FIȘĂ DE LUCRU · PYTHON · CLASA A IX-A · FIȘA 9 · LECȚIA PY9-09

Subprograme (2): variabile locale și globale, funcții predefinite

Durată: 100 min Nivel: mate-info · Python

CE VEI ȘTI DUPĂ LECȚIE

- Prezici efectul variabilelor locale, globale și al transmiterii unui int sau a unei liste.
- Alegi corect între int, round, math.floor, math.ceil și folosești len, min, max, sum cu atenție la cazurile limită.
- Rescrii cod cu global în funcții cu parametri și return și construiești o mică aplicație modulară.

Ce trebuie să știi (o pagină)

Noțiune / funcție	Exemplu → rezultat	De reținut
variabilă locală	creată în funcție (și parametrii)	dispare la `return`; nu se vede din afară
variabilă globală	creată în afara funcțiilor	o funcție o poate **citi**
atribuire în funcție	`total = total + 1`	o face locală → `UnboundLocalError`
`global x`	permite modificarea	de evitat; folosește parametri + `return`
int transmis	`x = x * 2` în funcție	argumentul **nu** se schimbă
listă transmisă	`v.append(0)` în funcție	lista se schimbă **și** pentru apelant
`abs`, `round`	`abs(-3) → 3`, `round(2.5) → 2`	la „,5” exact: numărul **par**
`int`	`int(-3.9) → -3`	taie zecimalele (spre 0)
`math.floor`, `math.ceil`	`-3.9 → -4`, respectiv `-3`	în jos / în sus
`math.sqrt`	`math.sqrt(25) → 5.0`	real; `ValueError` pentru negative
conversii	`int("7")`, `float("2.5")`, `str(3)`	`int("3.5")` → `ValueError`
`len`, `sum`	pentru `[]`: `0`, `0`	media unei liste goale → `ZeroDivisionError`
`min`, `max`	`max([])` → `ValueError`	textele se compară alfabetic: `max("9", "10") → "9"`

SFAT

Înainte să scrii `import math` și `math.ceil(…)`, întreabă-te ce rezultat vrei pentru valori negative și pentru ,5. Diferențele dintre `int`, `round` și `floor` apar exact acolo.

Nivelul 1 · Prezice

★ citește și prezice

P1. Ce afișează? Ce se întâmplă la ultimul rând?

```
a = 1
def f():
    b = 2
    return a + b
print(f())
print(b)
```

P2. Ce afișează?

```
s = 100
def suma(n):
    s = 0
    for i in range(1, n + 1):
        s += i
    return s
print(suma(4), s)
```

P3. Ce afișează?

```
k = 0
def pas():
    global k
    k += 2
    return k
print(pas(), pas(), k)
```

P4. Ce afișează?

```
def f(x, v):
    x += 1
    v.append(x)
a, l = 1, []
f(a, l)
f(a, l)
print(a, l)
```

P5. `import math`, apoi: `print(round(2.5), round(3.5), round(-2.5), int(-3.9), math.floor(-3.9), math.ceil(-3.9), round(1234.5678, -2))`.

P6. Pentru `v = [4, -2, 7, 0, -5]`: `print(len(v), min(v), max(v), sum(v), max(v) - min(v), abs(min(v)))`.

P7. `print(int("7") * 2, "7" * 2, float("2") / 4, str(3.0) + "1", int(float("4.9")))`.

P8. Ce afișează pentru `n = 24` și pentru `n = 25`: `print(math.sqrt(n), int(math.sqrt(n)) ** 2 == n)`?

Nivelul 2 · Găsește și repară

★★ găsește și repară

P9. Programul dă `UnboundLocalError`. Explică de ce și rescrie funcția **fără** `global`, astfel încât programul să afișeze `15`.

```
total = 0
def adauga(pret):
    total = total + pret
adauga(10)
adauga(5)
print(total)
```

P10. Regulamentul spune: media se rotunjește „de la ,50 în sus". Programul folosește `nota = int(m)`. Ce dă pentru `9.5`, `9.49` și `6.5`? Ce dau `round(m)` și `math.floor(m + 0.5)`? Care variantă respectă regulamentul?

P11. Funcția calculează media unui elev, adăugând un 10 bonus. Ce afișează programul? De ce cele două apeluri dau rezultate diferite? Repar-o.

```
def media_cu_bonus(note):
    note.append(10)
    return sum(note) / len(note)
n = [6, 8]
print(media_cu_bonus(n), media_cu_bonus(n), n)
```

P12. *Rânduri amestecate.* Programul citește coordonatele a trei puncte, `A`, `B`, `C`, și afișează perimetrul triunghiului, rotunjit la 2 zecimale. Un rând este în plus. Ce afișează pentru `0 0 3 0 3 4`?

```
A)  import math
B)  def dist(x1, y1, x2, y2):
C)  return math.sqrt((x1 - x2) ** 2 + (y1 - y2) ** 2)
D)  xa, ya, xb, yb, xc, yc = map(int, input().split())
E)  p = dist(xa, ya, xb, yb) + dist(xb, yb, xc, yc) + dist(xc, yc, xa, ya)
F)  print(round(p, 2))
G)  return sqrt((x1 - x2) ** 2 + (y1 - y2) ** 2)
```

P13. Programul citește prețuri și afișează cel mai mare. Pentru `9 10 100` afișează `9`. De ce? Repară-l.

```
preturi = input().split()
print(max(preturi))
```

P14. Ce returnează `nota_max([])`? Ce se întâmplă la `nota_max([]) + 1`? Propune o variantă care tratează clar lista goală.

```
def nota_max(v):
    if len(v) > 0:
        return max(v)
```

Nivelul 3 · Scrie cod

★★ scrie cod

Fiecare program are funcții cu parametri și `return`, fără variabile globale.

P15. Distanțe. Scrie `distanta(x1, y1, x2, y2)` (cu `math.sqrt`) și `manhattan(x1, y1, x2, y2)` (cu `abs`: distanța mergând doar pe orizontală și verticală, ca pe străzile unui cartier). Afișează distanța obișnuită cu 2 zecimale, apoi distanța Manhattan.

intrare	ieșire
`0 0 3 4`	`5.00 7`
`1 1 4 -3`	`5.00 7`
`-2 3 -2 3`	`0.00 0`
`0 0 1 1`	`1.41 2`

P16. Notele arbitrilor. La gimnastică se elimină o notă maximă și o notă minimă, iar scorul este media celorlalte. Scrie `scor_final(v)`, folosind `sum`, `min`, `max`, `len`. Presupunem că sunt cel puțin 3 note.

intrare	ieșire
`9 8 10 7 9`	`8.67`
`5 5 5`	`5.00`
`10 1 9 9`	`9.00`

P17. Taxiul. Pornirea costă 3,50 lei, iar fiecare kilometru început costă 2,79 lei (2,1 km se plătesc ca 3 km). Scrie `cost(km)`.

intrare	ieșire
`1`	`6.29`
`2.1`	`11.87`
`0`	`3.50`
`10`	`31.40`

P18. Pătrate perfecte. Scrie `este_patrat(n)` folosind `math.sqrt` și `int`, apoi numără pătratele perfecte de pe un rând.

intrare	ieșire
`1 2 4 10 16 24 25`	`4`
`0 3`	`1`
`1000000 999999`	`1`

P19. Catalogul. Citește notele unui elev. Afișează: numărul de note, media cu 2 zecimale, nota minimă, nota maximă și media rotunjită „de la ,50 în sus” (funcția `rotunjeste(m)`).

intrare	ieșire
`7 10 9 8 10`	`5 8.80 7 10 9`
`9 10`	`2 9.50 9 10 10`
`8 9`	`2 8.50 8 9 9`
`4`	`1 4.00 4 4 4`

P20. Jos globalele! Rescrie programul de mai jos fără `global`, cu o funcție `media(v)` care folosește `sum` și `len`. Teste: `4 6 8` → `6.0`, `5` → `5.0`.

```
suma = 0
nr = 0
def adauga(x):
    global suma, nr
    suma += x
    nr += 1
for x in map(int, input().split()):
    adauga(x)
print(suma / nr)
```

Nivelul 4 · Pune mintea la contribuție

★★★ pune mintea la contribuție

P21. Banii și numerele reale. Ce afișează `print(round(2.675, 2), 0.1 + 0.1 + 0.1 == 0.3, sum([0.1, 0.1, 0.1]))`? De ce? Într-o aplicație de facturare, cum ai evita problema? *Indiciu:* bani, nu lei.

P22. sum() sau buclă? Scrie `suma_mea(v)` cu o buclă și verifică pe 1000 de liste aleatoare (`random.randint`) că dă același rezultat ca `sum(v)`, inclusiv pentru lista goală. Dă două argumente pentru `sum` și unul pentru care trebuie să știi totuși bucla.

P23. Problemă inversă. O listă de 4 numere naturale, scrisă crescător, are `min = 3`, `max = 9` și `sum = 20`. Câte astfel de liste există? Care sunt?

P24. Cine depinde de cine? Pentru programul de mai jos, scrie variabilele locale ale fiecărei funcții și variabilele globale. Ce afișează? Care funcție nu mai poate fi testată „singură" și de ce?

```
TAXA = 5
def pret(p):
    return p + TAXA
def total(a, b):
    s = pret(a) + pret(b)
    return s
print(total(10, 20))
TAXA = 0
print(total(10, 20))
```

P25. Mini-proiect: consumul de apă. Se citesc consumurile zilnice (litri) ale unei gospodării. Proiectează și scrie funcțiile `media(v)`, `ziua_maxima(v)` (numărul zilei, de la 1), `variatie_maxima(v)` (cea mai mare diferență, în valoare absolută, între două zile consecutive) și `zile_critice(v, prag)`.

Programul afișează un raport. Zilele critice sunt cele cu consum cu peste 20% mai mare decât media.

intrare	ieșire
`120 130 90 200 110 115 125`	`media: 127.14`, `maxim: 200 (ziua 4)`, `variație maximă: 110`, `zile critice: 1`
`100 100 100`	`media: 100.00`, `maxim: 100 (ziua 1)`, `variație maximă: 0`, `zile critice: 0`
`50`	`media: 50.00`, `maxim: 50 (ziua 1)`, `variație maximă: 0`, `zile critice: 0`

P26. Mini-atelier: o aplicație existentă. Un grup de trei elevi primește programul de mai jos (media notelor, cu validarea datelor):

```
def citeste_note():
    return list(map(float, input().split()))

def valida(note):
    for x in note:
        if x < 1 or x > 10:
            return False
    return len(note) > 0

def medie(note):
    return sum(note) / len(note)

def afiseaza(m):
    print(f"media: {m:.2f}")

note = citeste_note()
if valida(note):
    afiseaza(medie(note))
else:
    print("date invalide")
```

a. **Analiză și diagramă.** Desenează diagrama modulelor: câte o căsuță pentru fiecare funcție și pentru programul principal, cu săgeți „apelează". Ce primește și ce întoarce fiecare modul? Ce s-ar întâmpla dacă `valida` ar fi scoasă din program? b) **Două descompuneri.** Compară acest program cu o variantă într-o singură funcție `raport()`, care citește, validează, calculează și afișează: cum testezi `medie` în fiecare variantă? Ce schimbi dacă datele vin dintr-un fișier? c) **Lucru în grup.** Repartizați cele patru module între trei persoane. Ce trebuie stabilit **înainte** să scrie fiecare (numele funcțiilor, parametrii, ce întoarce)? Cum verifică fiecare, singur, modulul lui? d) **Completează aplicația.** Adaugă un modul nou `categorie(m)`: `insuficient` pentru `m < 5`, `suficient` pentru `m < 7`, `bine` pentru `m < 9`, altfel `foarte bine`. Programul trebuie să afișeze `media: 5.00 (suficient)`. Ce funcții din cod modifici, în afară de cea nouă?

intrare	ieșire
`4 5 6`	`media: 5.00 (suficient)`
`9 10`	`media: 9.50 (foarte bine)`

intrare	ieșire
`3 4`	`media: 3.50 (insuficient)`
`7 7.5`	`media: 7.25 (bine)`
`11 8` sau rând gol	`date invalide`

Probleme de lucru în clasă (pbinfo)

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

- [ ] #1255 **lipsa** — ușoară
  https://www.pbinfo.ro/probleme/1255/lipsa
- [ ] #435 **factoriprimi** — ușoară
  https://www.pbinfo.ro/probleme/435/factoriprimi

Temă pentru acasă (pbinfo)

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

- [ ] #365 **cifre-comune** — ușoară
  https://www.pbinfo.ro/probleme/365/cifre-comune
- [ ] #445 **pseudoperfect** — ușoară
  https://www.pbinfo.ro/probleme/445/pseudoperfect