stepik 9.4 Методы строк. Часть 2 15 из 15 шагов пройдено 46 из 46 баллов получено "Поколение Python": курс для начинающих Прогресс по курсу: 2128/2210 Тема урока: строки 8.1 Часть 1 8.2 Часть 2 1. Методы count() 9 Строковый тип данных 2. Meтод startswith() 3. Meтод endswith() 9.1 Индексация 4. Meтод find() 9.2 Срезы 5. Meтод rfind() 6. Meтод index() 9.3 Методы строк. Часть 1 7. Метод rindex() 9.4 Методы строк. Часть 2 8. Метод strip() 9. Метод lstrip() 9.5 Методы строк. Часть 3 10. Метод rstrip() 9.6 Форматирование строк 11. Метод replace() 9.7 Строки в памяти комп... Аннотация. Строковый тип данных, основные методы поиска и замены. 9.8 Сравнение строк Поиск и замена Экзамен 10 Итоговая раб... Методы поиска и замены строк внутри других строк. 10.1 Часть 1 \Diamond Каждый метод в этой группе поддерживает необязательные аргументы <start> и <end> . Как и в строковых срезах, действие метода ограничено частью исходной строки, начинающейся с позиции символа <start> и продолжающейся вплоть до позиции

символа <end> , но не включающей её. Если параметр <start> указан, а параметр <end> нет, то метод применяется к части исходной строки от <start> до конца строки. Если параметры не заданы, то подразумевается, что <start> = 0 , <end> = len(s) . Meтод count()

Metod count(<sub>, <start>, <end>) считает количество непересекающихся вхождений подстроки <sub> в исходную строку s .

Приведённый ниже код:

s = 'foo goo moo' print(s.count('oo'))

```
print(s.count('oo', 0, 8)) # подсчет с 0 по 7 символ
выводит:
```

```
2
                                                Метод startswith()
```

возвращает значение True, если исходная строка начинается с подстроки <prefix>, или False в противном случае.

Приведённый ниже код:

Meтод startswith(<prefix>, <start>, <end>) определяет, начинается ли исходная строка s подстрокой <prefix>. Метод

```
print(s.startswith('foo'))
 print(s.startswith('baz'))
выводит:
```

Метод endswith()

Meтод endswith(<suffix>, <start>, <end>) определяет, оканчивается ли исходная строка s подстрокой <suffix>. Метод

s = 'foobar'

True

True

False

False

```
возвращает значение True, если исходная строка оканчивается на подстроку <suffix>, или False в противном случае.
Приведённый ниже код:
```

s = 'foobar' print(s.endswith('bar')) print(s.endswith('baz'))

```
выводит:
```

Mетоды find(), rfind()

Meтод find(<sub>, <start>, <end>) находит индекс первого вхождения подстроки <sub> в исходной строке s . Если строка s не содержит подстроки <sub>, то метод возвращает значение -1. Мы можем использовать данный метод наравне с оператором

in для проверки: содержит ли заданная строка некоторую подстроку или нет. Приведённый ниже код:

print(s.find('foo'))

-1

s = 'foo bar foo baz foo qux'

```
print(s.find('bar'))
 print(s.find('qu'))
 print(s.find('python'))
выводит:
 0
```

вхождение подстроки <sub>, начиная с конца строки s.

Mетоды index(), rindex()

Meтод rfind(<sub>, <start>, <end>) идентичен методу find(<sub>, <start>, <end>), за тем исключением, что он ищет первое

Meтoд index(<sub>, <start>, <end>) идентичен методу find(<sub>, <start>, <end>), за тем исключением, что он вызывает

ошибку ValueError: substring not found во время выполнения программы, если подстрока <sub> не найдена. Meтoд rindex(<sub>, <start>, <end>) идентичен методу index(<sub>, <start>, <end>), за тем исключением, что он ищет

Meтоды find() и rfind() являются более безопасными, чем index() и rindex(), так как не приводят к возникновению ошибки во время выполнения программы.

Метод strip()

Метод Istrip()

Приведённый ниже код:

первое вхождение подстроки <sub>, начиная с конца строки s .

```
foo bar foo baz foo qux
print(s.strip())
```

Meтод strip() возвращает копию строки s , у которой удалены все пробелы, стоящие в начале и конце строки.

foo bar foo baz foo qux

выводит:

Meтод lstrip() возвращает копию строки s , у которой удалены все пробелы, стоящие в начале строки. Приведённый ниже код:

foo bar foo baz foo qux print(s.lstrip())

```
выводит (символом _ обозначены пробелы):
```

foo bar foo baz foo qux

Метод rstrip() возвращает копию строки s , у которой удалены все пробелы, стоящие в конце строки.

Метод rstrip()

Приведённый ниже код: foo bar foo baz foo qux

print(s.rstrip())

```
выводит (символом _ обозначены пробелы):
   ____foo bar foo baz foo qux
```

```
Метод replace(<old>, <new>) возвращает копию s со всеми вхождениями подстроки <old>, заменёнными на <new>.
```

Метод replace()

s = 'foo bar foo baz foo qux' print(s.replace('foo', 'grault'))

выводит:

Приведённый ниже код:

```
grault bar grault baz grault qux
```

Meтод replace() может принимать необязательный третий аргумент <count>, который определяет количество замен. Приведённый ниже код:

s = 'foo bar foo baz foo qux'

```
print(s.replace('foo', 'grault', 2))
выводит:
```

Примечание. Методы strip(), lstrip(), rstrip() могут принимать на вход необязательный аргумент <chars>.

Heoбязательный аргумент <chars> – это строка, которая определяет набор символов для удаления.

grault bar grault baz foo qux

9833

отдельный форум.

1055

Шаг 1

Примечания

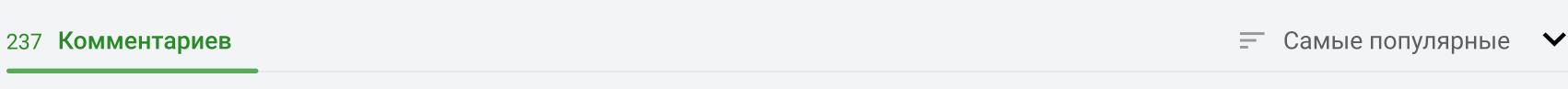
Приведённый ниже код: s = '-+-+abc+-+-'

print(s.strip('+-')) print(s.rstrip('+-')) print(s.lstrip('+-'))

```
выводит:
 abc
 -+-+abc
 abc+-+-
```

```
Следующий шаг 🗦
```

W Happy Pythoning!



Будьте вежливы и соблюдайте наши принципы сообщества. Пожалуйста, не оставляйте решения и подсказки в комментариях, для этого есть

```
Оставить комментарий
```

```
Показать обсуждения (237)
```