

9.5 Методы строк. Часть 3

Шаг 1

Тема урока: строки

1. Методы строк `isalnum()`
2. Метод `isalpha()`
3. Метод `isdigit()`
4. Метод `islower()`
5. Метод `isupper()`
6. Метод `isspace()`
7. Решение задач

Аннотация. Строковый тип данных, основные методы классификации символов.

Классификация символов

Методы в этой группе классифицируют строку на основе содержащихся в ней символов.

Метод `isalnum()`

Метод `isalnum()` определяет, состоит ли исходная строка из буквенно-цифровых символов. Метод возвращает значение `True`, если исходная строка является непустой и состоит **только** из буквенно-цифровых символов, или `False` в противном случае.

Приведённый ниже код:

```
s1 = 'abc123'  
s2 = 'abc$*123'  
s3 = ''  
  
print(s1.isalnum())  
print(s2.isalnum())  
print(s3.isalnum())
```

ВЫВОДИТ:

```
True  
False  
False
```

Обратите внимание, что метод `isalnum()` возвращает значение `True` даже в том случае, когда строка состоит только из буквенных или только из цифровых символов.

Приведённый ниже код:

```
s1 = 'BEEGEEK'  
s2 = '2202'  
  
print(s1.isalnum())  
print(s2.isalnum())
```

ВЫВОДИТ:

```
True  
True
```

Метод isalpha()

Метод `isalpha()` определяет, состоит ли исходная строка из буквенных символов. Метод возвращает значение `True`, если исходная строка является непустой и состоит **только** из буквенных символов, или `False` в противном случае.

Приведённый ниже код:

```
s1 = 'ABCAbc'  
s2 = 'abc123'  
s3 = ''  
  
print(s1.isalpha())  
print(s2.isalpha())  
print(s3.isalpha())
```

ВЫВОДИТ:

```
True  
False  
False
```

Метод isdigit()

Метод `isdigit()` определяет, состоит ли исходная строка **только** из цифровых символов. Метод возвращает значение `True`, если исходная строка является непустой и состоит **только** из цифровых символов, или `False` в противном случае.

Приведённый ниже код:

```
s1 = '1234567'  
s2 = 'abc123'  
s3 = ''  
  
print(s1.isdigit())  
print(s2.isdigit())  
print(s3.isdigit())
```

ВЫВОДИТ:

```
True  
False  
False
```

Метод islower()

Метод `islower()` определяет, являются ли **все** буквенные символы исходной строки строчными (имеют нижний регистр). Метод возвращает значение `True`, если все буквенные символы исходной строки являются строчными, или `False` в противном случае.

Приведённый ниже код:

```
s1 = 'abc'  
s2 = 'abc1$d'  
s3 = 'Abc1$D'  
  
print(s1.islower())  
print(s2.islower())  
print(s3.islower())
```

ВЫВОДИТ:

```
True  
True  
False
```

Обратите внимание, что метод `islower()` **игнорирует все небуквенные символы**.

Приведённый ниже код:

```
print('1234'.islower())
print('+-*/'.islower())
print('ab#%'.islower())
```

ВЫВОДИТ:

```
False
False
True
```

В первом и втором случаях у нас в строках отсутствуют буквенные символы в нижнем регистре, поэтому метод `islower()` и возвращает значение `False`.

Метод `isupper()`

Метод `isupper()` определяет, являются ли **все** буквенные символы исходной строки заглавными (имеют верхний регистр).

Метод возвращает значение `True`, если все буквенные символы исходной строки являются заглавными, или `False` в противном случае.

Приведённый ниже код:

```
s1 = 'ABC'
s2 = 'ABC1$D'
s3 = 'Abc1$D'

print(s1.isupper())
print(s2.isupper())
print(s3.isupper())
```

ВЫВОДИТ:

```
True
True
False
```

Обратите внимание, что метод `isupper()` **игнорирует все небуквенные символы**.

Приведённый ниже код:

```
print('5678'.isupper())
print('!?!_&'.isupper())
print('AB%$'.isupper())
```

ВЫВОДИТ:

```
False
False
True
```

В первом и втором случаях у нас в строках отсутствуют буквенные символы в верхнем регистре, поэтому метод `isupper()` и возвращает значение `False`.

Метод `isspace()`

Метод `isspace()` определяет, состоит ли исходная строка **только** из пробельных символов. Метод возвращает значение `True`, если строка состоит только из пробельных символов, или `False` в противном случае.

Приведённый ниже код:

```
s1 = ' '  
s2 = 'abc1$d'  
  
print(s1.isspace())  
print(s2.isspace())
```

ВЫВОДИТ:

```
True  
False
```

Для пустой строки метод `isspace()` также возвращает `False`, так как в этом случае строка не состоит из пробельных символов.

Приведённый ниже код:

```
s1 = ''  
print(s1.isspace())
```

ВЫВОДИТ:

```
False
```

❤️ Happy Pythoning! 🐍

Шаг 2

Что покажет приведённый ниже код?

```
s = 'aabbAA111ccDDaa'  
print(s.isalnum())  
print(s.isalpha())  
print(s.isdigit())
```

Чтобы решить это задание откройте <https://stepik.org/lesson/303084/step/2>

Шаг 3

Что покажет приведённый ниже код?

```
print('Cyberpunk 2077'.isalnum())
```

Чтобы решить это задание откройте <https://stepik.org/lesson/303084/step/3>

Шаг 4

Что покажет приведённый ниже код?

```
print('Cyberpunk'.isalnum())  
print('2077'.isalnum())
```

Чтобы решить это задание откройте <https://stepik.org/lesson/303084/step/4>

Шаг 5

Что покажет приведённый ниже код?

```
s = 'aabb!@$11cc'  
print(s.islower())
```

Чтобы решить это задание откройте <https://stepik.org/lesson/303084/step/5>

Шаг 6

Что покажет приведённый ниже код?

```
s = 'AAb!@$11CC'  
print(s.isupper())
```

Чтобы решить это задание откройте <https://stepik.org/lesson/303084/step/6>

Шаг 7

Что покажет приведённый ниже код?

```
print('2024-05-19'.islower())  
print('2024-05-19'.isupper())
```

Чтобы решить это задание откройте <https://stepik.org/lesson/303084/step/7>

Шаг 8

Что покажет приведённый ниже код?

```
s = '   abbc   '  
print(s.isspace())
```

Чтобы решить это задание откройте <https://stepik.org/lesson/303084/step/8>

Плохие комментарии

На платформе Stepik пользователи оставляют комментарии, но не все из них соответствуют правилам. Так, например, модератор Сэм считает неуместными комментариями те, которые представляют собой пустую строку или состоят только из пробелов. Подобные комментарии он удаляет – нечего засорять курс бесполезным материалом!

Ваша задача – написать программу, которая поможет Сэму проверять комментарии. Программа должна принимать на вход натуральное число n , а затем n строк, представляющих тексты комментариев. Для каждого комментария ваша программа должна выводить номер этого комментария (начиная с 1), затем двоеточие (:), затем через пробел его текст или сообщение «COMMENT SHOULD BE DELETED» (без кавычек), если комментарий должен быть удалён Сэмом.

Формат входных данных

На вход программе подаются натуральное число n , а затем – n строк.

Формат выходных данных

Для каждого комментария программа должна вывести его текст или сообщение «COMMENT SHOULD BE DELETED» (без кавычек), если комментарий должен быть удалён.

► Тестовые данные 

Sample Input 1:

```
4
Не проходит 7 тест, что у меня нет так?
Авторы, вы надоели уже со своей математикой и шахматами!

Почему у меня выходит ошибка IndexError?
```

Sample Output 1:

```
1: Не проходит 7 тест, что у меня нет так?
2: Авторы, вы надоели уже со своей математикой и шахматами!
3: COMMENT SHOULD BE DELETED
4: Почему у меня выходит ошибка IndexError?
```

Sample Input 2:

```
4

Почему мне не начислили баллы за задачу?! Я ее с первого раза решила
Я крутой, я решил контрольную за 7 минут..
```

Sample Output 2:

```
1: COMMENT SHOULD BE DELETED
2: COMMENT SHOULD BE DELETED
3: Почему мне не начислили баллы за задачу?! Я ее с первого раза решила
4: Я крутой, я решил контрольную за 7 минут..
```

Чтобы решить это задание откройте <https://stepik.org/lesson/303084/step/9>

Автомобильный номер 🚗 🌶️

В службе по дорожному движению решили оптимизировать процесс создания автомобильных номеров: вместо человека генерацию автомобильных номеров поручили некоторой GPT (модели машинного обучения). Как мы знаем, искусственный интеллект ещё сыроват и делает много ошибок, поэтому его результаты следует тщательно проверять. Корректный автомобильный номер (в России) имеет следующий формат:



Напишите программу, которая принимает на вход строку и проверяет, является ли эта строка корректным автомобильным номером. Программа должна вывести «YES» (без кавычек), если искусственный интеллект справился со своей задачей, или «NO» (без кавычек) в противном случае. В нашей задаче корректным автомобильным номером будем считать следующие форматы:

```
<БУКВА><ЦИФРА><ЦИФРА><ЦИФРА><БУКВА><БУКВА>_<ЦИФРА><ЦИФРА>
```

```
<БУКВА><ЦИФРА><ЦИФРА><ЦИФРА><БУКВА><БУКВА>_<ЦИФРА><ЦИФРА><ЦИФРА>
```

где <ЦИФРА> – это любая цифра, а <БУКВА> – это одна из букв кириллицы АВЕКМНОРСТУХ.

Формат входных данных

На вход программе подаётся одна строка – сгенерированный ИИ автомобильный номер.

Формат выходных данных

Программа должна вывести текст в соответствии с условием задачи.

Примечание. Подробнее почитать про автомобильные номера можно по ссылке (https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5_%D0%B7%D0%BD%D0%B0%D0%BA%D0%B8_%D1%82%D1%80%D0%B0%D0%BD%D1%81%D0%BF%D0%BE%D1%80%D1%82%D0%BD%D1%8B%D1%85_%D1%81%D1%80%D0%B5%D0%B4%D1%81%D1%82%D0%B2_%D0%B2_%D0%A0%D0%BE%D1%81%D1%81%D0%B8%D0%B8).

► Тестовые данные ●

Sample Input 1:

```
A123BC_45
```

Sample Output 1:

```
YES
```

Sample Input 2:

```
АЯ123В_45
```

Sample Output 2:

```
NO
```

Чтобы решить это задание откройте <https://stepik.org/lesson/303084/step/10>

Шаг 11

Проверь никнейм 🧑🌶️

Во время собеседования вам предложили решить задачу на валидацию имени пользователя. Пользователь пытается создать никнейм для своего аккаунта в соцсети Y. Правила для корректного никнейма в соцсети Y следующие:

- никнейм должен начинаться с символа @
- никнейм должен содержать от 5 до 15 (включительно) символов (включая первый символ @)
- никнейм должен содержать только **строчные** буквы и цифры (помимо первого символа @)

Напишите программу, которая выводит «Correct» (без кавычек), если никнейм соответствует **всем** вышеприведенным правилам, или «Incorrect» (без кавычек) в противном случае.

Формат входных данных

На вход программе подаётся одна строка – никнейм.

Формат выходных данных

Программа должна вывести текст в соответствии с условием задачи.

Примечание. Обратите внимание, что никнейму необязательно содержать строчные буквы и цифры **одновременно**, никнейм может содержать только строчные буквы или только цифры (помимо первого символа @). Например, следующие никнеймы считаются корректными:

```
@duncan
@1111
```

► Тестовые данные ●

Sample Input 1:

```
@paul_atreides
```

Sample Output 1:

```
Incorrect
```

Sample Input 2:

```
@chani7
```

Sample Output 2:

```
Correct
```

Чтобы решить это задание откройте <https://stepik.org/lesson/303084/step/11>