Лекція 3. Рядок тексту

У цій лекції, на відміну від більшості попередніх ми не будемо працювати із числами: дійсними або цілими. Ця лекція присвячена виключно роботі з текстом, але от питання: як відноситься рядок тексту до табличних величин?

Ми вже знаємо, що існує не один тип даних, який відноситься до табличних величин. Це списки та кортежі, про які ми дізналися в попередній лекції, а також ще багато інших. Тому існують функції переведення складних типів даних між собою (зі списків в кортежі та навпаки і т.п.). Але нас цікавить тільки одна, функція ***list()***, яка призначена для перетворення будь-якої табличної величини в список. Розглянемо на прикладі:

```
>>> a=(1, 2, 3)
>>> a=list(a)
>>> a
[1, 2, 3]
```

Зверніть увагу! Застосовувати подібні функції до окремих значень не можна, лише до табличних величин, або простих різнотипних списків:

```
>>> a=1
>>> list(a)
Traceback (most recent call last):
  File "<pyshell#12>", line 1, in <module>
    list(a)
TypeError: 'int' object is not iterable
```

Давайте подивимося, що відбудеться з текстом, якщо застосувати цю функцію до рядка тексту:

```
>>> a="text"
>>> a=list(a)
>>> a
['t', 'e', 'x', 't']
```

Саме так, текст, хоч і відноситься до простих типів даних, але він все одно є найпростішою табличною величиною, бо текстовий фрагмент складається з менших однотипних частинок — символів.

3.1 Робота з рядком тексту

Оскільки рядок тексту — це таблична величина, то для нього будуть справедливі більшість функцій та алгоритмів, які ми вивчали раніше. Наприклад, ми можемо отримувати зріз або звернутися до окремого символу рядка тексту. Це робиться так, як і зі списком:

```
>>> a="Hello, World!"
>>> a[:5]
'Hello'
>>> a[7:12]
'World'
>>> a[5]
','
```

Також із вже відомих нам функцій є функція *len()*, яка повертає кількість елементів у тексті:

```
>>> a="Hello, World!"
>>> len(a)
13
```

Чи пам'ятаєте ви метод *split()*, яким ми користувалися для введення списку значень? А за допомогою даного методу ми можемо розбити рядок тексту за певним розділовим знаком. Наприклад:

```
>>> a="слово1 слово2 слово3"
>>> a=a.split()
>>> a
['слово1', 'слово2', 'слово3']
>>> a="слово1, слово2, слово3"
>>> a=a.split(", ")
>>> a
['слово1', 'слово2', 'слово3']
```

Коротко про інші методи для обробки тексту дивіться в таблиці 2.

Метод	Призначення
рядок_тексту.**isdigit()**	Для перевірки: чи складається рядок тексту тільки з цифр. Відповідь у вигляді *True/False*.
рядок_тексту.**isalpha()**	Для перевірки: чи складається рядок тексту тільки з символів. Відповідь у вигляді *True/False*.

рядок_тексту.**isalnum()**	Для перевірки: чи складається рядок тексту з цифр та символів. Відповідь у вигляді *True/False*.
рядок_тексту.**islower()**	Для перевірки: чи складається рядок тексту з символів у нижньому регістрі. Відповідь у вигляді *True/False*.
рядок_тексту.**isupper()**	Для перевірки: чи складається рядок тексту з символів у верхньому регістрі. Відповідь у вигляді *True/False*.
рядок_тексту.**istitle()**	Для перевірки: чи починається кожне слово рядка з великої літери. Відповідь у вигляді *True/False*.
рядок_тексту.**upper()**	Переведення рядка тексту до верхнього регістру.
рядок_тексту.**lower()**	Переведення рядка тексту до нижнього регістру.
розділовий_знак.**join**(*список*)	Для збірки тексту зі списку з розділовим знаком. Метод обернений до *split()*. **Зверніть увагу!** Всі елементи списку повинні бути текстовими фрагментами.
рядок_тексту.**capitalize()**	Переведення першого символу до верхнього регістру, а всі інші до нижнього.
рядок_тексту.**lstrip()**	Видалення пробілів на початку рядка.
рядок_тексту.**rstrip()**	Видалення пробілів в кінці рядка.
рядок_тексту.**strip()**	Видалення пробілів на початку та в кінці рядка.
рядок_тексту.**swapcase()**	Для заміни регістру.
рядок_тексту.**title()**	Переведення першої літери кожного слова до верхнього регістру.

Табл. 2

Рядок тексту відноситься до незмінних послідовностей, як і кортежі, тому ми можемо тільки створювати та доповнювати його, і отримувати дані:

```
>>> a="Hello, World!"
>>> a[3]="L"
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
```

```
    a[3]="L"
TypeError: 'str' object does not support item assignment
>>> del a[5]
Traceback (most recent call last):
  File "<pyshell#4>", line 1, in <module>
    del a[5]
TypeError: 'str' object doesn't support item deletion
```

Давайте розглянемо приклад простої задачі. На вході маємо слово, в будь-якому регістрі та будь-якою мовою. Програма аналізує слово та перевіряє, чи може це слово однаково читатися як зліва на право, так і справа на ліво. Відповідь ми отримаємо у вигляді *True/False*.

Здавалося, що це задача в одну дію, але це не так. Спочатку ми повинні перевести всі символи до одного регістру та видалити непотрібні пробіли на початку та в кінці, які користувач може випадково ввести, а потім вивести результат перевірки, тобто:

```
a=str(input("Введіть слово: "))
# видалення зайвих пробілів
a=a.strip()
# переведення символів до нижнього регістру
a=a.lower()
print(a==a[::-1])
```

Запустимо нашу програму та для приклад: введемо слова "дід" та "баба" (мал. 6).

```
======================
Введіть слово: дід
True
>>>
======================
Введіть слово: баба
False
>>>
======================
Введіть слово:  ДІд
True
>>>
```

Мал. 6

Для кращого засвоєння матеріалу створимо більш складну програму. На вході маємо те саме слово, а на виході – два слова, утворені з нього, які складаються: перше — лише з літер верхнього регістру, а друге — лише нижнього (мал. 7).

Для розв'язання даної задачі ми створимо два нових порожніх рядки тексту та запустимо цикл *for* для кожного символу введеного слова, далі – перевірка регістру даного символу і, в залежності від результату, цей символ додається до порожнього рядка:

```
a=str(input("Введіть слово: "))
```

```
# видалення зайвих пробілів
a=a.strip()
b=""
c=""
for i in a:
    if i.isupper():
        b=b+i
    else:
        c=c+i
print(b)
print(c)
```

```
========================
Введіть слово: СлоВо
СВ
лоо
>>>
```

Мал. 7

Контрольні запитання

1. Яке призначення функції *list()*?
2. Чи відноситься рядок тексту до табличних величин?
3. Текст – це змінна або незмінна величина?
4. Які основні операції можна виконувати з рядком тексу як табличною величиною?
5. Назвіть основні методи для обробки текстових величин.

Практичні завдання

Командний інтерфейс

1. Створіть нову програму мовою Python. На вході маємо ім'я, прізвище та по батькові користувача саме в такому порядку і в один рядок. На виході маємо текстовий рядок з даними користувача, але в іншому порядку: прізвище, ім'я та по-батькові.

2. Створіть нову програму мовою Python. На вході маємо два слова, записаних в один рядок. Якщо перше слово є оберненим до другого, то на виході матимемо *True*, а якщо ні – *False*. Регістр літер в словах не має значення.

3. Створіть нову програму мовою Python. Користувач вводить будь-які слова в один рядок через крапку. Задача програми вивести цей самий рядок, слова якого розділені крапкою, але кожне слово в зворотному порядку.

4. Створіть нову програму мовою Python. На вході маємо два слова. Програма повинна вивести спільні літери в цих словах.