FIȘĂ DE LUCRU · PYTHON · CLASA A IX-A · FIȘA 15 · LECȚIA PY9-15

Clasa list din Python

Durată: 150 min Nivel: mate-info · Python

CE VEI ȘTI DUPĂ LECȚIE

- Prezici efectul operatorilor și al metodelor listei, inclusiv ce întorc.
- Repari greșelile tipice: a = a.sort(), b = a în loc de copie, index pe o valoare lipsă, ștergerea în timpul parcurgerii.
- Scrii programe scurte de gestiune (stocuri, coș, to-do list) cu metodele potrivite.

Ce trebuie să știi (o pagină)

Ce vrei	Scrii	Modifică lista?	Întoarce
element / ultimul	`a[i]`, `a[-1]`	—	elementul
apartenență	`x in a`, `x not in a`	nu	`True` / `False`
lipire, repetare	`a + b`, `a * 3`	nu (listă nouă)	lista nouă
comparație	`==`, `!=`, `<`, `>` (lexicografic)	nu	`True` / `False`
bucată	`a[i:j]`, `a[:k]`, `a[k:]`, `a[::-1]`	nu (listă nouă)	lista nouă
adaugă la sfârșit	`a.append(x)` / `a.extend(b)`	da	`None`
inserează pe poziția `i`	`a.insert(i, x)`	da	`None`
scoate de pe poziția `i`	`a.pop(i)` (implicit ultimul)	da	elementul scos
șterge prima apariție	`a.remove(x)`	da	`None` (`ValueError` dacă lipsește)
caută / numără	`a.index(x)`, `a.count(x)`	nu	poziția / numărul
inversează, golește	`a.reverse()`, `a.clear()`	da	`None`
sortează	`a.sort()` / `sorted(a)`	da / **nu**	`None` (deci niciodată `a = a.sort()`) / lista nouă
copiază	`a.copy()`, `a[:]`, `list(a)`	nu	lista nouă
alt nume pentru aceeași listă	`b = a`	—	`b is a` este `True`

Nivelul 1 · Prezice

★ citește și prezice

P1. `a = [1, 2]`, `b = a * 2 + [3]`. Ce afișează `print(b, len(b), b.count(2), 4 in b)`?

P2. `True` sau `False`? `print([1, 10] < [1, 9], [3] == [3.0], ["b"] > ["abc"], [] < [0])`

P3. Ce afișează?

```
s = [5, 3, 8]
s.insert(1, 7)
s.append(s[0] + s[-1])
s.remove(8)
print(s, s.index(13))
```

P4. Ce afișează cele două `print`-uri?

```
a = [1, 2, 3]
print(a.append(4))
x = a.pop(0)
print(x, a)
```

P5. Pentru `t = [3, 1, 4, 1, 5, 9, 2, 6]`, ce afișează `print(t[2:5], t[-3:], t[1::3], t[5:2])`?

P6. Ce conțin cele trei liste la final?

```
a = [1, 2, 3]
b = a
c = a[:]
a[0] = 9
b.append(4)
print(a, b, c)
```

P7. Ce afișează?

```
p = [30, 10, 20]
q = sorted(p, reverse=True)
p.reverse()
print(p, q)
```

P8. `append`, `extend` sau `+=`? Ce afișează?

```
a = [1]; a.append([2, 3])
b = [1]; b.extend([2, 3])
c = [1]; c += [4]
print(a, len(a), b, c)
```

Nivelul 2 · Găsește și repară

★★ găsește și repară

P9. Programul trebuie să șteargă **toate** zerourile din `a = [0, 0, 1, 0, 2]`, dar afișează `[1, 0, 2]`. De ce rămâne un zero? Scrie două variante corecte.

```
for x in a:
    if x == 0:
        a.remove(x)
print(a)
```

P10. Programul trebuie să afișeze nota cea mai mică după sortare. Ce eroare apare și de ce?

```
note = [8, 5, 10]
note = note.sort()
print(note[0])
```

P11. Programul calculează prețurile cu reducere de 20%, dar vrea să păstreze și prețurile vechi. Afișează `vechi: [80, 32, 200]`. Explică și repară.

```
preturi = [100, 40, 250]
reduse = preturi
for i in range(len(reduse)):
    reduse[i] = reduse[i] * 80 // 100
print("vechi:", preturi)
print("noi:", reduse)
```

P12. Pentru `monitor` afișează `raftul 3`, dar pentru `boxe` programul se oprește. Ce eroare apare? Repară-l să afișeze `nu avem`.

```
stoc = ["mouse", "tastatura", "monitor"]
cautat = input()
print("raftul", stoc.index(cautat) + 1)
```

P13. *Rânduri amestecate.* Programul citește coșul de cumpărături (un rând), apoi produsele pe care clientul le scoate (alt rând), elimină câte o bucată din fiecare (dacă există), afișează coșul, apoi îl golește la plată și afișează numărul de produse rămase. Un rând este în plus.

```
A)  cos.clear()
B)  scoase = input().split()
C)      if p in cos:
D)  print(cos)
E)  cos = input().split()
F)          cos.remove(p)
G)  for p in scoase:
H)          cos.pop(p)
I)  print(len(cos))
```

Ce afișează pentru `lapte paine oua lapte` și `lapte cafea`?

P14. *Traducere greșită din C++.* Găsește **patru** greșeli în varianta Python. Ce trebuie să afișeze?

```
vector<int> v;
for (int x : {5, 2, 8}) v.push_back(x);
v.insert(v.begin() + 1, 7);
sort(v.begin(), v.end());
cout << v.size();
```

```
v = []
for x in [5, 2, 8]:
    v.push(x)
v.insert(v.begin() + 1, 7)
sort(v)
print(v.size())
```

Nivelul 3 · Scrie cod

★★ scrie cod

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

P15. Două depozite. Pe două rânduri sunt produsele din depozitul 1 și din depozitul 2. Afișează lista comună (concatenarea), iar pe rândul următor produsele care apar în **ambele** depozite, o singură dată, în ordinea din depozitul 1 (sau `-`).

intrare	ieșire
`pix caiet carioca` / `rigla pix guma`	`pix caiet carioca rigla pix guma` / `pix`
`a b` / `c d`	`a b c d` / `-`
`x y x` / `y x`	`x y x y x` / `x y`

P16. Inserare ordonată. Pe primul rând este o listă sortată crescător (poate fi goală), pe al doilea un număr `x`. Inserează `x` astfel încât lista să rămână sortată, **fără** `sort()`: cauți poziția, apoi `insert`.

intrare	ieșire
`2 5 9 12` / `7`	`2 5 7 9 12`
`2 5 9` / `1`	`1 2 5 9`
`2 5 9` / `20`	`2 5 9 20`
(rând gol) / `4`	`4`
`3 3 8` / `3`	`3 3 3 8`

P17. Rotația. Citește o listă și un număr `k ≥ 0`. Rotește lista la stânga cu `k` poziții folosind **slicing** (primele `k` elemente trec la sfârșit).

intrare	ieșire
`1 2 3 4 5` / `2`	`3 4 5 1 2`
`1 2 3 4 5` / `0`	`1 2 3 4 5`
`1 2 3 4 5` / `7`	`3 4 5 1 2`
`A` / `3`	`A`

P18. Fără cea mai mică notă. Profesorul ignoră cea mai mică notă (o singură dată, chiar dacă apare de mai multe ori). Afișează media celorlalte cu 2 zecimale, sau `prea puține note` dacă sunt mai puțin de două.

intrare	ieșire
`7 4 9 10`	`8.67`
`5 5 8`	`6.50`
`9`	`prea puține note`
`10 10`	`10.00`

P19. To-do list. Citește `n`, apoi `n` comenzi: `add X` (adaugă la sfârșit), `urgent X` (pune la început), `done X` (șterge, dacă există), `sort`, `clear`. `X` poate avea mai multe cuvinte: `input().split(maxsplit=1)` taie rândul doar la primul spațiu. La final afișează sarcinile separate prin `, ` sau `nimic de făcut`.

intrare	ieșire
`4` / `add teme` / `add sport` / `urgent pastile` / `done sport`	`pastile, teme`
`4` / `add b` / `add a` / `add c` / `sort`	`a, b, c`
`3` / `add x` / `clear` / `done x`	`nimic de făcut`
`2` / `done y` / `add citit carte`	`citit carte`

P20. Fără dubluri. Citește o listă de numere și afișează-o fără valorile repetate (păstrezi **prima** apariție, în ordinea inițială), iar pe rândul următor câte elemente au fost eliminate.

intrare	ieșire
`3 1 3 2 1 3`	`3 1 2` / `3`
`5 6 7`	`5 6 7` / `0`
`4 4 4 4`	`4` / `3`
(rând gol)	(rând gol) / `0`

Nivelul 4 · Pune mintea la contribuție

★★★ pune mintea la contribuție

P21. Pe loc sau listă nouă? Pentru fiecare pereche, cele două instrucțiuni dau același conținut pentru `a`. Folosește `v = a` înainte și `print(a is v)` după, ca să afli care **modifică lista existentă** și care **creează alta**: (a) `a += [3]` și `a = a + [3]`; (b) `a.insert(0, 0)` și `a = [0] + a`; (c) `a.reverse()` și `a = a[::-1]`. De ce contează asta când lista a fost primită ca parametru de o funcție?

P22. Copia superficială. Ce afișează? De ce lista `m` s-a modificat într-un loc, dar nu și în celălalt?

```
m = [[1, 2], [3, 4]]
c = m.copy()
c[0].append(9)
c[1] = [0]
print(m, c)
```

P23. insert(0, x) costă. Vrei lista numerelor `n − 1, …, 1, 0`. Varianta A face `a.insert(0, i)` pentru `i` de la 0 la `n − 1`. Varianta B face `append(i)`, apoi un singur `reverse()`. La fiecare `insert(0, …)` toate elementele existente se mută cu o poziție. Câte mutări face A pentru `n = 1000`? Rulează ambele variante pentru `n = 20000` și `n = 40000` (cu `time.perf_counter()`). Ce observi când dublezi `n`?

P24. Problemă inversă. Găsește lista `a` cu 4 elemente pentru care, simultan: `sorted(a) == [1, 3, 3, 5]`, `a.index(3) == 2`, `a.count(3) == 2` și `a[::-1] < a`. Câte soluții are fără ultima condiție?

P25. Mini-proiect: stocul unui magazin. Scrie trei funcții care lucrează pe o listă `stoc` (fiecare bucată de produs este un element): `adauga(stoc, produs, buc)` (folosește `extend` și `*`), `vinde(stoc, produs)` (șterge o bucată; întoarce `True` sau `False` dacă nu există) și `raport(stoc)` (întoarce o listă de texte `"produs: bucăți"`, în ordine alfabetică, fiecare produs o dată). Testează scenariul:

```
stoc = []
adauga(stoc, "pix", 3)
adauga(stoc, "caiet", 2)
print(vinde(stoc, "pix"), vinde(stoc, "guma"))   # True False
print(raport(stoc))                              # ['caiet: 2', 'pix: 2']
inainte = stoc.copy()
vinde(stoc, "caiet"); vinde(stoc, "caiet")
print(raport(stoc), raport(inainte))             # ['pix: 2'] ['caiet: 2', 'pix: 2']
```

Probleme de lucru în clasă (pbinfo)

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

- [ ] #1452 **stergere-element** — ușoară
  https://www.pbinfo.ro/probleme/1452/stergere-element
- [ ] #158 **inserare** — ușoară
  https://www.pbinfo.ro/probleme/158/inserare
- [ ] #488 **afisare** — ușoară
  https://www.pbinfo.ro/probleme/488/afisare

Temă pentru acasă (pbinfo)

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

- [ ] #500 **verificareordonat** — ușoară
  https://www.pbinfo.ro/probleme/500/verificareordonat
- [ ] #505 **identice** — ușoară
  https://www.pbinfo.ro/probleme/505/identice
- [ ] #162 **permcirc** — medie
  https://www.pbinfo.ro/probleme/162/permcirc