ALGORITMIKUS ALAPOK & PYTHON MEGOLDÁSOK

Elemi Programozási Tételek a Gyakorlatban

Összegzés, átlagolás, megszámlálás, eldöntés, szélsőértékek és lineáris keresés ciklusokkal és beépített függvényekkel

Klasszikus algoritmusok vs. Modern Python eszközök

STRUKTÚRA

A Hat Alapvető Algoritmuscsalád

1. Összegzés & Átlag

Értékek göngyölítése és számtani közép számítása.

2. Megszámlálás

Adott tulajdonságú elemek darabszámának felderítése.

3. Eldöntés

Létezés vizsgálata korai ciklusbontással.

4. Min & Max Kiválasztás

Legkisebb és legnagyobb elem felderítése 0. indexről indulva.

5. Lineáris Keresés

Elem meglétének és pontos indexének azonosítása.

1. TÉTEL — AKKUMULÁCIÓ

Összegzés Tétele (Költések Összegzése)

```python
# Bolti vásárlási tételek összeadása
kiadasok = [4500, 12000, 3200, 8900]

# 1. Hagyományos for ciklus
osszeg = 0
for tetel in kiadasok:
    osszeg += tetel

# 2. Beépített Python megoldás
gyors_osszeg = sum(kiadasok)

print(f"Végösszeg: {osszeg} Ft")
```

Akkumulátor változó

Kezdőértéke mindig 0, amelyhez ciklislépésenként hozzáadjuk az elemet.

sum() függvény

Közvetlen C-szintű összegzés, tiszta és hibamentes kód.

2. TÉTEL — SZÁMTANI KÖZÉP

Átlagolás Tétele (Tanulmányi Átlag)

```
# Tantárgyi érdemjegyek átlagolása
jegyek = [5, 4, 5, 3, 4, 5]

# sum() és len() kombinációja
if len(jegyek) > 0:
    atlag = sum(jegyek) / len(jegyek)
    print(f"Átlag: {atlag:.2f}")
else:
    print("Üres lista!")
```

Összeg és darabszám hányadosa

Az összegzés tételére épül, amelyet a tároló méretével osztunk.

Nullával osztás védelme

Üres lista esetén a len() értéke 0, amit kötelező if feltétellel védeni.

3. TÉTEL — FELTÉTELES SZÁMLÁLÁS

Megszámlálás Tétele (Fagypont Alatti Napok)

```python
# 0 fok alatti mérések számlálása
homersekletek = [3, -2, 5, -4, 0, -1]

fagynapok = 0
for fok in homersekletek:
    if fok < 0:
        fagynapok += 1

print(f"Fagyos napok: {fagynapok} db")
```

Szűrés + Számlálás

A számláló változó csak akkor növekszik 1-gyel, ha az if feltétel True.

count() metódus

Pontos értékegyezésnél közvetlenül elérhető: lista.count(ertek).

4. TÉTEL — LÉTEZÉSVIZSGÁLAT

Eldöntés Tétele (Allergén Ellenőrzés)

```
# Mogyoró jelenlétének vizsgálata
alapanyagok = ["liszt", "cukor", "mogyoró", "vaj"]

# 1. Ciklusos korai kilépés
van_mogyoro = False
for elem in alapanyagok:
    if elem == "mogyoró":
        van_mogyoro = True
        break  # Azonnali kilépés!

# 2. Python in operátor
if "mogyoró" in alapanyagok:
    print("Allergén jelen van!")
```

Korai kilépés (break)

Amint megtaláljuk az elemet, a ciklus leáll, nem vizsgálja a maradékot.

in operátor eleganciája

A Python nyelvi szinten támogatja a logikai létezésvizsgálatot.

5. TÉTEL — SZÉLSŐÉRTÉKEK

Minimum és Maximum (Repülőjegy Árak)

```python
# Legolcsóbb ajánlat felderítése
arak = [42000, 19500, 68000, 31000]

# 1. Aranyszabály: a legelső elem a bázis!
legkisebb = arak[0]
for ar in arak:
    if ar < legkisebb:
        legkisebb = ar

# 2. Beépített függvények
min_ar = min(arak)
max_ar = max(arak)

print(f"Legolcsóbb jegy: {min_ar} Ft")
```

Kezdőérték szabály

Soha ne használj fix 0-t! Mindig a lista legelső eleme (lista[0]) a bázis.

min() és max()

A beépített függvények egyetlen sorban visszaadják a szélsőértékeket.

6. TÉTEL — POZÍCIÓ ÉS INDEX

Lineáris Keresés (Versenyző Helyezése)

```python
# Név és pontos index megkeresése
futok = ["Anna", "Bence", "Csilla", "Dávid"]
keresett = "Csilla"

# 1. Ciklus indexléptetéssel
talalt_idx = -1
for i in range(len(futok)):
    if futok[i] == keresett:
        talalt_idx = i
        break

# 2. Beépített .index() metódus
if keresett in futok:
    pozicio = futok.index(keresett)
    print(f"{keresett} indexe: {pozicio}")
```

Index és érték felderítése

Nemcsak a létezést igazolja, hanem megadja az elem pontos címét.

.index() metódus

Mindig in operátorral együtt használd, mert hiányzó elemnél ValueError-t dob.

ÖSSZEGZÉS

Algoritmikus Eszköztár Gyorsnézet

Tétel	Ciklusos Művelet	Python Eszköz	Gyakorlati Cél
Összegzés	`osszeg += elem`	`sum(lista)`	Végösszeg kalkuláció
Átlagolás	`osszeg / darab`	`sum() / len()`	Érdemjegy átlag
Megszámlálás	`if feltetel: db += 1`	`lista.count()`	Fagynapok száma
Eldöntés	`if feltetel: break`	`elem in lista`	Allergén riasztás
Szélsőérték	`if elem < min: min = elem`	`min(), max()`	Legolcsóbb járat
Lineáris Keresés	`if lista[i] == x: break`	`lista.index()`	Versenyző helyezés



Algoritmusok Magabiztos Használata

A klasszikus tételek logikájának megértése fejleszti a programozói gondolkodást, a beépített Python függvények pedig gyors és tiszta kódot eredményeznek.

#Algoritmika #PythonAlapok