jos și adună monedele din celule. Câte monede poate aduna, maximum, până în dreapta-jos?

intrare	ieșire
`3 3` · `1 3 1` · `1 5 1` · `4 2 1`	`12`
`1 4` · `2 0 3 1`	`6`
`2 2` · `5 1` · `1 5`	`11`

P18. Codul secret. Literele sunt codate `A = 1, B = 2, …, Z = 26`, iar mesajul a fost scris fără spații. În câte feluri poate fi decodat un șir de cifre? (`226` = `2 2 6`, `22 6`, `2 26`.) Atenție la cifra `0`.

intrare	`12`	`226`	`1111`	`2101`	`30`
ieșire	`2`	`3`	`5`	`1`	`0`

P19. Playlist pentru drum. Drumul durează `T` minute. Alegi melodii (fiecare cel mult o dată) cu durata totală cât mai mare, dar cel mult `T`. Afișează durata totală.

intrare (`n T` , apoi duratele)	ieșire
`3 10` · `3 5 4`	`9`
`4 15` · `4 8 6 3`	`15`
`2 7` · `8 9`	`0`

P20. Testerul de greedy. Citește `m` monede (crescător, prima este 1) și o sumă `S`. Afișează câte monede dă greedy, câte dă optimul (PD) și `OK` sau `GRESIT`.

intrare	ieșire
`3` · `1 3 4` · `6`	`3 2 GRESIT`
`4` · `1 5 10 25` · `30`	`2 2 OK`
`3` · `1 7 10` · `15`	`6 3 GRESIT`

Nivelul 4 · Pune mintea la contribuție

★★★ pune mintea la contribuție

P21. Vânătoarea de contraexemple. Monedele sunt `{1, 5, x}`, cu `x` de la 6 la 10. Pentru ce `x` greedy greșește la cel puțin o sumă? Care este cea mai mică sumă la care greșește? Ce observi?

P22. Cadouri cu buget. Ai `B` lei și prețurile unor cadouri. (a) De ce „cele mai ieftine primele" dă **numărul maxim** de cadouri? Explică printr-un argument de schimb. (b) Arată, cu un contraexemplu cu 3 prețuri, că același greedy **nu** dă suma cheltuită maximă (cel mult `B`).

P23. Memorie minimă. (a) Rescrie scările (1 sau 2 trepte) cu doar două variabile, fără vector. (b) La drumuri în matrice păstrezi un singur vector `d[1..m]`: pe obstacol `d[j] = 0`, altfel `d[j] += d[j-1]` (pentru `j > 1`). De ce aici bucla după `j` merge **crescător**, iar la rucsac descrescător? Verifică pe grila din P4.

P24. Ordinea buclelor. Ambele programe pornesc cu `d[0] = 1`. Unul numără modurile de plată în care **ordinea contează** (1+2 ≠ 2+1), celălalt pe cele în care nu contează. Care este care? Ce afișează fiecare pentru monedele 1, 2 și `S = 5`?

```
// A
for (int s = 1; s <= S; s++)
    for (int k = 1; k <= m; k++)
        if (mon[k] <= s) d[s] += d[s - mon[k]];
// B
for (int k = 1; k <= m; k++)
    for (int s = mon[k]; s <= S; s++) d[s] += d[s - mon[k]];
```

P25. Problema inversă. Într-o matrice 3 × 3 (drumuri doar la dreapta / în jos) pui **un singur** obstacol, nu pe start și nu pe final. Unde trebuie pus ca să rămână exact 3 drumuri? Găsește toate pozițiile. Se poate obține exact 4? De ce?

Probleme de lucru în clasă (pbinfo)

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

- [ ] #353 **spectacole** — ușoară
  https://www.pbinfo.ro/probleme/353/spectacole
- [ ] #1877 **kmax** — ușoară
  https://www.pbinfo.ro/probleme/1877/kmax
- [ ] #1991 **trepte2** — ușoară
  https://www.pbinfo.ro/probleme/1991/trepte2
- [ ] #385 **sumtri** — ușoară
  https://www.pbinfo.ro/probleme/385/sumtri
- [ ] #396 **sclm** — ușoară
  https://www.pbinfo.ro/probleme/396/sclm

Temă pentru acasă (pbinfo)

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

- [ ] #92 **proiecte** — ușoară
  https://www.pbinfo.ro/probleme/92/proiecte
- [ ] #1340 **rucsac** — medie
  https://www.pbinfo.ro/probleme/1340/rucsac
- [ ] #4186 **monede** — ușoară
  https://www.pbinfo.ro/probleme/4186/monede
- [ ] #392 **cladire** — ușoară
  https://www.pbinfo.ro/probleme/392/cladire
- [ ] #1886 **rucsac1** — medie
  https://www.pbinfo.ro/probleme/1886/rucsac1