FIȘĂ DE LUCRU · PYTHON · CLASA A VIII-A · FIȘA 06 · LECȚIA PY8G-06

Căutarea unui element și verificări

Durată: 50 min Nivel: gimnaziu · Python

CE VEI ȘTI DUPĂ LECȚIE

- Prezici rezultatul unei căutări liniare și al unei verificări cu steag.
- Repari steaguri puse greșit și limite greșite la range.
- Scrii programe care caută, verifică și decid.

Ce trebuie să știi (o pagină)

Întrebarea	Steag la început	Când îl schimbi
Există un element cu proprietatea?	`gasit = False`	găsești unul → `True`
Au **toate** elementele proprietatea?	`ok = True`	găsești unul care **nu** o are → `False`
La ce poziție este `x`?	`poz = -1`	găsești `x` → `poz = i`, apoi `break`
Este șirul crescător?	`ok = True`	`v[i] > v[i + 1]` → `False`

Cod	Observație
`for i in range(n - 1):`	când compari `v[i]` cu `v[i + 1]`
`break`	oprește bucla imediat
`if gasit:` / `if not gasit:`	folosești direct steagul, nu `== True`

SFAT

La „toate elementele...” caută **excepția**: un singur element care strică regula este suficient ca răspunsul să fie `False`.

Nivelul 1 · Prezice

★ citește și prezice

P1. Ce afișează?

```
v = [12, 7, 9, 20, 15]
x = 8
gasit = False
for e in v:
    if e == x:
        gasit = True
print(gasit)
```

P2. Ce afișează?

```
v = [3, 6, 6, 9]
x = 6
poz = -1
for i in range(len(v)):
    if v[i] == x:
        poz = i
        break
print(poz)
```

P3. Același program ca la P2, dar **fără** `break`. Ce afișează? Ce înseamnă acum `poz`?

```
v = [3, 6, 6, 9]
x = 6
poz = -1
for i in range(len(v)):
    if v[i] == x:
        poz = i
print(poz)
```

P4. Ce afișează?

```
v = [4, -1, 7]
ok = True
for x in v:
    if x <= 0:
        ok = False
print(ok)
```

P5. Ce afișează?

```
v = [1, 3, 3, 2, 5]
ok = True
for i in range(len(v) - 1):
    if v[i] > v[i + 1]:
        ok = False
print(ok)
```

P6. Ce afișează?

```
v = [5, 3, 8, 2, 2, 1]
c = 0
for i in range(len(v) - 1):
    if v[i] > v[i + 1]:
        c += 1
print(c)
```

Nivelul 2 · Găsește și repară

★★ găsește și repară

P7. Vrem poziția lui `8` în listă (`1`), dar programul afișează `-1`. Găsește greșeala.

```
v = [4, 8, 15]
x = 8
poz = -1
for i in range(len(v)):
    if v[i] == x:
        poz = i
    else:
        poz = -1
print(poz)
```

P8. Vrem să aflăm dacă **toate** numerele sunt pare (`[2, 4, 6, 9]` are unul impar, deci răspunsul corect este `False`). Programul afișează `True` . Repară-l.

```
v = [2, 4, 6, 9]
ok = False
for x in v:
    if x % 2 == 0:
        ok = True
print(ok)
```

P9. Verificăm dacă lista este crescătoare, dar programul se oprește cu o eroare. Care este ea? Repară-l.

```
v = [1, 4, 6, 9]
ok = True
for i in range(len(v)):
    if v[i] > v[i + 1]:
        ok = False
print(ok)
```

P10. Vrem să verificăm dacă șirul este **strict** crescător (fără elemente egale). Pentru `[1, 2, 2, 3]` răspunsul corect este `False` , dar programul afișează `True` . Repară-l.

```
v = [1, 2, 2, 3]
ok = True
for i in range(len(v) - 1):
    if v[i] > v[i + 1]:
        ok = False
print(ok)
```

Nivelul 3 · Scrie cod

★★ scrie cod

P11. Există? Citește `n` , un rând cu `n` numere și apoi un număr `x` . Afișează `da` dacă `x` se află în șir, altfel `nu` .

intrare	ieșire

`5`
`4 8 15 16 23`
`15`
	`da`

intrare	ieșire

`3`
`1 2 3`
`7`
	`nu`

`1`
`5`
`5`
	`da`

P12. Unde este? Aceleași date. Afișează poziția **primei** apariții a lui `x` (de la 0) sau `-1` dacă nu există.

intrare	ieșire

`4`
`3 6 6 9`
`6`
	`1`

`3`
`1 2 3`
`5`
	`-1`

`5`
`7 7 7 7 7`
`7`
	`0`

P13. Toate pare? Citește `n` și un rând cu `n` numere. Afișează `da` dacă toate sunt pare, altfel `nu`.

intrare	ieșire
`3` `2 4 6`	`da`
`4` `2 4 5 6`	`nu`
`1` `0`	`da`

P14. Crescător? Citește `n` și un rând cu `n` numere. Afișează `da` dacă șirul este crescător (elementele egale sunt permise), altfel `nu`.

intrare	ieșire
`5` `1 2 2 5 9`	`da`
`4` `1 3 2 4`	`nu`
`1` `5`	`da`

intrare	ieșire
`3` `9 8 7`	`nu`

P15. Ultima apariție. Citește `n`, un rând cu `n` numere și apoi `x`. Afișează poziția **ultimei** apariții a lui `x` sau `-1`.

intrare	ieșire

`5`
`2 5 2 5 2`
`5`
	`3`

`3`
`1 2 3`
`2`
	`1`

`3`
`1 2 3`
`9`
	`-1`

Nivelul 4 · Provocare

★★★ pune mintea la contribuție

P16. Ce fel de șir? Citește `n` (cel puțin 2) și un rând cu `n` numere. Afișează `crescător` dacă fiecare element este **strict** mai mic decât următorul, `descrescător` dacă fiecare este strict mai mare decât următorul, altfel `neordonat`.

intrare	ieșire
`4` `1 3 5 8`	`crescător`
`3` `9 4 1`	`descrescător`
`4` `1 3 2 4`	`neordonat`
`3` `2 2 3`	`neordonat`

P17. Codul secret. Codul secret este lista `[3, 1, 4, 1, 5]`. Jucătorul scrie 5 cifre pe un rând. Afișează câte cifre sunt pe **poziția corectă** (aceeași poziție și aceeași valoare).

intrare	ieșire
`3 1 4 1 5`	`5`
`1 1 1 1 1`	`2`
`5 4 3 2 1`	`0`
`3 4 4 2 5`	`3`

P18. Elementul-sumă. Citește `n` și un rând cu `n` numere. Afișează `da` dacă există un element egal cu **suma celorlalte** (de exemplu în `1 2 3 6`, numărul `6` este `1 + 2 + 3`), altfel `nu`. Hint: întâi calculezi suma totală.

intrare	ieșire
`4` `1 2 3 6`	`da`
`3` `1 2 4`	`nu`
`3` `5 0 5`	`da`