Теоретический материал к занятию Строки и их методы

Строковый тип данных, в Python представленный типом str, один из наиболее часто используемых в программировании. В числе инструментов работы с ним — индексация и срезы.
Давайте рассмотрим основные возможности при работе со строками

1.Перебор строки

```
word = ‘hello’
for char in word:
   print(char)
```

2. Конкатенация (объединение) строк

```
word = ‘hello’
print(word + ‘ hello’)
```

3. Нахождение длины строки

```
word = ‘hello’
print(len(word))
```

4. Проверка вхождения в строку

```
word = ‘pencil’
if ‘pen’ in word:
   print(‘Yes’)
```

5. Умножение строки на число

```
print(‘hello’ * 3)
```

Строки состоят из символов. Очень часто нам надо обратиться к конкретному символу в строке. Для этого в Python используются квадратные скобки [], в которых указывается индекс (номер) нужного символа в строке. Такая операция обращения по определенному индексу называется **индексацией**.
Рассмотрим строку:

```
s = 'Python'
```

Несложно предположить, что нумерация символов начинается с 0.

Выражение	Результат	Пояснение
s[0]	P	первый символ строки
s[1]	y	второй символ строки
s[2]	t	третий символ строки
s[3]	h	четвертый символ строки
s[4]	o	пятый символ строки
s[5]	n	шестой символ строки

В Python в отличии от других языков индекс может быть отрицательным. Зная это можно одинаково легко обращаться к последнему, предпоследнему и т.д. символу строки не зная даже ее длины.

Выражение	Результат	Пояснение
s[-6]	P	первый символ строки
s[-5]	y	второй символ строки
s[-4]	t	третий символ строки
s[-3]	h	четвертый символ строки
s[-2]	o	пятый символ строки
s[-1]	n	шестой символ строки

Иногда нужно работать с целыми частями строки, в таком случае мы используем срезы (slices). С помощью среза мы можем получить несколько символов исходной строки, создав диапазон индексов разделенных двоеточием s[x:y].

Рассмотрим строку `s = 'abcdefghij'`.

Индексы	0	1	2	3	4	5	6	7	8	9
Строка	a	b	c	d	e	f	g	h	i	j
Индексы	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

Следующий код:

```
s = 'abcdefghij'
print(s[2:5])
print(s[0:6])
print(s[2:7])
```

Выводит:

```
cde
abcdef
cdefg
```

Мы можем использовать отрицательные индексы для создания срезов. При использовании отрицательных индексов первый параметр среза должен быть меньше второго, либо должен быть пропущен. Следующий код

```
s = 'abcdefghij'
print(s[-9:-4])
print(s[-3:])
print(s[:-3])
```

выводит

```
bcdef
hij
abcdefg
```

В Python методы строкового типа можно разделить на три группы:

- Конвертация регистра;
- Поиск и замена;
- Классификация символов.

Методы конвертации регистра. Методы в этой группе выполняют преобразование регистра для строк:

- **capitalize** *[к´эпиталайз] - сделать заглавным, от capital - заглавный, первый, ..*

```
word = 'hello world'
print(word.capitalize())
```

Результатом выполнения такой программы будет

Hello world

- **swapcase** *[су´опкэйс] - сменить регистр, от swap - менять, case - регистр, падеж, случай..*

```
word = 'hello world'
print(word.swapcase())
```

Результатом выполнения такой программы будет

HELLO WORLD

- **title** *[т´айтл] - заголовок, титул*

```
word = 'hello world'
print(word.title())
```

Результатом выполнения такой программы будет

Hello World

- **lower** *[л´оуэ] - нижний*

```
word = 'HELLO world'
print(word.lower())
```

Результатом выполнения такой программы будет

hello world

- **upper** *[´апэ] - верхний*

```
word = 'HELLO world'
print(word.upper())
```

Результатом выполнения такой программы будет

HELLO WORLD

Методы поиска и замены. Методы этой группы — средства поиска и замены строк внутри других строк:

- **count** *[к´аунт] - считать*

```
word = 'hello world'
print(word.count('l')
```

Результатом выполнения такой программы будет

3

- **startswith** *[ст´артсуиð] starts - начинается, with - с*

```
word = 'hello world'
print(word.startswith('he'))
```

Результатом выполнения такой программы будет

True

- **endswith** *[´эндсуиð] ends - кончается, with - с*

```
word = 'hello world'
print(word.endswith('rld'))
```

Результатом выполнения такой программы будет

True

- **find, rfind** *[ф´айнд] находить, искать*

```
word = 'hello world'
print(word.find('o'))
```

Результатом выполнения такой программы будет

4

- **index, rindex** *[´индекс]*

```
word = 'hello world'
print(word.index('o'))
```

Результатом выполнения такой программы будет

4

- **strip, lstrip, rstrip** *[стр´ип] обрывать*

```
word = '   hello world   '
```

```
print(word.strip())
```

Результатом выполнения такой программы будет

```
hello world
```

- **replace** *[рипл´эйс] замещать*

```
word = 'hello world'
print(word.replace('w', 'wooo'))
```

Результатом выполнения такой программы будет

```
hello woooorld
```