Строки и форматирование

Строки, или же текст - один из основных типов в анализе данных. В разделе *"Типы данных"* мы познакомились с простейшими операциями над строками:

- Объявление строк
- Преобразование чисел в строки
- Сложение строк (или же **конкатенация**)
- Умножение строки на целое число

Два важных принципа работы со строками:

- Строки - **неизменяемый** тип данных: в уже созданной переменной типа `str` нельзя изменить символ по отдельности.
- Строки - это списки символов. Практически все методы у типа `list` и `str` совпадают.

Операции над строками

Получить длину строки

По аналогии со списком, у строки можно получить длину, или количество символов в строке:

```
text = "Hello, world!"

print(len(text))      # выведет 13
```

Получить символ или диапазон (подстроку)

При помощи оператора `[]` можно получить как отдельный символ в строке, так и подстроку:

```
text = "ayo what the dog doin"

print(text[0])    # первый символ в строке           выведет `a`
print(text[4])    # 4-й символ в строке              выведет ` ` (пробел)
print(text[-1])   # последний символ в строке        выведет `n`

print(text[4:9:]) # строка с 4-й по 9-й символы      выведет `what`
print(text[:4:])  # строка с первого по 4-й символы  выведет `ayo`
```

```
print(text[::-1]) # строка в обратном порядке         выведет `niod god eht tahw oya
print(text[::5])  # каждый 5-й элемент в строке       выведет `ahhgn`
```

Преобразование между заглавными и строчными буквами

Строку можно преобразовать как в заглавные, так и строчные буквы. Для этого используются методы `upper()` и `lower()` соответственно:

```
text = "A QUIck BrOwN FoX jumpED oVER LAzY doG!"

print(text.upper())           # выведет A QUICK BROWN FOX JUMPED OVER LAZY DOG!
print(text.lower())           # выведет a quick brown fox jumped over lazy dog!
```

Проверить, начинается/заканчивается строка на определенный символ/строку

Для того, чтобы узнать, начинается/заканчивается строка на определенный символ или другую строку, даны методы `startswith(...)` и `endswith(...)` соответственно:

```
text = "Hello, World!"

print(text.startswith("H"))          # выведет True
print(text.startswith("Hello"))      # выведет True
print(text.startswith("world"))      # выведет False
print(text.endswith("!"))            # выведет True
print(text.endswith("cat"))          # выведет False
```

Разделить строку на список других строк по определенному символу

Для этого используется метод `split(...)`.

> Уточнение: метод `split()` можно вызвать и без аргументов. Это эквивалентно вызову метода `split(" ")` с аргументом "пробел":
> `text.split()` ⟷ `text.split(" ")`

```
text = "cat dog fox cuckoo"
animals = text.split()      # разделяет строку `text` по пробелам и формирует список

for animal in animals:
    print(animal)
```

Вывод программы:

```
cat
dog
fox
cuckoo
```

Еще один пример с разделением по строке:

```
text = "drifting;! in;! the;! ocean;! all;! alone"
print(text.split(";! "))
```

Вывод программы:

```
['drifting', 'in', 'the', 'ocean', 'all', 'alone']
```

Подсчитать количество символов в строке

По аналогии со списком для этого можно использовать метод `count(...)`:

```
txt = "Hello world!"

print(txt.count("l")) # выведет 3
print(txt.count("q")) # выведет 0
```

Частный случай: содержится ли определенный символ в строке:

```
txt = "Hello world"

print("l" in txt)     # выведет True
print("q" in txt)     # выведет False

if "l" in txt:
    print(f"{txt} contains symbol l")   # выведет `Hello world contains symbol l`
```