



Tema 4.

Stringuri în Python



Ce ne așteaptă?

1. Definirea și accesarea stringurilor

2. Metodele stringurilor

1. Care sunt structurile de date și secvențele în Python
2. Cum se definește un string?
3. Ce reprezintă indicii stringului și pentru ce se folosesc?
4. Ce reprezintă operatorul de feliere și pentru ce se folosește?
5. Ce reprezintă imutabilitatea stringurilor?
6. Care sunt operatorii aplicați stringurilor?
7. Care sunt metodele de modificare minusculă-majusculă?
8. Care sunt metodele de eliminarea a spațiilor goale?
9. Care sunt metodele de căutare a substringurilor?
10. Care sunt metodele de divizare și de unificare a stringurilor
11. Ce metoda se utilizează pentru substituirea substringurilor?
12. Care sunt metodele de verificare a conținutului stringurilor?
13. Care sunt metodele de verificare a majusculilor-minusculilor?
14. Ce metode verifică conținutul de început și de sfârșit?

1. Definirea și accesarea stringurilor

Structuri de date

- **Structura de date** – păstrează un grup de obiecte individuale ca o singură entitate
- **Secvența** - structura de date ce permite citirea datelor prin parcurgerea element cu element
- **Principalele structuri de date în Python:**
 - String – o secvență imutabilă ordonată de caractere
 - List – o secvență mutabilă ordonată de obiecte
 - Tuple – o secvență imutabilă ordonată de obiecte
 - Dictionary – o structură de perechi “cheie-valoare”
 - Set – o structură neordonată de obiecte

Definirea stringurilor

- **Sintaxa:**

nume_variabila = 'string 1' # definire prin apostroafe - string pe o linie
nume_variabila = "string 2" # definire prin ghilimele - string pe o linie
nume_variabila = """string 3""" # definire prin apostroafe triple - string multi-linie
nume_variabila = """string 4""" # definire prin ghilimele triple - string multi-linie

- **Utilizare apostrof sau ghilimele in string**

```
s1 = "Bun venit în limbajul 'Python'"
s2 = 'Bun venit în limbajul "Python"'
s3 = """Bun venit în limbajul 'Python'"""
print(s1)
print(s2)
print(s3)
print(type(s1))
print(type(s2))
print(type(s3))
```

- **Stringul gol**

```
s1 = ""
print(s1)
```

Indecșii în string

- **Indexul – poziția caracterului în string**
- **Caracteristicile indecșilor**
 - Indecși pozitivi - primul simbol are indexul 0, al doilea 1 ...
 - Indecși negativi – ultimul simbol are indexul -1, penultimul -2 ...
 - Indexul este de tip int – altfel eroare TypeError
 - Indexul trebuie sa nu depășească lungimea stringului – altfel eroare IndexError
 - Indexul permite accesarea simbolului utilizând paranteze pătrate []
 - Simbolul accesat prin index este de tip string

0	1	2	3	4	5	6	7	8
A	B	C	D	E	F	G	H	I
-9	-8	-7	-6	-5	-4	-3	-2	-1

Accesarea caracterelor stringului cu ajutorul indecșilor

- Exemple de utilizare:

```
s1 = "Universitatea Tehnică a Moldovei"  
print(s1[0])  
print(s1[9])
```

```
s1 = "Universitatea Tehnică a Moldovei"  
print(s1[-1])  
print(s1[-5])
```

```
s1 = "Universitatea Tehnică a Moldovei"  
print(s1[1.6])
```

```
s1 = "Universitatea Tehnică a Moldovei"  
print(s1[100])
```

```
s1 = "Universitatea Tehnică a Moldovei"  
print(s1[6])  
print(type(s1[6]))
```

Felierea (slicing) stringurilor în Python

- felie (slice) – o parte a stringului
- Sintaxa:

nume_string[start:stop]

sau

nume_string[start:stop:pas]

start – valoarea primului index al slice in string (valoarea implicita este 0)

stop – valoarea primului index al stringului care nu se mai include în slice (valoarea implicita este utimul index)

pas – pasul de incrementare a indexului la accesare (valoarea implicita este 1)

- **Caracteristicile felierii**
 - Valorile sunt de tip int – altfel eroare TypeError
 - Slice-ul accesat prin slicing este de tip string

Felierea stringurilor în Python

- **Exemple de utilizare:**

```
s = "Universitatea Tehnică a Moldovei"  
print(s[:])  
print(s[::])  
print(s[0:9:1])  
print(s[0:9:2])  
print(s[2:4:1])  
print(s[::2])  
print(s[2::])  
print(s[:4:])  
print(s[-4:-1])  
print(s[::-1])
```

Imutabilitatea stringurilor

- **Imutabilitate** – proprietatea obiectului de a nu fi modificat
- **Exemple de verificare a imutabilității stringurilor:**

```
s = "Universitatea Tehnică a Moldovei"  
print(s)  
print(s[0])  
s[0]="u"
```

Operații matematice asupra stringurilor

- **Adunarea 2 stringuri = concatenare:**

- ambii termeni vor fi de tip string – altfel eroarea TypeError

```
s1 = "Universitatea "  
s2 = "Tehnică"  
s3 = s1 + s2  
print(s3)
```

- **Multiplicarea unui string:**

- un argument va fi de tip string iar altul de tip int – altfel eroarea TypeError

```
s1 = "Universitatea Tehnică "  
num = 3  
s2 = s1 * num  
print(s2)
```

- **Conversia altui tip compatibil de date în string – funcția str()**

```
n = 3  
print(n)  
print(type(n))  
s = str(n)  
print(s)  
print(type(s))
```

Lungimea stringului și operatorii de apartenență

- Lungimea stringului – funcția len()

```
s = "Universitatea Tehnică a Moldovei"  
lung = len(s)  
print(f'Stringul s contine {lung} caractere')
```

- Verificarea prezenței unui caracter (unei felii) în string

```
s = "Universitatea Tehnică a Moldovei"  
print("Tehnică" in s)  
print("Python" in s)
```

- Verificarea lipsei unui caracter (unei felii) în string

```
s = "Universitatea Tehnică a Moldovei"  
print("Tehnică" not in s)  
print("Python" not in s)
```

2. Metodele stringurilor

Metode de modificare minuscule-majuscule

- **Sintaxa**

<i>nume_string.upper()</i>	# toate devin majuscule
<i>nume_string.lower()</i>	# toate devin minuscule
<i>nume_string.swapcase()</i>	# minuscule devin majuscule și invers
<i>nume_string.title()</i>	# primele litere din cuvinte devin majuscule
<i>nume_string.capitalize()</i>	# doar prima litera a primului cuvânt devine majusculă

- **Exemplu de utilizare**

```
mesaj = 'sunt student la Universitatea Tehnică'  
print(mesaj.upper())  
print(mesaj.lower())  
print(mesaj.swapcase())  
print(mesaj.title())  
print(mesaj.capitalize())  
print(mesaj)
```

Metode de eliminare a spațiilor goale

- Sintaxa

nume_string.rstrip()

elimina spatiile goale din partea dreapta a stringului

nume_string.lstrip()

elimina spatiile goale din partea stanga a stringului

nume_string.strip()

elimina spatiile goale ambele parti ale stringului

- Exemplu de utilizare

```
s = "    Universitatea Tehnică a Moldovei    "  
s1= s.rstrip()  
s2= s.lstrip()  
s3= s.strip()  
print(s)  
print(len(s))  
print(s1)  
print(len(s1))  
print(s2)  
print(len(s2))  
print(s3)  
print(len(s3))
```

Metode de căutare a unui substring într-un string

- **Sintaxa**

nume_string.find(substring, start, stop)

returnează indexul de început al substringului în string

nume_string.rfind(substring, start, stop)

returnează indexul de început al substringului în string
începând de la final

nume_string.index(substring, start, stop)

returnează indexul de început al substringului în string

nume_string.rindex(substring, start, stop)

returnează indexul de început al substringului în string
începând de la final

- **find() vs index()**

- find() returnează -1 dacă substringul lipsește în string
- index() returnează o eroare ValueError dacă substringul lipsește în string

- **Exemplu de utilizare**

```
s="Universitatea Tehnică a Moldovei"  
print(s.find("t"))  
print(s.find("t", 9, 12))  
print(s.rfind("t"))  
print(s.find("y"))  
print(s.index("t"))  
print(s.index("t", 9, 12))  
print(s.rindex("t"))  
print(s.index("y"))
```

Metode de divizare și de unificare a stringurilor

- Sintaxa

nume_string.split(separator)

formează o lista de substringuri obținute
prin divizarea conform separatorului

separator.join(lista_substringuri)

formează un string din substringurile listei
plasând separatorul specificat între ele

- Exemplu de utilizare

```
s="Universitatea Tehnică a Moldovei"  
list_string = s.split(" ")  
print(s)  
print(list_string)
```

```
list_string=["Universitatea", "Tehnică", "a", "Moldovei"]  
s = "_".join(list_string)  
print(list_string)  
print(s)
```


Metode de numărarea a aparițiilor și de amplasare a substringurilor

- Sintaxa

nume_string.count(substring) # numarul de apariție a substringului în string

nume_string.replace(substring_vechi, substring_nou) # înlocuiește substring_vechi cu substring_nou

- Exemplu de utilizare

```
s="Universitatea Tehnică a Moldovei"  
print(s.count("e"))  
print(s.count("ve"))  
print(s.count("y"))
```

```
s1="Sunt profesor la Universitatea Tehnică a Moldovei"  
s2 =s1.replace(„profesor", "student")  
print(s1)  
print(s2)
```

Metode de verificare a conținutului stringurilor

- **.isalnum()** – verifică dacă toate caracterele sunt doar litere sau cifre

```
s="grupa FET-211"  
rez = s.isalnum()  
print(rez)
```

- **.isalpha()** – verifică dacă toate caracterele sunt doar litere

```
s="grupa FET-211"  
rez = s.isalpha()  
print(rez)
```

- **.isdigit()** – verifică dacă toate caracterele sunt doar cifre

```
s="grupa FET-211"  
rez = s.isdigit()  
print(rez)
```

- **.isspace()** – verifică dacă toate caracterele sunt spații goale

```
s="grupa FET-211"  
rez = s.isspace()  
print(rez)
```

Metode de verificare a majusculelor-minusculelor

- **.islower()** – verifică dacă toate caracterele sunt minuscule

```
s="Universitatea Tehnică a Moldovei"  
rez = s.islower()  
print(rez)
```

- **.isupper()** – verifică dacă toate caracterele sunt majuscule

```
s="Universitatea Tehnică a Moldovei"  
rez = s.isupper()  
print(rez)
```

- **.istitle()** – verifică dacă toate cuvintele încep cu majusculă

```
s="Universitatea Tehnică a Moldovei"  
rez = s.istitle()  
print(rez)
```

Metode de verificare a caracterelor de la început și la sfârșit

- Sintaxa

nume_string.startswith(substring) # verifică dacă stringul începe cu substringul dat

nume_string.endswith(substring) # verifică dacă stringul sfârșește cu substringul dat

- Exemplu de utilizare

```
s="Universitatea Tehnică a Moldovei"  
print(s.startswith("Univer"))  
print(s.startswith("Tehnic"))
```

```
s="Universitatea Tehnică a Moldovei"  
print(s.endswith("dovei"))  
print(s.endswith("."))
```