Лекція 4. Обробка строкових даних

Лектор – доцент, кандидат фізико-математичних наук
Кириченко Віктор Вікторович

ПЛАН ЛЕКЦІЇ

1. Основні методи для роботи з рядками
2. Регулярні вирази

Основні методи для роботи з рядками

`.lower()`

Перетворює всі символи в рядку в нижній регістр.

```
text = "HELLO World"
print(text.lower())  # hello world
```

`.upper()`

Перетворює всі символи в рядку в верхній регістр.

```
text = "hello world"
print(text.upper())  # HELLO WORLD
```

`.capitalize()`

Перетворює перший символ на великий, решта на малі.

```
text = "hello world"
print(text.capitalize())  # Hello world
```

`.title()`

Перетворює перший символ кожного слова на великий.

```
text = "hello world from python"
print(text.title())  # Hello World From Python
```

`.swapcase()`

Змінює регістр кожного символу: великі на малі і навпаки.

```
text = "Hello World"
print(text.swapcase())  # hELLO wORLD
```

Основні методи для роботи з рядками

`.strip()`

Видаляє пробіли на початку та в кінці рядка.

```
text = "  Hello World  "
print(text.strip())  # Hello World
```

`.lstrip()`

Видаляє пробіли лише зліва.

```
text = "  Hello World"
print(text.lstrip())  # Hello World
```

`.rstrip()`

Видаляє пробіли лише справа.

```
text = "Hello World  "
print(text.rstrip())  # Hello World
```

`.replace(old, new)`

Замінює всі входження підрядка `old` на підрядок `new`.

```
text = "hello world"
print(text.replace("world", "python"))  # hello python
```

`.split()`

Розбиває рядок на список за заданим роздільником (за замовчуванням — пробіл).

```
text = "hello world python"
print(text.split())  # ['hello', 'world', 'python']
```

`.join(iterable)`

Об'єднує елементи з ітерабельного об'єкта в єдиний рядок, використовуючи рядок, на якому викликається метод, як роздільник.

```
words = ["hello", "world", "from", "python"]
print(" ".join(words))  # hello world from python
```

Основні методи для роботи з рядками

`.find(sub)`

Повертає індекс першого входження підрядка `sub` або `-1`, якщо підрядок не знайдено.

```
text = "hello world"
print(text.find("world"))  # 6
print(text.find("python"))  # -1
```

`.rfind(sub)`

Повертає індекс останнього входження підрядка `sub` або `-1`, якщо підрядок не знайдено.

```
text = "hello world world"
print(text.rfind("world"))  # 12
```

`.index(sub)`

Повертає індекс першого входження підрядка `sub`. Якщо підрядок не знайдено, викидає помилку.

```
text = "hello world"
print(text.index("world"))  # 6
```

`.count(sub)`

Повертає кількість входжень підрядка `sub` в рядок.

```
text = "hello world world"
print(text.count("world"))  # 2
```

`.startswith(prefix)`

Перевіряє, чи починається рядок з підрядка `prefix`.

```
text = "hello world"
print(text.startswith("hello"))  # True
```

`.endswith(suffix)`

Перевіряє, чи закінчується рядок на підрядок `suffix`.

```
text = "hello world"
print(text.endswith("world"))  # True
```

Основні методи для роботи з рядками

`.format()`

Форматує рядок за допомогою позиційних або ключових аргументів.

```
name = "John"
age = 30
text = "My name is {} and I am {} years
old.".format(name, age)
print(text)  # My name is John and I am 30
years old.
```

f-рядки (f-strings)

Форматування рядків за допомогою змінних без необхідності виклику `.format()`.

```
name = "John"
age = 30
text = f"My name is {name} and I am {age}
years old."
print(text)  # My name is John and I am 30
years old.
```

Основні методи для роботи з рядками

`.isalpha()`

Перевіряє, чи складається рядок тільки з літер.

```
text = "hello"
print(text.isalpha())  # True
```

`.isdigit()`

Перевіряє, чи складається рядок тільки з цифр.

```
text = "12345"
print(text.isdigit())  # True
```

`.isspace()`

Перевіряє, чи складається рядок тільки з пробілів.

```
text = "   "
print(text.isspace())  # True
```

`.isupper()`

Перевіряє, чи всі символи в рядку великі.

```
text = "HELLO"
print(text.isupper())  # True
```

`.islower()`

Перевіряє, чи всі символи в рядку малі.

```
text = "hello"
print(text.islower())  # True
```

Регулярні вирази

Регулярні вирази (Regular Expressions, RegEx) — це шаблони для пошуку і маніпуляції з текстом. Вони дозволяють описати закономірності в тексті за допомогою спеціального синтаксису.

У Python підтримка регулярних виразів надається стандартним модулем `re`.

Регулярні вирази

Символ	Опис
`.`	Будь-який символ, крім нового рядка
`^`	Початок рядка
`$`	Кінець рядка
`*`	0 або більше входжень
`+`	1 або більше входжень
`?`	0 або 1 входження
`{n}`	Рівно `n` входжень
`{n,m}`	Від `n` до `m` входжень
`[]`	Будь-який символ із вказаного діапазону
`	`	`
`()`	Групування виразів
`\`	Екранування спецсимволів

Регулярні вирази

Шаблон	Опис
`\d`	Цифра (0–9)
`\D`	Не цифра
`\w`	Словесний символ (латиниця, цифра або _)
`\W`	Все, крім `\w`
`\s`	Пробіл
`\S`	Все, крім пробілу
`\b`	Границя слова
`\B`	Не границя слова

Регулярні вирази

Приклади регулярних виразів

- email `\b[\w.-]+@[\w.-]+\.\w{2,4}\b`
- дати `\d{4}-\d{2}-\d{2}`
- телефон `\+?\d{10,13}`

Імпорт модуля

```
import re
```

Приклади:

```
re.findall(r"\d+", "Ціни: 100, 200, 300")  # ['100', '200', '300']

re.sub(r"\d+", "#", "Мій номер 12345")  # 'Мій номер #'

re.split(r"\s+", "Python є крутий")  # ['Python', 'є', 'крутий']
```

ДЯКУЮ ЗА УВАГУ