Curs 1.Tipurile de variabile în Python și operatori logici

1. Tipurile de variabile în Python

Python este un limbaj **dinamic**(high level language): tipul variabilei se stabilește automat în funcție de valoarea atribuită, fără a fi nevoie de o declarație explicită.Acest lucru face mai ușor pentru utilizator să scrie cod.

1.1. Tipuri numerice

- **int** -- numere întregi (pozitive, negative sau zero)

```
a = 42
b = -7
c = 0
print(type(a))  # <class 'int'>
```

- **float** -- numere reale (cu virgulă zecimală)

```
pi = 3.14159
nota = 9.75
print(type(pi))  # <class 'float'>
```

- **complex** -- numere complexe (cu parte reală și imaginară)

```
z = 2 + 3j
print(type(z))   # <class 'complex'>
```

1.2. Tipuri de secvențe

- **str** -- șiruri de caractere

```
mesaj = "Salut, lume!"
print(mesaj[0])   # S
print(len(mesaj)) # 12
```

- **list** -- colecție ordonată, modificabilă

```
lista = [1, 2, 3, "patru"]
lista.append(5)       # adaugă un element
print(lista)          # [1, 2, 3, 'patru', 5]
```

- **tuple** -- colecție ordonată, nemodificabilă

```
tuplu = (10, 20, 30)
print(tuplu[1])   # 20
```

1.3. Colecții set și dicționare

- **set** -- mulțime neordonată, fără elemente duplicate

```
s = {1, 2, 3, 3, 4}
print(s)  # {1, 2, 3, 4}
```

- **dict** -- colecție de perechi `cheie: valoare`

```
d = {"nume": "Ana", "vârsta": 25}
print(d["nume"])      # Ana
d["oras"] = "Cluj"    # adăugăm o cheie nouă
```

1.4. Tipuri logice și speciale

- **bool** -- valori logice `True` sau `False`

```
este_student = True
print(type(este_student))  # <class 'bool'>
```

- **NoneType** -- absența unei valori (echivalent cu `null`)

```
x = None
print(type(x))  # <class 'NoneType'>
```

---

2. Operatorii logici în Python

Operatorii logici lucrează cu valori **boolean** și returnează `True` sau `False`.

2.1. Operatorul and

Returnează `True` **doar dacă ambele condiții sunt adevărate**.

```
a = 5
print(a > 2 and a < 10)   # True
print(a > 2 and a > 10)   # False
```

2.2. Operatorul or

Returnează `True` dacă **cel puțin una** dintre condiții este adevărată.

```
a = 5
print(a < 2 or a < 10)    # True (a < 10 este adevărat)
print(a < 2 or a > 10)    # False (niciuna nu e adevărată)
```

2.3. Operatorul not

Inversează valoarea logică.\

- `not True` → `False`
- `not False` → `True`

```
a = 5
print(not(a > 2 and a < 10))   # False
```

2.4. Tabel logic

`Expresie` → `Rezultat`

- `True and True` → `True`
- `True and False` → `False`
- `False and False` → `False`
- `True or False` → `True`
- `False or False` → `False`
- `not True` → `False`
- `not False`→ `True`

---

3. Exemple practice

3.1. Verificarea parității unui număr

```
numar = 12
if numar % 2 == 0 and numar > 0:
    print("Număr par pozitiv")
```

3.2. Verificarea autentificării

```
user = "admin"
parola = "1234"

if user == "admin" and parola == "1234":
    print("Autentificare reușită!")
else:
    print("Date incorecte.")
```

3.3. Utilizare or

```
temperatura = 35
if temperatura < 0 or temperatura > 30:
    print("Avertizare meteo!")
```

---

4. Concluzie

- Python are **tipuri de date dinamice** și flexibile.\
- Cele mai utilizate tipuri: `int`, `float`, `str`, `bool`, `list`, `dict`.
- Operatorii logici (`and`, `or`, `not`) sunt esențiali pentru expresii condiționale și decizii în program.
- Combinarea tipurilor de date cu operatorii logici permite scrierea de programe robuste și clare.