

# Шпаргалка «Работа со строками»

## split() — разделение строк

Преобразуем строку в список; в скобках после `split` указываем, какой разделитель использовать для разбиения:

```
blok_string = 'Ночь. Улица. Фонарь. Аптека'
blok_list = blok_string.split(' ') # Разбиваем по точке и пробелу.

# Будет напечатано: ['Ночь', 'Улица', 'Фонарь', 'Аптека']
# Разделитель удаляется из результата.
# Точки и пробела в элементах списка нет.
```

Если надо преобразовать строку в список, разделив её по пробелам — в скобках после `split` можно ничего не указывать: пробел — это значение по умолчанию.

```
blok_string = 'Ночь. Улица. Фонарь. Аптека'
blok_list = blok_string.split() # В скобках ничего не указываем.

# Будет напечатано: ['Ночь.', 'Улица.', 'Фонарь.', 'Аптека']
# Точки сохранились, пробелы удалены,
# ведь мы разделяли строку на элементы по пробелам.
```

## Получить элемент из списка по его индексу

Последнее слово можно получить в списке не только по индексу `len(blok_list) - 1`, но и проще, по индексу `-1`: отрицательные индексы отсчитываются от конца коллекции. Они устроены вот так (индексы сверху и снизу — эквивалентны):

0	1	2	3
['Ночь', 'Улица', 'Фонарь', 'Аптека']			
-4	-3	-2	-1

## f-строки

Если перед началом строки (перед открывающей кавычкой) поставить литеру `f`, то прямо в строку можно встраивать переменные (обозначая их фигурными скобками). Значения переменных будут автоматически конвертированы в строку.

```
one_hundred = 100
rubles = 'рублей'
friends = 'друзей'

print(f'Не имей {one_hundred} {rubles}, а имей {one_hundred} {friends}.')
# Будет напечатано: Не имей 100 рублей, а имей 100 друзей.
# Для создания строки не потребовалось даже превращать
# значение переменной one_hundred из числа в строку.
# Всё само собой сделалось!
```

В f-строки можно подставлять не только переменные, но и результаты вычислений:

```
one_hundred = 100
five_hundred = 500

print(f'{one_hundred} + {five_hundred} = {one_hundred + five_hundred}')
# Будет напечатано: 100 + 500 = 600
```

В f-строке можно обратиться к элементам списка:

```
ruussian_alphabet = ['а', 'б', 'в', 'г', 'д', 'е', 'ё', 'ж', 'з', 'и', 'й', 'к', 'л',
                    'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э',
                    'ю', 'я']

print(f'{ruussian_alphabet[-1]} — последняя буква в алфавите.')
```

Обратиться к элементам словаря по ключу тоже можно прямо из f-строки:

```
favorite_songs = {
    'Тополиный пух': 'Иванушки international',
    'Город золотой': 'Аквариум',
    'Звезда по имени Солнце': 'Кино'
}

song = 'Город золотой'
print(f'{song} — одна из известных песен группы {favorite_songs[song]}')
```

## Метод join(): объединение элементов списка в строку

Это как `split()` наоборот: метод `join()` «склеивает» элементы списка в строку. В получившейся строке между элементами исходного списка можно добавить какой-нибудь разделитель — текстовый символ или набор символов. Важное отличие `join()`: этот метод применяется к разделителю, а список, который надо превратить в строку, передаётся в аргумент этого метода.

```
words_list = ['раз', 'два', 'три', 'четыре', 'пять', 'вышел', 'зайчик',
              'погулять']
# Для разделения применим дефис:
new_string = '-'.join(words_list)

print(new_string)
# Будет напечатано: раз-два-три-четыре-пять-вышел-зайчик-погулять
```