INFORMATICĂ · PYTHON · CLASA A VIII-A · TABLOURI UNIDIMENSIONALE · LECȚIA 1

Recapitulare: decizii și repetiții în Python

Durată: 50 min Nivel: gimnaziu · Python

CE VEI ȘTI DUPĂ LECȚIE

- Îți amintești cele trei structuri: liniară, alternativă (if) și repetitivă (for, while).
- Citești numere cu input() și afișezi rezultate cu print().
- Scrii corect blocurile cu două puncte și indentare.

1. Ce știi deja

Anul acesta lucrăm cu **șiruri de numere**. Ca să le stăpânim, ne trebuie trei lucruri din clasa a VII-a. Le repetăm în Python.

Structura	Ce face	În Python
liniară	pașii se execută unul după altul	rânduri una sub alta
alternativă	alege între două drumuri	`if` … `else`
repetitivă	repetă pași	`for` sau `while`

2. Citire și afișare

`input()` citește un rând de la tastatură. Rezultatul este **text**, deci pentru un număr scrii `int(...)`.

```
a = int(input())
b = int(input())
print("Suma este", a + b)
```

Tastezi `4` și `9`:

```
Suma este 13
```

ATENȚIE — GREȘELI FRECVENTE

Fără `int(...)`, Python lipește textele: `"4" + "9"` dă `"49"`, nu `13`.

3. Decizia: if / elif / else

După `if` scrii o condiție și **două puncte**. Rândurile care depind de ea se trag spre dreapta cu **4 spații** (indentare).

```python
x = int(input())
if x % 2 == 0:
    print("par")
else:
    print("impar")
print("gata")
```

Pentru `7` afișează:

```
impar
gata
```

`print("gata")` nu este tras spre dreapta, deci se execută mereu. Când ai mai mult de două drumuri, folosești `elif` („altfel, dacă”):

```python
nota = 8
if nota >= 9:
    print("foarte bine")
elif nota >= 5:
    print("bine")
else:
    print("mai încearcă")
```

```
bine
```

Compari	Scrii	Atenție
egal	`==`	un singur `=` înseamnă atribuire
diferit	`!=`	
și / sau	`and` / `or`	`x > 0 and x < 10`

4. Repetiția cu for

`for` repetă de un număr cunoscut de ori. `range(a, b)` dă numerele de la `a` până la `b - 1`: **capătul din dreapta nu se ia**.

```python
for i in range(1, 6):
    print(i, end=" ")
```

```
1 2 3 4 5
```

Scrii	Valorile lui `i`
`range(5)`	0, 1, 2, 3, 4
`range(1, 6)`	1, 2, 3, 4, 5
`range(2, 11, 2)`	2, 4, 6, 8, 10
`range(5, 0, -1)`	5, 4, 3, 2, 1

`end=" "` îi spune lui `print` să pună un spațiu după valoare, nu să treacă pe rândul următor.

5. Repetiția cu while

`while` repetă **cât timp** o condiție este adevărată. Folosește-l când nu știi de câte ori trebuie să repeți. Exemplu: de câte ori poți tăia un număr în două, până ajunge sub 1?

```
n = 100
pasi = 0
while n >= 1:
    n = n // 2
    pasi += 1
print(pasi)
```

```
7
```

`//` este împărțirea întreagă (`100 // 2` este `50`, `7 // 2` este `3`), iar `pasi += 1` înseamnă `pasi = pasi + 1`.

> **SFAT**
>
> Dacă programul nu se mai oprește, probabil nu modifici nimic în buclă. Oprește-l cu `Ctrl + C` și verifică variabila din condiție.

6. Verifică-te

1. Ce afișează `print(17 // 5, 17 % 5)`?
2. Ce este greșit: `if x = 5:`?
3. Câte numere afișează `for i in range(10):`?
4. Cu ce înlocuiești `range(1, 6)` ca să obții și numărul 6?
5. Ce se întâmplă dacă uiți `:` după `while n > 0`?
6. Ce afișează `for i in range(3, 8): print(i, end=" ")`?
7. Scrie o buclă care afișează numerele pare de la 2 la 20.

Răspunsuri

1. `3 2` (17 = 3 · 5 + 2).
2. Pentru comparare se scrie `==`: `if x == 5:`.
3. 10 numere: de la 0 la 9.
4. Cu `range(1, 7)`.
5. Python dă eroare de sintaxă (`SyntaxError`), iar programul nu pornește.
6. `3 4 5 6 7`.
7. `for i in range(2, 21, 2): print(i, end=" ")`.

Probleme de lucru în clasă (pbinfo)

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

- [ ] #105 **max2** — ușoară
  https://www.pbinfo.ro/probleme/105/max2
- [ ] #109 **paritate** — ușoară
  https://www.pbinfo.ro/probleme/109/paritate
- [ ] #51 **suma** — ușoară
  https://www.pbinfo.ro/probleme/51/suma

Temă pentru acasă (pbinfo)

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

- [ ] #2 **suma-n** — ușoară
  https://www.pbinfo.ro/probleme/2/suma-n
- [ ] #327 **afisarenumere** — ușoară
  https://www.pbinfo.ro/probleme/327/afisarenumere
