Практическая работа 5 Работа со строками в Python

Цель работы: познакомится с методами работы со строками.
Учащийся должен:
Владеть:
Навыками составления линейных алгоритмов на языке программирования Python с использованием строковых данных;
Уметь:
Применять функции и методы строк при обработке строковых данных;
Знать:
Операции и методы обработки строк.

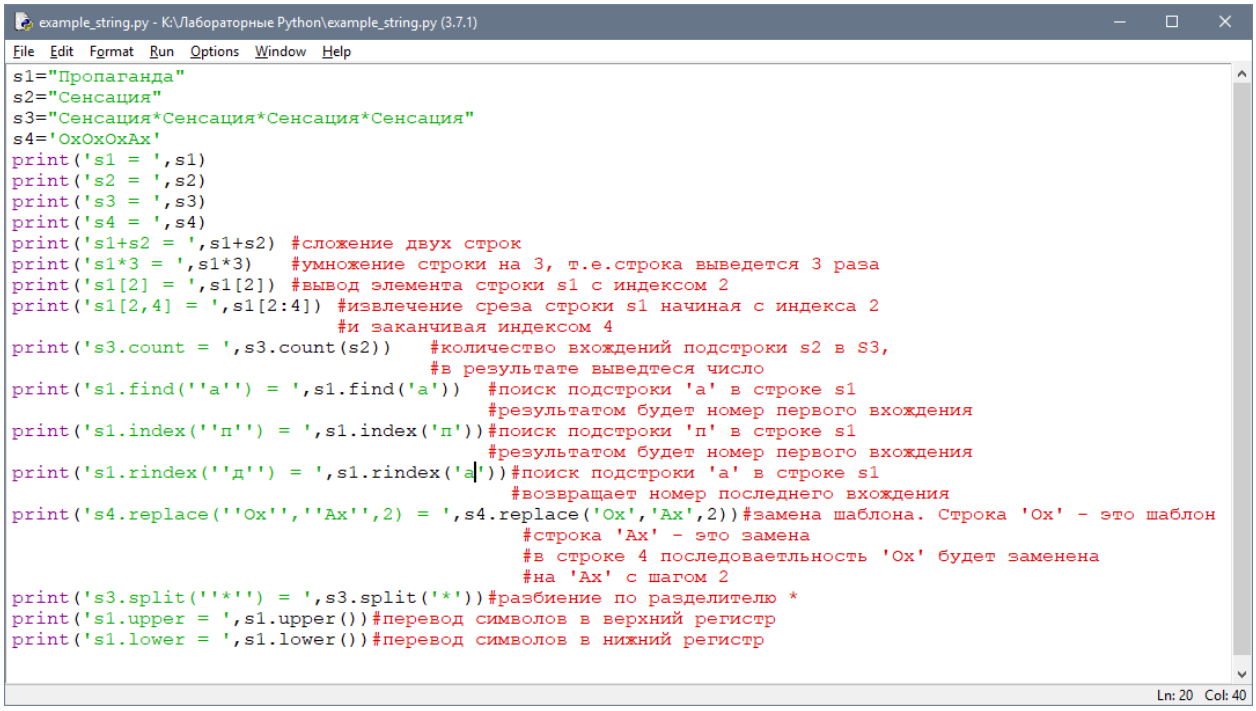
Строка — базовый тип представляющий из себя неизменяемую последовательность символов; str от «string» — «строка».

Функции и методы работы со строками

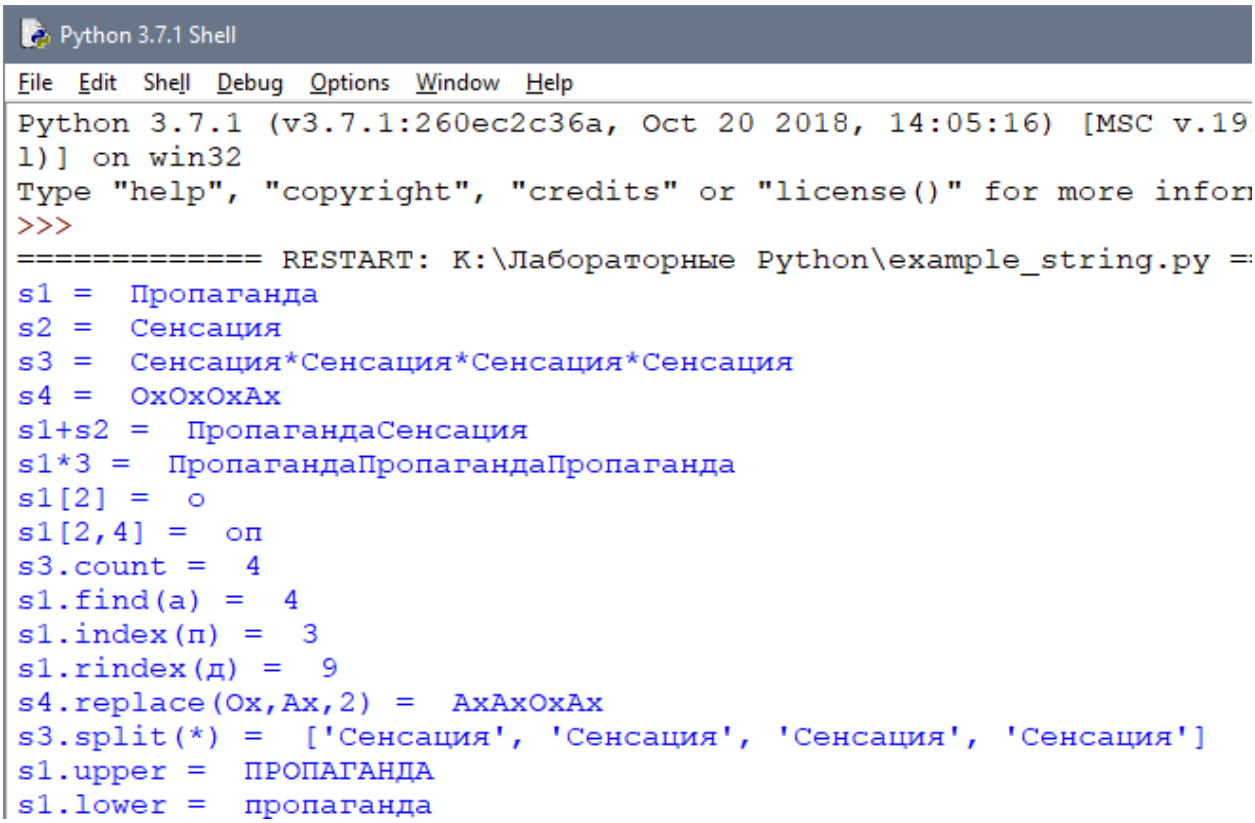
Функция или метод	Назначение
S1 + S2	Конкатенация (сложение строк)
S1 * 3	Повторение строки
S[i]	Обращение по индексу
S[i:j:step]	Извлечение среза
len(S)	Длина строки
S.join(список)	Соединение строк из последовательности str через разделитель, заданный строкой
S1.count(S[, i, j])	количество вхождений подстроки s в строку s1. Результатом является число. Можно указать позицию начала поиска i и окончания поиска *j*
S.find(str, [start],[end])	Поиск подстроки в строке. Возвращает номер первого вхождения или -1
S.index(str, [start],[end])	Поиск подстроки в строке. Возвращает номер первого вхождения или вызывает ValueError
S.rindex(str, [start],[end])	Поиск подстроки в строке. Возвращает номер последнего вхождения или вызывает ValueError
S.replace(шаблон, замена)	Замена шаблона
S.split(символ)	Разбиение строки по разделителю

S.upper()	Преобразование строки к верхнему регистру
S.lower()	Преобразование строки к нижнему регистру

Ниже приведена программа, демонстрирующая использование функций и методов работы со строками.



Пример программы на Python



Результат выполнения программы с использованием функций и методов работы со строками

Пример

Вариант 0

Проверить, будет ли строка читаться одинаково справа налево и слева направо (т. е. является ли она палиндромом).

Решение

Сначала введём строку командой: s=input('Введите строку ').

Затем определим логическую переменную flag и присвоим ей значение 1: flag=1.

Для начала в введённой строке нужно удалить пробелы. Для этого воспользуемся циклической конструкцией for, которая выполнится столько раз, какую имеет длину строка. Длину строки определим функцией len(s).

В теле цикла будем проверять следующее условие: s[i]!=' '. Данное логическое выражение будет истинно в том случае, если i-ый элемент строки не будет равен пробелу, тогда выполнится команда следующая после двоеточия: string+=s[i].

К сроке string, которая была объявлена в начале программы, будет добавляться посимвольно строка s, но уже без пробелов.

Для проверки строки на "палиндром" воспользуемся циклической конструкцией for.

Длина половины строки находится делением нацело на 2. Если количество символов нечетно, то стоящий в середине не учитывается, т.к. его сравниваемая пара - он сам.

Количество повторов цикла равно длине половины строки. Длину строки определим функцией len(s), где аргумент введённая нами строка s. Зная длину строки, можно вычислить количество повторов цикла. Для этого целочисленно разделим длину строки на 2: len(s)//2.

Для задания диапазона для цикла используем функцию range(), в которой аргументом будет являться половина длины строки: range(len(s//2)).

for i in range(len(s//2)).

Если символ с индексом i не равен "симметричному" символу с конца строки (который находится путем индексации с конца)

if s[i] != s[-1-i],

то переменной flag присваивается значение 0 и происходит выход из цикла командой break.

Далее, при помощи условной конструкции if-else в зависимости от значения flag либо - 0, либо -1 выводится сообщение, что строка палиндром, либо нет.

```
s=input('Введите строку \n')
flag=1
string=''
for i in range(len(s)):
    if s[i]!=' ':
        string+=s[i]
print(string)
for i in range(len(s)//2):
    if string[i]!=string[-i-1]:
        flag=0
        break
if flag: print('Палиндром')
else: print('не палиндром')
```

Пример программы на Python

```
Введите строку
а роза упала на лапу азора
арозаупаланалапуазора
Палиндром
```

Результат выполнения программы

Задания для самостоятельной работы (по вариантам)

Вариант 1

Дана строка, содержащая русскоязычный текст. Найти количество слов, начинающихся с буквы "е".

Вариант 2

В строке заменить все двоеточия (:) знаком процента (%). Подсчитать количество замен.

Вариант 3

В строке удалить символ точку (.) и подсчитать количество удаленных символов.

Вариант 4

В строке заменить букву(а) буквой (о). Подсчитать количество замен. Подсчитать, сколько символов в строке.

Вариант 5

В строке заменить все заглавные буквы строчными.

Вариант 6

В строке удалить все буквы "а" и подсчитать количество удаленных символов.

Вариант 7

Дана строка. Преобразовать ее, заменив звездочками все буквы "п", встречающиеся среди первых n/2 символов. Здесь n - длина строки.

Вариант 8

Дана строка, заканчивающаяся точкой. Подсчитать, сколько слов в строке.

Вариант 9

Определить, сколько раз в тексте встречается заданное слово.

Вариант 10

Дана строка-предложение на английском языке. Преобразовать строку так, чтобы каждое слово начиналось с заглавной буквы.

Вариант 11

Дана строка. Подсчитать самую длинную последовательность подряд идущих букв «н». Преобразовать ее, заменив точками все восклицательные знаки.

Вариант 12

Дана строка. Вывести все слова, оканчивающиеся на букву "я".

Вариант 13

Дана строка символов, среди которых есть одна открывающаяся и одна закрывающаяся скобки. Вывести на экран все символы, расположенные внутри этих скобок.

Вариант 14

Дана строка. Вывести все слова, начинающиеся на букву "а" и слова оканчивающиеся на букву "я".

Вариант 15

Дана строка текста. Подсчитать количество букв «т» в строке.

Форматирование строк с помощью метода format

Если для подстановки требуется только один аргумент, то значение - сам аргумент:

>>>

```
>>> 'Hello, {}!'.format('Vasya')

'Hello, Vasya!'
```

А если несколько, то значениями будут являться все аргументы со строками подстановки (обычных или именованных):

>>>

```
>>> '{0}, {1}, {2}'.format('a', 'b', 'c')

'a, b, c'

>>> '{}, {}, {}'.format('a', 'b', 'c')

'a, b, c'

>>> '{2}, {1}, {0}'.format('a', 'b', 'c')

'c, b, a'

>>> '{2}, {1}, {0}'.format(*'abc')

'c, b, a'

>>> '{0}{1}{0}'.format('abra', 'cad')

'abracadabra'

>>> 'Coordinates: {latitude},
{longitude}'.format(latitude='37.24N', longitude='-115.81W')

'Coordinates: 37.24N, -115.81W'

>>> coord = {'latitude': '37.24N', 'longitude': '-115.81W'}
```

```
>>> 'Coordinates: {latitude}, {longitude}'.format(**coord)

'Coordinates: 37.24N, -115.81W'
```

Однако метод format умеет большее. Вот его синтаксис:

```
поле замены     ::=  "{" [имя поля] ["!" преобразование] [":"
спецификация] "}"

имя поля        ::=  arg_name ("." имя атрибута | "[" индекс "]")*

преобразование  ::=  "r" (внутреннее представление) | "s"
(человеческое представление)

спецификация    ::=  см. ниже
```

Например:

>>>

```
>>> "Units destroyed: {players[0]}".format(players = [1, 2, 3])

'Units destroyed: 1'

>>> "Units destroyed: {players[0]!r}".format(players = ['1', '2',
'3'])

"Units destroyed: '1'"
```

Теперь спецификация формата:

```
спецификация ::=
[[fill]align][sign][#][0][width][,][.precision][type]

заполнитель  ::=  символ кроме '{' или '}'

выравнивание ::=  "<" | ">" | "=" | "^"

знак         ::=  "+" | "-" | " "

ширина       ::=  integer
```

```
точность      ::=  integer

тип           ::=  "b" | "c" | "d" | "e" | "E" | "f" | "F" | "g" |
"G" |

                   "n" | "o" | "s" | "x" | "X" | "%"
```

Выравнивание производится при помощи символа-заполнителя. Доступны следующие варианты выравнивания:

Флаг	**Значение**
'<'	Символы-заполнители будут справа (выравнивание объекта по левому краю) (по умолчанию).
'>'	выравнивание объекта по правому краю.
'='	Заполнитель будет после знака, но перед цифрами. Работает только с числовыми типами.
'^'	Выравнивание по центру.

Опция "знак" используется только для чисел и может принимать следующие значения:

Флаг	**Значение**
'+'	Знак должен быть использован для всех чисел.
'-'	'-' для отрицательных, ничего для положительных.

'Пробел'	'-' для отрицательных, пробел для положительных.

Поле "тип" может принимать следующие значения:

Тип	**Значение**
'd', 'i', 'u'	Десятичное число.
'o'	Число в восьмеричной системе счисления.
'x'	Число в шестнадцатеричной системе счисления (буквы в нижнем регистре).
'X'	Число в шестнадцатеричной системе счисления (буквы в верхнем регистре).
'e'	Число с плавающей точкой с экспонентой (экспонента в нижнем регистре).
'E'	Число с плавающей точкой с экспонентой (экспонента в верхнем регистре).
'f', 'F'	Число с плавающей точкой (обычный формат).
'g'	Число с плавающей точкой. с экспонентой (экспонента в нижнем регистре), если она меньше, чем -4 или точности, иначе обычный формат.

'G'	Число с плавающей точкой. с экспонентой (экспонента в верхнем регистре), если она меньше, чем -4 или точности, иначе обычный формат.
'c'	Символ (строка из одного символа или число - код символа).
's'	Строка.
'%'	Число умножается на 100, отображается число с плавающей точкой, а за ним знак %.

И напоследок, несколько примеров:

>>>

```
>>> coord = (3, 5)

>>> 'X: {0[0]};  Y: {0[1]}'.format(coord)

'X: 3;  Y: 5'

>>> "repr() shows quotes: {!r}; str() doesn't:
{!s}".format('test1', 'test2')

"repr() shows quotes: 'test1'; str() doesn't: test2"

>>> '{:<30}'.format('left aligned')

'left aligned                  '

>>> '{:>30}'.format('right aligned')

'                 right aligned'

>>> '{:^30}'.format('centered')

'           centered           '
```

```
>>> '{:*^30}'.format('centered')  # use '*' as a fill char

'**********centered**********'

>>> '{:+f}; {:+f}'.format(3.14, -3.14)  # show it always

'+3.140000; -3.140000'

>>> '{: f}; {: f}'.format(3.14, -3.14)  # show a space for positive
numbers

' 3.140000; -3.140000'

>>> '{:-f}; {:-f}'.format(3.14, -3.14)  # show only the minus --
same as '{:f}; {:f}'

'3.140000; -3.140000'

>>> # format also supports binary numbers

>>> "int: {0:d};  hex: {0:x};  oct: {0:o};  bin: {0:b}".format(42)

'int: 42;  hex: 2a;  oct: 52;  bin: 101010'

>>> # with 0x, 0o, or 0b as prefix:

>>> "int: {0:d};  hex: {0:#x};  oct: {0:#o};  bin:
{0:#b}".format(42)

'int: 42;  hex: 0x2a;  oct: 0o52;  bin: 0b101010'

>>> points = 19.5

>>> total = 22

>>> 'Correct answers: {:.2%}'.format(points/total)

'Correct answers: 88.64%'
```