FIȘĂ DE LUCRU · PYTHON · CLASA A VIII-A · FIȘA 10 · LECȚIA PY8G-10

Vectori de frecvență: cifre și note

Durată: 50 min Nivel: gimnaziu · Python

CE VEI ȘTI DUPĂ LECȚIE

- Prezici conținutul unui vector de frecvență.
- Repari greșeli de dimensiune, de inițializare și de limite.
- Scrii programe cu frecvențe pentru cifre, note și zaruri.

Ce trebuie să știi (o pagină)

Ce vrei	Cod
vector de frecvență pentru cifre	`f = [0] * 10`
vector pentru note de la 1 la 10	`f = [0] * 11`
numeri o valoare `x`	`f[x] += 1`
cifrele unui număr `n`	`while n > 0:` ↵ `····f[n % 10] += 1` ↵ `····n //= 10`
de câte ori apare `k`	`f[k]`
valoarea cu cea mai mare frecvență	`best = 0` ↵ `for k in range(len(f)):` ↵ `····if f[k] > f[best]: best = k`
câte valori diferite apar	numeri indicii cu `f[k] > 0`

LA BAC

Dimensiunea vectorului de frecvență este **valoarea maximă + 1**. Cu 10 căsuțe, singurii indici valizi sunt 0...9.

Nivelul 1 · Prezice

★ citește și prezice

P1. Ce afișează?

```
f = [0] * 5
for x in [1, 3, 3, 0, 3, 1]:
    f[x] += 1
print(f)
```

P2. Ce afișează?

```
n = 40503
f = [0] * 10
while n > 0:
    f[n % 10] += 1
    n //= 10
print(f[0], f[3], f[7])
```

P3. Ce afișează?

```
n = 777177
f = [0] * 10
while n > 0:
    f[n % 10] += 1
    n //= 10
print(f[7], f[1])
```

P4. Câte cifre **diferite** are numărul? Ce afișează programul?

```
n = 112233
f = [0] * 10
while n > 0:
    f[n % 10] += 1
    n //= 10
k = 0
for c in range(10):
    if f[c] > 0:
        k += 1
print(k)
```

P5. Ce afișează?

```
note = [5, 5, 8, 10, 8, 5]
f = [0] * 11
for x in note:
    f[x] += 1
best = 0
for g in range(11):
    if f[g] > f[best]:
        best = g
print(best, f[best])
```

P6. Ce afișează?

```
f = [0] * 7
for x in [2, 6, 6, 1, 2, 6]:
    f[x] += 1
for i in range(1, 7):
    if f[i] == 0:
        print(i, end=" ")
```

Nivelul 2 · Găsește și repară

★★ găsește și repară

P7. Programul numără notele, dar se oprește cu o eroare. De ce? Repară-l.

```
f = [0] * 10
for nota in [3, 10, 7]:
    f[nota] += 1
print(f)
```

P8. Pentru `n = 1221` ar trebui să obținem `[0, 2, 2, 0, 0, 0, 0, 0, 0, 0]`, dar programul afișează altceva. Care este greșeala?

```
n = 1221
while n > 0:
    f = [0] * 10
    f[n % 10] += 1
    n //= 10
print(f)
```

P9. Programul se oprește cu o eroare de tip `TypeError`. Care este cauza? Repară-l.

```
n = 123
f = [0] * 10
while n > 0:
    f[n % 10] += 1
    n = n / 10
print(f)
```

P10. Vrem frecvențele notelor de la 1 la 10, dar în afișare lipsește nota 10. Repară-l.

```
f = [0, 0, 1, 0, 0, 0, 2, 0, 0, 3, 1]
for nota in range(1, 10):
    print(nota, f[nota])
```

Nivelul 3 · Scrie cod

★★ scrie cod

P11. Cifrele unui număr. Citește un număr natural `n` (mai mare ca 0). Afișează pentru fiecare cifră care apare, în ordine crescătoare, un rând de forma `cifra: de_câte_ori`.

intrare	ieșire

`2023123`	`0: 1`
	`1: 1`
	`2: 3`
	`3: 2`

`7`	`7: 1`
`1100`	`0: 2` `1: 2`

P12. Cifra preferată. Citește un număr natural `n` (mai mare ca 0). Afișează cifra care apare de cele mai multe ori; la egalitate, cea mai mică.

intrare	ieșire
`2023123`	`2`

intrare	ieșire
`5`	`5`
`1122`	`1`

P13. Câte cifre diferite? Citește un număr natural `n` (mai mare ca 0) și afișează câte cifre **diferite** are.

intrare	ieșire
`2023123`	`4`
`777`	`1`
`1234567890`	`10`

P14. Clasamentul notelor. Citește `n` și un rând cu `n` note (între 5 și 10). Pentru fiecare notă de la 10 la 5 (descrescător) care apare, afișează `nota: de_câte_ori`.

intrare	ieșire

`8`
`9 7 10 9 8 7 9 10`
	`10: 2`
	`9: 3`
	`8: 1`
	`7: 2`

`3` `5 5 5`	`5: 3`
`4` `10 10 10 10`	`10: 4`

P15. Zarul norocos. Citește `n` și un rând cu `n` aruncări de zar (valori de la 1 la 6). Afișează fața care a apărut cel mai des și de câte ori, pe același rând. La egalitate, fața cea mai mică.

intrare	ieșire
`6` `3 5 3 1 3 5`	`3 3`
`4` `6 6 1 1`	`1 2`
`1` `4`	`4 1`

Nivelul 4 · Provocare

★★★ pune mintea la contribuție

P16. Aceleași cifre. Citește două numere naturale `a` și `b`, câte unul pe rând. Afișează `da` dacă au **aceleași cifre**, cu aceleași frecvențe, în orice ordine (de exemplu `1123` și `3211`), altfel `nu`. Hint: două liste se compară cu `==`.

intrare	ieșire
`1123` `3211`	`da`
`1123` `1233`	`nu`
`10` `100`	`nu`
`5` `5`	`da`

P17. Cel mai mare număr. Citește un număr natural `n` (mai mare ca 0). Afișează cel mai mare număr care se poate forma cu **toate** cifrele lui, fără să sortezi lista: folosește frecvențele, de la cifra 9 la 0. (Verifică întâi dacă merge pentru `2023123`.)

intrare	ieșire
`2023123`	`3322210`
`105`	`510`
`7`	`7`

P18. Valori mai mari. Citește `n` și un rând cu `n` numere între 0 și 1000. Afișează câte valori **diferite** apar. Câte căsuțe are vectorul de frecvență? De ce nu ai putea folosi aceeași metodă pentru numere până la `1 000 000 000`?

intrare	ieșire
`6` `5 1000 5 7 1000 0`	`4`
`3` `1 1 1`	`1`
`4` `10 20 30 40`	`4`