IT CAREER HUB

Python

Методы строк

Преподаватель

Портрет

Имя Фамилия

Текущая должность

Количество лет опыта

Какой у Вас опыт - ключевые кейсы

Самые яркие проекты

Дополнительная информация по вашему усмотрению

Корпоративный e-mail

Социальные сети (по желанию)

Важно

- Камера должна быть включена на протяжении всего занятия
- В течение занятия вопросы задавать в чате или когда преподаватель спрашивает, есть ли у Вас вопросы
- Вести себя уважительно и этично по отношению к остальным участникам занятия
- Организационные вопросы по обучению решаются с кураторами, а не на тематических занятиях
- Во время занятия будут интерактивные задания, будьте готовы включить камеру или демонстрацию экрана по просьбе преподавателя

План занятия

- Итерируемый объект
- Цикл for
- Функция range
- Операторы break, continue, else в цикле for
- Вложенные циклы
- Вложенные циклы с использованием while и for

ICH IT CAREER HUB

ОСНОВНОЙ БЛОК

IT CAREER HUB

Методы строк

Полезно знать

ICH
IT CAREER HUB

Методы строк

Для работы со строками доступен широкий набор встроенных методов. Эти методы позволяют выполнять различные операции.

Полезно знать

ICH
IT CAREER HUB

Методы проверки предиката

Это методы, которые возвращают True или False в зависимости от выполнения определённого условия для строки.

Таблица с предикатными методами

ICH
IT CAREER HUB

Метод	Описание
isalpha()	Проверяет, состоит ли строка только из букв.
isdigit()	Проверяет, состоит ли строка только из цифр.
isalnum()	Проверяет, состоит ли строка только из букв и цифр.
isspace()	Проверяет, состоит ли строка только из пробелов.
islower()	Проверяет, все ли символы строки в нижнем регистре.
isupper()	Проверяет, все ли символы строки в верхнем регистре.
istitle()	Проверяет, начинается ли каждое слово в строке с заглавной буквы.
startswith()	Проверяет, начинается ли строка с указанной подстроки.
endswith()	Проверяет, заканчивается ли строка на указанную подстроку.

Таблица с предикатными методами

Метод	Описание
isdecimal()	Проверяет, состоит ли строка только из десятичных цифр.
isnumeric()	Проверяет, состоит ли строка только из числовых символов.
isascii()	Проверяет, состоит ли строка только из символов ASCII.
isprintable()	Проверяет, все ли символы строки являются печатаемыми.
isidentifier()	Проверяет, является ли строка допустимым идентификатором в Python.

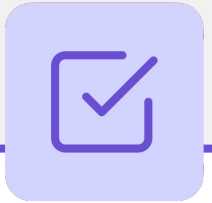
Пример

```
# проверяет, состоит ли строка только из букв (без пробелов и других символов)
print("Hello".isalpha())

# проверяет, состоит ли строка только из цифр
print("12345.1".isdigit())

# проверяет, состоит ли строка только из букв и цифр (без пробелов и специальных символов)
print("Hello123".isalnum())

# проверяет, состоит ли строка только из пробельных символов (пробелы, табуляция и т.д.)
print("   ".isspace())

# проверяет, находятся ли все символы строки в нижнем регистре
print("Hello".islower())
```

Пример

```
# проверяет, находятся ли все символы строки в верхнем регистре
print("HELLO".isupper())

# проверяет, начинается ли каждое слово в строке с заглавной буквы
print("Hello world".istitle())

# проверяет, начинается ли строка с указанной подстроки
print("Hello world".startswith("Hello"))

# проверяет, заканчивается ли строка указанной подстрокой
print("Hello world!".endswith("world"))
```

Методы строк



Также есть и более специфичные методы, такие как проверка на печатаемые символы, десятичные числа или допустимые идентификаторы.

Пример

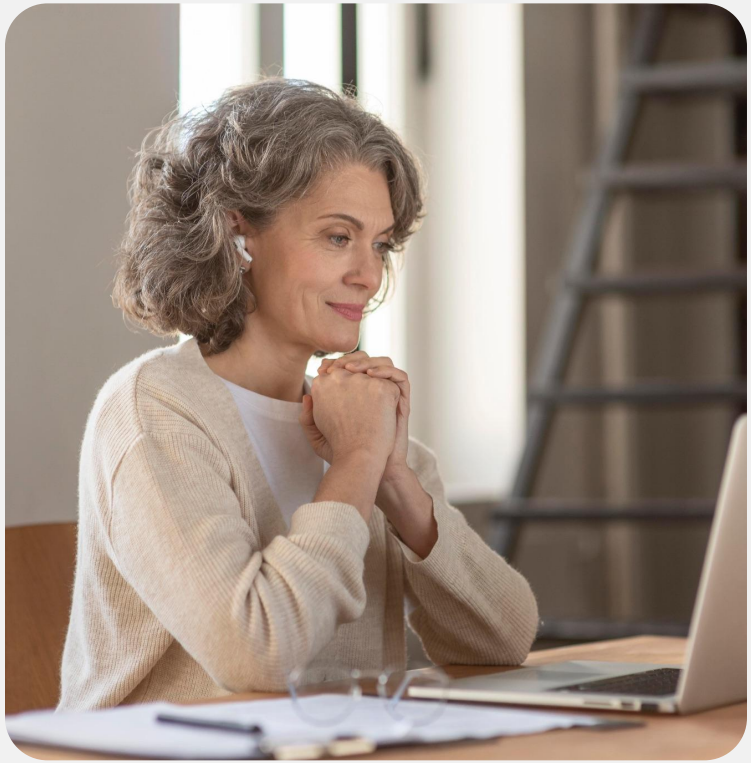
```
# проверяет, состоит ли строка только из десятичных цифр, без символов вроде "²"
print("12345²".isdecimal())

# проверяет, состоит ли строка только из числовых символов, включая дробные числа или надстрочные символы
print("½".isnumeric())

# проверяет, состоит ли строка только из символов ASCII
print("Привет".isascii())

# проверяет, все ли символы в строке являются печатаемыми, например без символа перевода строки
print("Hello\nWorld".isprintable())

# проверяет, является ли строка допустимым идентификатором (названием переменной)
print("1variable".isidentifier())
```

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

ЗАДАНИЕ

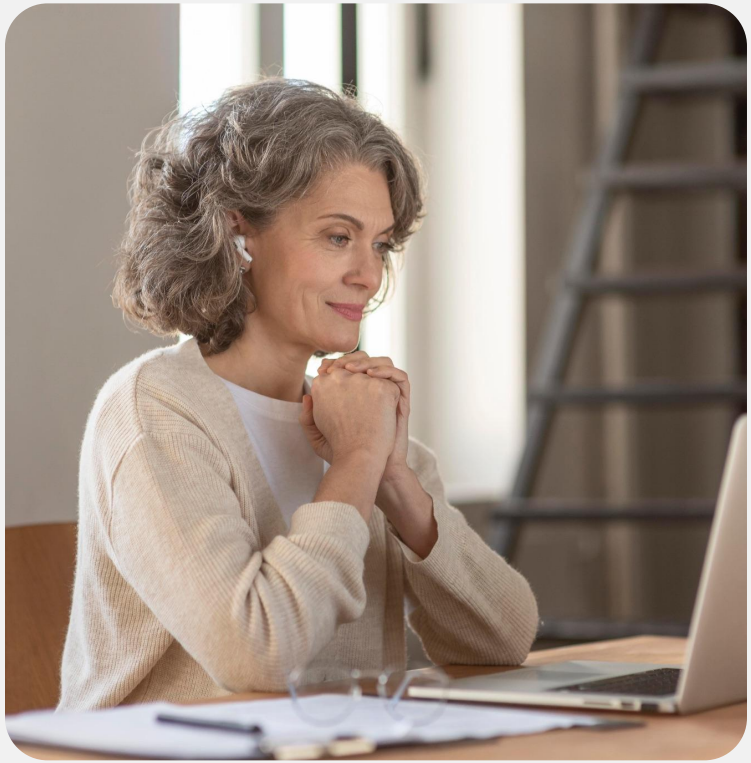


Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
text = "Python3"

print(text.isalpha())
```

a. True
b. False
c. Ошибка
d. None

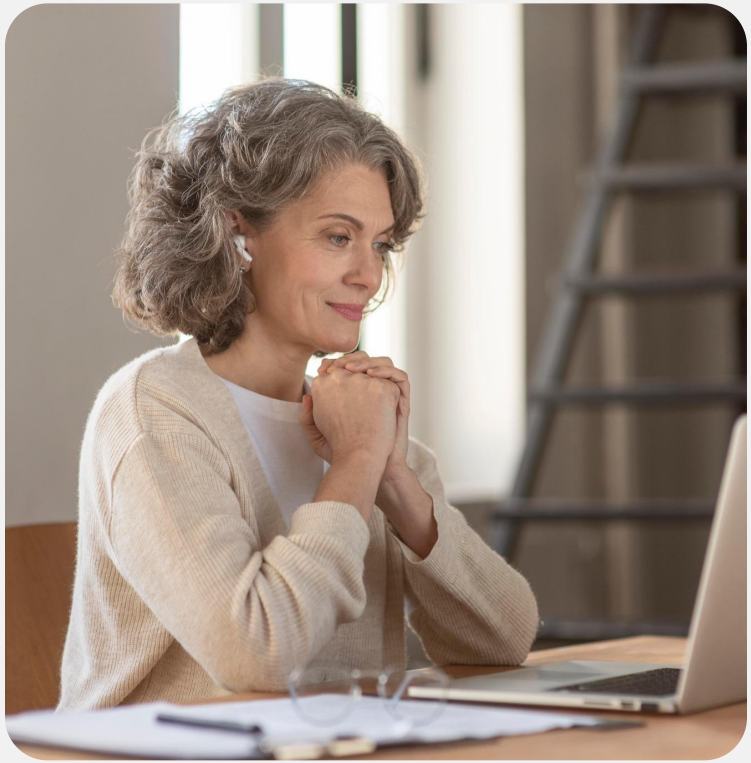


Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
text = "Python3"

print(text.isalpha())
```

a. True
b. False
c. Ошибка
d. None

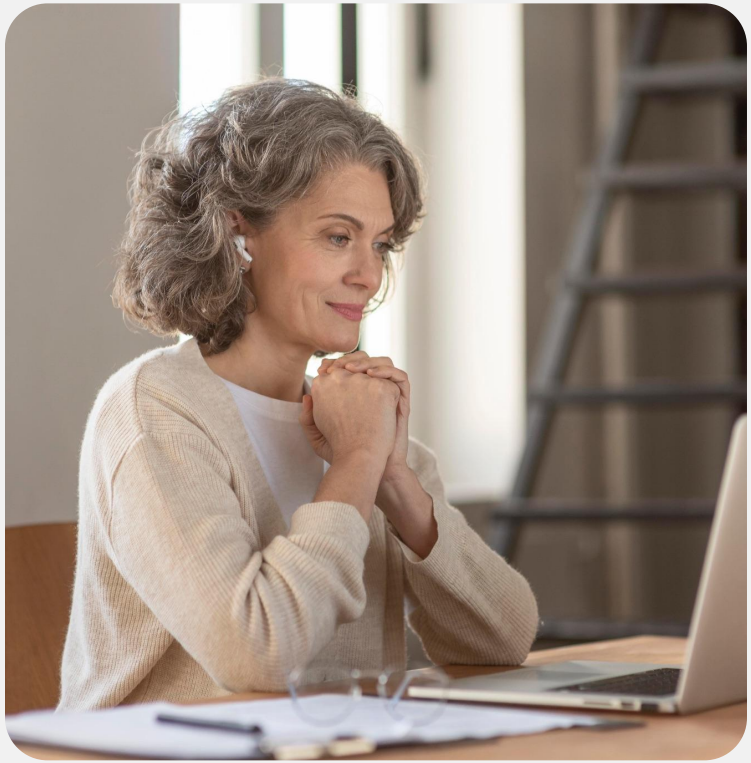


Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
text = "Hello World"

print(text.istitle())
```

a. True
b. False
c. Ошибка
d. None



Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
text = "Hello World"

print(text.istitle())
```

a. True
b. False
c. Ошибка
d. None

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

Поиск подстроки в строке

Поиск подстроки в строке

Python предоставляет несколько способов для поиска подстроки в строке. Эти методы позволяют определить, содержится ли подстрока в строке, и найти её местоположение, если она присутствует, а также подсчитать количество вхождений подстроки.

Таблица с предикатными методами

Метод	Описание
find()	Возвращает индекс первого вхождения подстроки. Если подстрока не найдена, возвращает -1.
rfind()	Возвращает индекс последнего вхождения подстроки. Если подстрока не найдена, возвращает -1.
index()	Возвращает индекс первого вхождения подстроки. Если подстрока не найдена, вызывает ValueError.
rindex()	Возвращает индекс последнего вхождения подстроки. Если подстрока не найдена, вызывает ValueError.
count()	Возвращает количество вхождений подстроки в строке.

Методы find() и rfind()

find():

- Ищет первое вхождение подстроки и возвращает индекс её начала.
- Если подстрока не найдена, возвращает -1.

rfind():

- Ищет последнее вхождение подстроки и возвращает индекс её начала.
- Если подстрока не найдена, также возвращает -1.

Пример

```python
text = "Python is awesome. Python is dynamic."

# Поиск первого вхождения подстроки
print(text.find("Python"))

# Поиск последнего вхождения подстроки
print(text.rfind("Python"))

# Подстрока не найдена
print(text.find("Java"))
```

Методы index() и rindex()

index():

- Работает аналогично find(), возвращает индекс первого вхождения подстроки.
- Если подстрока не найдена, вызывает исключение ValueError.

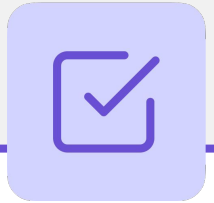
rindex():

- Работает аналогично rfind(), возвращает индекс последнего вхождения подстроки.
- Если подстрока не найдена, также вызывает исключение ValueError.

Пример

```
text = "Python is awesome. Python is dynamic."

# Поиск первого вхождения подстроки
print(text.index("Python"))

# Поиск последнего вхождения подстроки
print(text.rindex("Python"))

# Попытка найти отсутствующую подстроку вызывает ошибку
print(text.index("Java"))  # ValueError
```

Метод count()



Возвращает количество вхождений подстроки в строке, т.к. подсчитывает сколько раз подстрока найдена в строке. Этот метод всегда безопасен и не вызывает исключений.

Пример

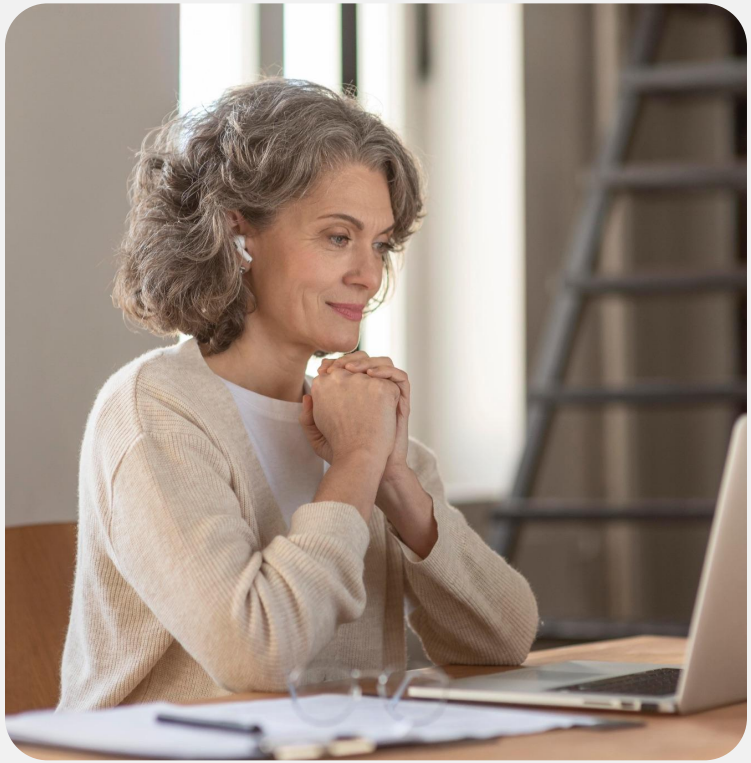
```
# Подсчёт количества вхождений подстроки
print(text.count("Python"))

# Если подстрока не найдена, возвращает 0
print(text.count("Java"))

# Следующую подстроку ищет после конца предыдущей, т.е. без пересечений
text = "hahahahahaha"

print(text.count("ha"))

print(text.count("haha"))
```

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

ЗАДАНИЕ

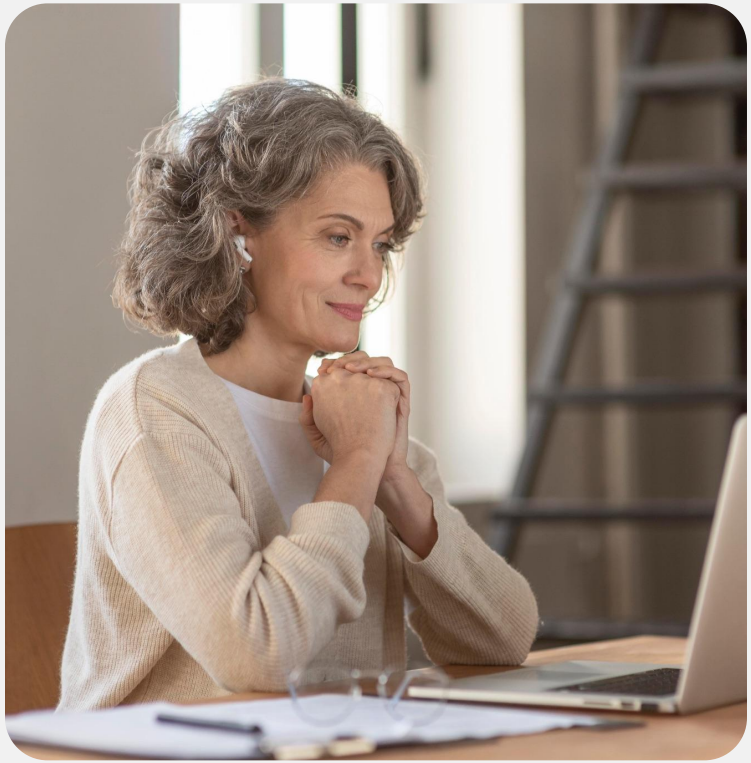


Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
text = "Python is awesome. Python is dynamic."

print(text.find("is"))
```

a. 23
b. 7
c. 8
d. -1

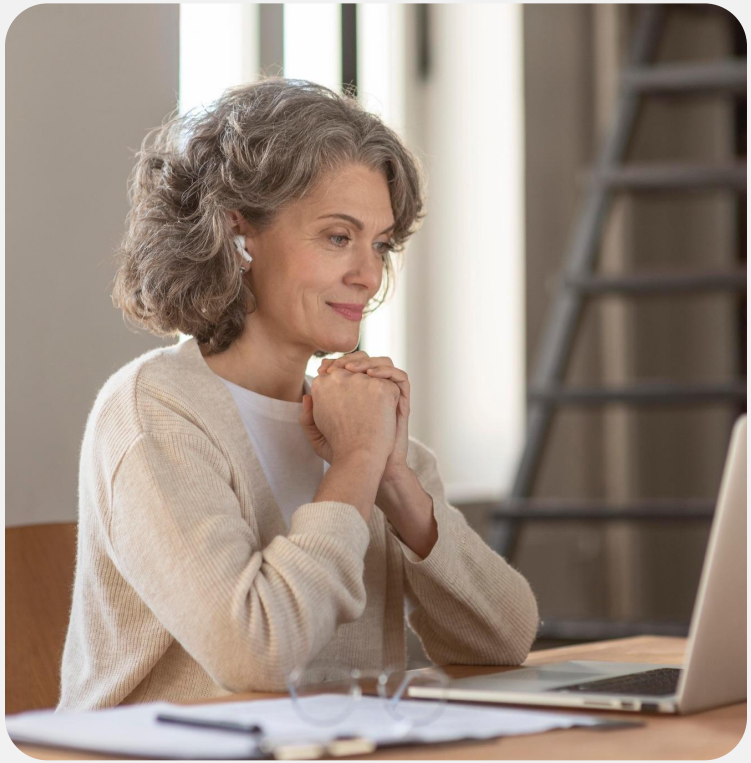


Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
text = "Python is awesome. Python is dynamic."

print(text.find("is"))
```

a. 23
b. 7
c. 8
d. -1

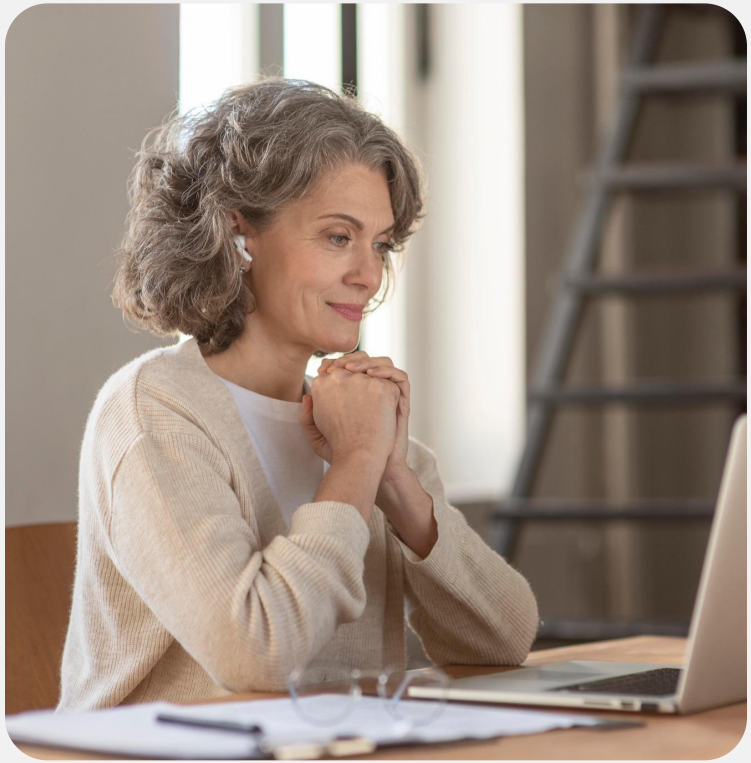


Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
text = "33333"

print(text.count("33"))
```

a. 1
b. 2
c. 3
d. 5



Выберите правильный вариант ответа

Какой результат выведет следующий код?

```
text = "33333"

print(text.count("33"))
```

a. 1
b. 2
c. 3
d. 5

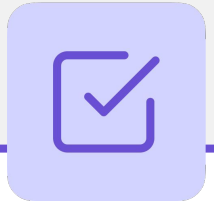
IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

Методы изменения регистра

Методы изменения регистра



В Python существуют несколько методов для работы с регистром строк. Эти методы позволяют преобразовывать символы строки в верхний или нижний регистр, а также обеспечивать правильное форматирование с заглавными буквами в начале каждого слова или строки.

Таблица методов изменения регистра

ICH
IT CAREER HUB

Метод	Описание
upper()	Преобразует все символы строки в верхний регистр.
lower()	Преобразует все символы строки в нижний регистр.
capitalize()	Преобразует первую букву строки в заглавную, остальные символы делает строчными.
title()	Преобразует первую букву каждого слова строки в заглавную.
swapcase()	Меняет регистр всех символов строки на противоположный.
casefold()	Преобразует строку в нижний регистр с учётом языковых особенностей для более точного сравнения.

Пример

```
text = "Hello, worlD!"
# Преобразование строки в верхний регистр
print(text.upper())
# Преобразование строки в нижний регистр
print(text.lower())

# Первая буква строки становится заглавной, остальные строчными
print(text.capitalize())

# Первая буква каждого слова становится заглавной
print(text.title())

# Меняет регистр всех символов на противоположный
print(text.swapcase())

# Приводит строку к нижнему регистру с учётом особенностей языка
print("Straße".casefold())
```

Полезно знать

Метод replace

Используется для замены всех вхождений одной подстроки на другую в строке.

Методы index() и rindex()

Синтаксис

```
string.replace(old, new, count=-1)
```

Разбор

- **old** — подстрока, которую нужно заменить.
- **new** — подстрока, на которую будет заменена старая подстрока.
- **count** — количество замен. Если не указано, заменяются все вхождения.

Пример

```python
# Пример замены всех вхождений

text = "I love Python, Python is great"

new_text = text.replace("Python", "Java")

print(new_text)

# Пример с ограничением замен

text = "apple apple apple"

new_text = text.replace("apple", "orange", 1)

print(new_text)
```

Полезно знать

ICH
IT CAREER HUB

Методы split() и join()

Используются для работы со строками, позволяя легко разделять строку на части и объединять список строк в одну строку с помощью разделителя.

Таблица методов split() и join()

Метод	Описание
split(sep, maxsplit=-1)	Разделяет строку по указанному разделителю и возвращает список строк. Можно ограничить количество разбиений.
join(iterable)	Объединяет элементы последовательности в строку, используя указанный разделитель.

Полезно знать

ICH
IT CAREER HUB

Метод split()

Разделяет строку на части по указанному разделителю и возвращает список этих частей.

Методы index() и rindex()

Синтаксис

```
string.split(sep, maxsplit=-1)
```

Разбор

- **sep** — символ или подстрока, по которой происходит разделение.
- **maxsplit** — максимальное количество разбиений.

Пример

```python
# Пример 1: Разделение строки по пробелам
text = "apple banana cherry"
fruits = text.split()
print(fruits)

# Пример 2: Разделение строки по запятой
text = "apple,banana,cherry"
fruits = text.split(",")
print(fruits)

# Пример 3: Разделение строки с ограничением количества разбиений
text = "apple---banana---cherry"
fruits = text.split("---", 1)
print(fruits)
```

Полезно знать

IT CAREER HUB

Метод split()

Работает по-разному в зависимости от того, указан пробел как разделитель или нет.

string.split():

Синтаксис

```
text = "apple     banana\t cherry\n"

fruits = text.split()    # Разделяет по любым пробельным символам

print(fruits)
```

Разбор

- Если не указать разделитель, метод автоматически разделяет строку по любым пробельным символам.
- Кроме того, несколько подряд идущих пробелов будут восприниматься как один элемент.

string.split(" "):

Синтаксис

```
text = "apple     banana\t cherry\n"

fruits = text.split(" ")  # Разделяет только по пробелам

print(fruits)
```

Разбор

- В этом случае метод разделяет строку только по пробелам.
- Если между словами несколько пробелов, они не будут восприниматься как один элемент, и это в результате создаст пустые строки.

Полезно знать

IT CAREER HUB

Метод join()

Объединяет элементы последовательности (списка, кортежа и т.д.) в одну строку, используя указанный разделитель.

string.split(" "):

Синтаксис

```
separator.join(iterable)
```

Разбор

- **separator** — строка, которая будет использоваться в качестве разделителя между элементами.
- **iterable** — последовательность, элементы которой нужно объединить в строку.

Пример

```python
# Пример 1: Объединение списка строк с пробелом

fruits = ['apple', 'banana', 'cherry']
text = " ".join(fruits)
print(text)

# Пример 2: Объединение списка строк с запятой и пробелом

fruits = ['apple', 'banana', 'cherry']
text = ", ".join(fruits)
print(text)

# Пример 3: Объединение списка строк по пустому символу

litters = ['a', 'b', 'c', 'd', 'e']
text = "".join(litters)
print(text)
```

IT CAREER HUB

ВОПРОСЫ

IT CAREER HUB

Методы strip, lstrip, rstrip

Полезно знать

IT CAREER HUB

Методы strip, lstrip, rstrip

Используются для удаления пробелов и других ненужных символов из строк.

Таблица методов

Метод	Описание
strip()	Удаляет пробелы или указанные символы с обеих сторон строки.
lstrip()	Удаляет пробелы или указанные символы только с левой стороны строки.
rstrip()	Удаляет пробелы или указанные символы только с правой стороны строки.

Полезно знать

IT CAREER HUB

Метод strip()

Удаляет пробелы или указанные символы с обоих концов строки (слева и справа).

string.split(" "):

Синтаксис

```
string.strip([chars])
```

Разбор

- **chars (необязательный)** — символы, которые нужно удалить. Если не указано, по умолчанию удаляются пробелы.

Пример

```
text = "\t  hello world\n  "
print(text.strip())  # Удаляет все пробельные символы


text_with_chars = "***hello***"
print(text_with_chars.strip("*"))  # Удаляет указанный символ


text_with_chars = "**--*hello---***"
print(text_with_chars.strip("*-"))  # Удаляет все указанные символы
```

Полезно знать

ICH
IT CAREER HUB

Методы lstrip() и rstrip()

Удаляют пробелы или указанные символы только с левой или правой стороны строки соответственно.

Пример

```
text = "  hello world  "

print(text.lstrip())  # Вывод: "hello world  "

text_with_chars = "***hello***"

print(text_with_chars.lstrip("*"))  # Вывод: "hello***"

text = "  hello world  "

print(text.rstrip())  # Вывод: "  hello world"

text_with_chars = "***hello***"

print(text_with_chars.rstrip("*"))  # Вывод: "***hello"
```

IT CAREER HUB

ВОПРОСЫ

Домашнее задание

1. **Звёздочки вместо чисел**

   Напишите программу, которая заменяет все цифры в строке на звёздочки `*`.

   ```
   text = "My number is 123-456-789"
   ```

2. **Количество символов**

   Напишите программу, которая подсчитывает количество вхождений всех символов в строке. Нужно вывести только символы, которые встречаются более 1 раза (игнорируя регистр), с указанием их количества. Выводите повторяющиеся символы только один раз.

   text = "Programming in python"

3. **Увеличение чисел**

   Напишите программу, которая заменяет все числа в строке на их эквивалент, умноженный на 10.

   text = "I have 5 apples and 10 oranges, price is 0.5 each. Card number is ....3672."

Заключение