PYTHON PROGRAMOZÁS ALAPJAI

Aritmetikai Operátorok és Vezérlési Szerkezetek

Számítási mechanizmusok, hatékony operátorhasználat és döntési logikák Pythonban

15 fejezetes átfogó útmutató

A Tananyag Két Pillére

ÁTTEKINTÉS



1. Számítási Műveletek

Hét beépített aritmetikai operátor.

Precedencia és zárójelezési szabályok.

Lebegőpontos számábrázolási sajátságok.



2. Döntési Szerkezetek

Egy- és többirányú feltételes elágazások.

Logikai kifejezések és rövidzár.

Egyágú ternáris kifejezések.

Alapvető Aritmetika

+

Összeadás

Számok összege vagy szövegösszefűzés.

```
5 + 3 → 8
```

-

Kivonás

Két numerikus érték különbsége.

```
10 - 4 → 6
```

*****

Szorzás

Szorzat vagy szekvencia ismétlése.

```
4 * 3 → 12
```

/

Valós Osztás

Eredménye minden esetben lebegőpontos.

```
7 / 2 → 3.5
```

Egész Osztás és Maradék

// Egész Osztás (Floor Div)

Levágja a törtrészt a legközelebbi kisebb egész felé kerekítve.

```
17 // 5 # Eredmény: 3
-17 // 5 # Eredmény: -4
```

% Maradékképzés (Modulo)

Az osztás után fennmaradó egész maradékot adja vissza.

```
17 % 5 # Eredmény: 2
14 % 2 # Eredmény: 0 (páros)
```

Hatványozás és Unáris Művelet

Kettős csillag: **

Hatványozásra nem a kalap jelet (^) használjuk.

Tört kitevővel gyökvonás is közvetlenül végezhető.

Az unáris mínusz az érték ellentettjét képezi.

```
# Hatványozási példák
alap = 2
kitevo = 8
print(alap ** kitevo) # 256

# Négyzetgyök számítás:
gyok = 16 ** 0.5 # 4.0
```

Műveleti Erősség és Precedencia

Kiértékelési hierarchia magasabbtól az alacsonyabb felé

1. SZINT

Zárójelek ()

Bármely művelet felülbírálható.

2. SZINT

Hatványozás **

Jobbról balra asszociatív kötés.

3. SZINT

Szorzás, Osztás

*, /, //, % azonos szinten.

4. SZINT

Összeadás, Kivonás

+, - balról jobbra értékelve.

Aritmetikai Jó Gyakorlatok

Kifejezett Zárójelezés

Összetett képleteknél növeli az olvashatóságot.

Lebegőpontos Limit

0.1 + 0.2 nem pontosan 0.3 a memóriában.

Nullával Való Osztás

Mindig ellenőrizd az osztó értékét futás előtt.

Döntéshozatal a Kódban

VEZÉRLÉSI ELV

Feltételtől függő végrehajtás

A programok ritkán futnak lineárisan elejétől a végéig.

Logikai feltételek döntenek az egyes kódblokkok lefutásáról.



```python
#!/usr/bin/env python3
from Crypto.PublicKey import RSA
from secret import FLAG

for i in range(1, 6):
    with open('{}.pub.pem'.format(i), 'rb') as f:
        key = RSA.importKey(f.read())
    with open('{}.enc'.format(i), 'wb') as f:
        f.write(key.encrypt(FLAG.encode(), None)[0])
```

Az Alapvető if Utasítás

```
pontszam = 85

if pontszam ⩾ 50:
    print("Sikeres vizsga!")

# Csak True feltételnél fut le
```

Kötelező Kettőspont
A feltétel végét kettőspont jelzi.

Blokk Indentálás
A kódblokkot 4 szóköz behúzás definiálja.

Kétirányú Döntés: if – else

IGAZ ÁG

if feltétel:

Akkor fut le, ha a logikai kifejezés True értékű.

HAMIS ÁG

else:

Minden egyéb esetben automatikusan ez aktiválódik.

Többágú Szelekció: elif Lánc

```
if pontszam ⩾ 90:
    jegy = "Jeles"
elif pontszam ⩾ 80:
    jegy = "Jó"
elif pontszam ⩾ 60:
    jegy = "Közepes"
else:
    jegy = "Elégtelen"
```

Kizárólagosság

Csak a legelső igaz feltétel ága hajtódik végre.

Opcionális Zárás

Az else ág elhagyható, ha nincs alapértelmezett teendő.

Rövidzár Kiértékelés (Short-Circuit)

AND művelet

Első hamisnál megáll

Megelőzi a nullával való osztási hibákat.

```
if b ≠ 0 and a / b > 2:
```

OR művelet

Első igaznál megáll

Felesleges további feltételeket vizsgálni.

```
if admin or jog_ervenyes:
```

Egysoros Feltételes Kifejezés

SYNTAX

Ternáris Operátor

Tömör értékadás egyetlen sorban kifejezésként.

```
# Érték if feltétel else Alternatíva
statusz = "Páros" if x % 2 == 0 else
"Páratlan"

abszolút = -x if x < 0 else x
```

Tipikus Kezdő Bakik

= helyett ==

Egyenlőség vizsgálatához mindig dupla egyenlőségjel kell.

Kettőspont Hiánya

Az if, elif és else sorok végén kötelező szintaktikai elem.

== True Redundancia

A logikai változó önmagában is érvényes feltétel.



Összefoglalás és Következő Lépések

Aritmetikai műveletekkel számolunk, elágazásokkal döntünk — ez minden algoritmus fundamentuma.

Következő témakör: While és For ciklusok

Image Sources



https://w0.peakpx.com/wallpaper/327/815/HD-wallpaper-minimalism-simple-background-text-python-programming-code-rare-gallery-minimal-programmer.jpg

Source: www.peakpx.com