СТРОКИ PYTHON

ПОНЯТИЕ СТРОКИ

СТРОКИ - УПОРЯДОЧЕННЫЕ ПОСЛЕДОВАТЕЛЬНОСТИ СИМВОЛОВ. ДЛИНА СТРОКИ ОГРАНИЧЕНА ОБЪЕМОМ ОПЕРАТИВНОЙ ПАМЯТИ КОМПЬЮТЕРА. СТРОКИ ПОДДЕРЖИВАЮТ ОБРАЩЕНИЕ К ЭЛЕМЕНТУ ПО ИНДЕКСУ, ПОЛУЧЕНИЕ СРЕЗА, КОНКАТЕНАЦИЮ, ПОВТОРЕНИЕ, ПРОВЕРКУ НА ВХОЖДЕНИЕ).

СТРОКИ ОТНОСЯТСЯ К НЕИЗМЕНЯЕМЫМ ТИПАМ ДАННЫХ. ПОЭТОМУ, ПРАКТИЧЕСКИ ВСЕ СТРОКОВЫЕ МЕТОДЫ В КАЧЕСТВЕ ЗНАЧЕНИЯ ВОЗВРАЩАЮТ НОВУЮ СТРОКУ. ПРИ ИСПОЛЬЗОВАНИИ НЕБОЛЬШИХ СТРОК ЭТО НЕ ВЫЗЫВАЕТ НИКАКИХ ПРОБЛЕМ, НО ПРИ РАБОТЕ С БОЛЬШИМИ СТРОКАМИ МОЖНО СТОЛКНУТЬСЯ С ПРОБЛЕМОЙ НЕХВАТКИ ПАМЯТИ.

СТРОКИ В PYTHON МОГУТ ЗАКЛЮЧАТЬСЯ КАК В ОДИНОЧНЫЕ, ТАК И ДВОЙНЫЕ КАВЫЧКИ. ОДНАКО, НАЧАЛО И КОНЕЦ СТРОКИ ДОЛЖНЫ ОБРАМЛЯТЬСЯ ОДИНАКОВЫМ ТИПОМ КАВЫЧЕК.

```
S = 'SPAM"S‘
```

```
S = "SPAM'S"
```

СТРОКОВЫЕ ТИПЫ PYTHON

1. ***STR*** – UNICODE-СТРОКА. ЭТО СТРОКИ В АБСТРАКТНОЙ КОДИРОВКЕ, ПОЗВОЛЯЮЩИЕ ХРАНИТЬ СИМВОЛЫ UNICODE И ПРОИЗВОДИТЬ МАНИПУЛЯЦИИ НАД НИМИ.
2. ***BYTES*** – НЕИЗМЕНЯЕМАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ БАЙТОВ. КАЖДЫЙ ЭЛЕМЕНТ ПОСЛЕДОВАТЕЛЬНОСТИ МОЖЕТ ХРАНИТЬ ЧИСЛО *ОТ 0 ДО 255*, КОТОРОЕ ОБОЗНАЧАЕТ КОД СИМВОЛА. ОБЪЕКТ ТИПА BYTES ПОДДЕРЖИВАЕТ БОЛЬШИНСТВО СТРОКОВЫХ МЕТОДОВ И, ЕСЛИ ЭТО ВОЗМОЖНО, ВЫВОДИТСЯ КАК ПОСЛЕДОВАТЕЛЬНОСТЬ СИМВОЛОВ. ДОСТУП ПО ИНДЕКСУ ВОЗВРАЩАЕТ ЦЕЛОЕ ЧИСЛО, А НЕ СИМВОЛ. ФУНКЦИЯ ***LEN()*** ВЕРНЕТ КОЛИЧЕСТВО БАЙТОВ, А НЕ СИМВОЛОВ.
3. ***BYTEARREY*** – ИЗМЕНЯЕМАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ БАЙТОВ. ДАННЫЙ ТИП АНАЛОГИЧЕН ПРЕДЫДУЩЕМУ ТИПУ, НО ПОЗВОЛЯЕТ ИЗМЕНЯТЬ ЭЛЕМЕНТЫ ПО ИНДЕКСУ И СОДЕРЖИТ ДОПОЛНИТЕЛЬНЫЕ МЕТОДЫ, ДАЮЩИЕ ВОЗМОЖНОСТЬ ДОБАВЛЯТЬ И УДАЛЯТЬ ЭЛЕМЕНТЫ.

ВО ВСЕХ СЛУЧАЯХ, КОГДА РЕЧЬ ИДЕТ О ТЕКСТОВЫХ ДАННЫХ, СЛЕДУЕТ ИСПОЛЬЗОВАТЬ ТИП ***STR***. *ТИПЫ BYTES И BYTEARRAY СЛЕДУЕТ ИСПОЛЬЗОВАТЬ ДЛЯ ЗАПИСИ БИНАРНЫХ ДАННЫХ* – НАПРИМЕР, ИЗОБРАЖЕНИЙ, А ТАКЖЕ ДЛЯ ПРОМЕЖУТОЧНОГО ХРАНЕНИЯ ТЕКСТОВЫХ ДАННЫХ.

СОЗДАНИЕ СТРОК

- С ПОМОЩЬЮ ФУНКЦИИ

STR ([ОБЪЕКТ], КОДИРОВКА[, ОБРАБОТКА ОШИБОК]])

ЕСЛИ УКАЗАН ТОЛЬКО ПЕРВЫЙ ПАРАМЕТР, ТО ФУНКЦИЯ ВОЗВРАЩАЕТ СТРОКОВОЕ ПРЕДСТАВЛЕНИЕ ЛЮБОГО ОБЪЕКТА. ЕСЛИ ПАРАМЕТРЫ НЕ УКАЗАНЫ ВООБЩЕ, ТО ВОЗВРАЩАЕТСЯ ПУСТАЯ СТРОКА.

```
>>> str(), str([1, 2]), str((3, 4)), str({"x": 1})
('', '[1, 2]', '(3, 4)', "{'x': 1}")
```

- УКАЗАВ СТРОКУ МЕЖДУ АПОСТРОФАМИ ИЛИ ДВОИНЫМИ КАВЫЧКАМИ:

```
>>> 'строка', "строка", '"x": 5', "'x': 5"
('строка', 'строка', '"x": 5', "'x': 5")
>>> print('Строка1\nСтрока2')
Строка1
Строка2
```

СПЕЦИАЛЬНЫЕ СИМВОЛЫ

- \n — перевод строки;
- \r — возврат каретки;
- \t — знак табуляции;
- \v — вертикальная табуляция;
- \a — звонок;
- \b — забой;
- \f — перевод формата;
- \0 — нулевой символ (не является концом строки);
- \" — кавычка;
- \' — апостроф;
- \N — восьмеричное значение N. Например, \74 соответствует символу <;
- \xN — шестнадцатеричное значение N. Например, \x6a соответствует символу j;
- \\ — обратный слеш;
- \uxxxx — 16-битный символ Unicode. Например, \u043a соответствует русской букве к;
- \Uxxxxxxxx — 32-битный символ Unicode.

ОПЕРАЦИИ СО СТРОКАМИ

- ***ОБРАЩЕНИЕ К ОТДЕЛЬНОМУ СИМВОЛУ СТРОКИ ПО ИНДЕКСУ***:

```
>>> 'MORNING, AFTERNOON, NIGHT'[1]
'O'
>>> TDAY = 'MORNING, AFTERNOON, NIGHT'
>>> TDAY[4]
'I'
```

- ***ИЗВЛЕЧЕНИЕ ПОДСТРОКИ (СРЕЗ): [X:Y],***

X – ЭТО ИНДЕКС НАЧАЛА СРЕЗА

Y –ОКОНЧАНИЕ СТРОКИ, ПРИЧЕМ СИМВОЛ С НОМЕРОМ Y В СРЕЗ УЖЕ НЕ ВХОДИТ.

ЕСЛИ ОТСУТСТВУЕТ ПЕРВЫЙ ИНДЕКС, ТО СРЕЗ БЕРЕТСЯ ОТ НАЧАЛА ДО ВТОРОГО ИНДЕКСА; ПРИ ОТСУТСТВИИ ВТОРОГО ИНДЕКСА, СРЕЗ БЕРЕТСЯ ОТ ПЕРВОГО ИНДЕКСА ДО КОНЦА СТРОКИ.

```
>>> TDAY = 'MORNING, AFTERNOON, NIGHT'

>>> TDAY[0:7]

'MORNING'

>>> TDAY[9:-7]

'AFTERNOON'

>>> TDAY[-5:]

'NIGHT'
```

ОПЕРАЦИИ СО СТРОКАМИ

- ***ИЗВЛЕЧЕНИЕ СИМВОЛОВ С ШАГОМ:***

[X:Y:Z];

Z – ЭТО ШАГ, ЧЕРЕЗ КОТОРЫЙ ОСУЩЕСТВЛЯЕТСЯ ВЫБОР ЭЛЕМЕНТОВ.

```
>>> STR4 = "FULL BALL FILL PACK RING"
>>> STR4[::5]
'FBFPR'
>>> STR4[0:15:2]
'FL ALFL ‘
```

- ***КОНКАТЕНАЦИЯ (СЛОЖЕНИЕ):***

S=S1 + S2

```
>>> S1 = 'SPAM'
>>> S2 = 'EGGS'
>>> PRINT(S1 + S2)
'SPAMEGGS‘
```

- ***ДУБЛИРОВАНИЕ СТРОКИ***

```
>>> PRINT('SPAM' * 3)
SPAMSPAMSPAM
```

- ***ДЛИНА СТРОКИ (ФУНКЦИЯ LEN)***

```
>>> LEN('SPAM')
4
```

МЕТОДЫ РАБОТЫ СО СТРОКАМИ

МЕТОД — ЭТО ФУНКЦИЯ, ПРИМЕНЯЕМАЯ К ОБЪЕКТУ, В ДАННОМ СЛУЧАЕ — К СТРОКЕ. МЕТОД ВЫЗЫВАЕТСЯ В ВИДЕ

ИМЯ_ОБЪЕКТА.ИМЯ_МЕТОДА(ПАРАМЕТРЫ)

НАПРИМЕР, *S.FIND("E")* - ЭТО ПРИМЕНЕНИЕ К СТРОКЕ S МЕТОДА FIND С ОДНИМ ПАРАМЕТРОМ "E".

МЕТОДЫ

- ***FIND*** НАХОДИТ В ДАННОЙ СТРОКЕ (К КОТОРОЙ ПРИМЕНЯЕТСЯ МЕТОД) ДАННУЮ ПОДСТРОКУ (КОТОРАЯ ПЕРЕДАЕТСЯ В КАЧЕСТВЕ ПАРАМЕТРА). ФУНКЦИЯ ВОЗВРАЩАЕТ ИНДЕКС ПЕРВОГО ВХОЖДЕНИЯ ИСКОМОЙ ПОДСТРОКИ. ЕСЛИ ЖЕ ПОДСТРОКА НЕ НАЙДЕНА, ТО МЕТОД ВОЗВРАЩАЕТ ЗНАЧЕНИЕ -1.

```
S = 'HELLO‘
PRINT(S.FIND('E'))
# ВЕРНЁТ 1
PRINT(S.FIND('LL'))
# ВЕРНЁТ 2
PRINT(S.FIND('L'))
# ВЕРНЁТ -1
```

ЕСЛИ ВЫЗВАТЬ МЕТОД ***FIND*** С ТРЕМЯ ПАРАМЕТРАМИ ***S.FIND(T, A, B),*** ТО ПОИСК БУДЕТ ОСУЩЕСТВЛЯТЬСЯ *В СРЕЗЕ S[A:B]*. ЕСЛИ УКАЗАТЬ ТОЛЬКО ДВА ПАРАМЕТРА ***S.FIND(T, A),*** ТО ПОИСК БУДЕТ ОСУЩЕСТВЛЯТЬСЯ В СРЕЗЕ *S[A:]*, ТО ЕСТЬ НАЧИНАЯ С СИМВОЛА С ИНДЕКСОМ A И ДО КОНЦА СТРОКИ.

МЕТОД ***S.FIND(T, A, B)*** ВОЗВРАЩАЕТ ИНДЕКС В СТРОКЕ S, А НЕ ИНДЕКС ОТНОСИТЕЛЬНО СРЕЗА.

МЕТОДЫ РАБОТЫ СО СТРОКАМИ

- ***RFIND*** ВОЗВРАЩАЕТ ИНДЕКС ПОСЛЕДНЕГО ВХОЖДЕНИЯ ДАННОЙ СТРОКИ (“ПОИСК СПРАВА”).

```
S = 'HELLO‘
PRINT(S.FIND('L'))
# ВЕРНЁТ 2
PRINT(S.RFIND('L'))
# ВЕРНЁТ 3
```

- ***REPLACE*** *ЗАМЕНЯЕТ ВСЕ ВХОЖДЕНИЯ ОДНОЙ СТРОКИ НА ДРУГУЮ:*

S.REPLACE(OLD, NEW)

ЗАМЕНИТЬ В СТРОКЕ S ВСЕ ВХОЖДЕНИЯ ПОДСТРОКИ OLD НА ПОДСТРОКУ NEW.

```
PRINT('Hello'.REPLACE('L', 'L'))      # ВЕРНЁТ 'HELLO‘
```

ЕСЛИ МЕТОДУ ***REPLACE*** ЗАДАТЬ ЕЩЕ ОДИН ПАРАМЕТР: ***S.REPLACE(OLD, NEW, COUNT)***, ТО ЗАМЕНЕНЫ БУДУТ НЕ ВСЕ ВХОЖДЕНИЯ, А ТОЛЬКО НЕ БОЛЬШЕ, ЧЕМ ПЕРВЫЕ ***COUNT*** ИЗ НИХ.

```
PRINT('Abrakadabra'.REPLACE('a', 'A', 2))      #ВЕРНЁТ 'AbrAkAdabra‘
```

- *МЕТОД COUNT ПОДСЧИТЫВАЕТ КОЛИЧЕСТВО ВХОЖДЕНИЙ ОДНОЙ СТРОКИ В ДРУГУЮ СТРОКУ:*

S.COUNT(T)

ВОЗВРАЩАЕТ *ЧИСЛО ВХОЖДЕНИЙ СТРОКИ T ВНУТРИ СТРОКИ S*. ПРИ ЭТОМ ПОДСЧИТЫВАЮТСЯ ТОЛЬКО НЕПЕРЕСЕКАЮЩИЕСЯ ВХОЖДЕНИЯ, НАПРИМЕР:

```
print('Abracadabra'.count('a'))
# вернёт 4
print(('a' * 10).count('aa'))
# вернёт 5
```

МЕТОДЫ РАБОТЫ СО СТРОКАМИ

S.split(символ)	Разбиение строки по разделителю
S.isdigit()	Состоит ли строка из цифр
S.isalpha()	Состоит ли строка из букв
S.isalnum()	Состоит ли строка из цифр или букв
S.islower()	Состоит ли строка из символов в нижнем регистре
S.isupper()	Состоит ли строка из символов в верхнем регистре
S.isspace()	Состоит ли строка из неотображаемых символов (пробел, символ перевода страницы ('\f'), "новая строка" ('\n'), "перевод каретки" ('\r'), "горизонтальная табуляция" ('\t') и "вертикальная табуляция" ('\v'))
S.istitle()	Начинаются ли слова в строке с заглавной буквы
S.upper()	Преобразование строки к верхнему регистру

МЕТОДЫ РАБОТЫ СО СТРОКАМИ

S.lower()	Преобразование строки к нижнему регистру
S.startswith(str)	Начинается ли строка S с шаблона str
S.endswith(str)	Заканчивается ли строка S шаблоном str
S.join(список)	Сборка строки из списка с разделителем S
ord(символ)	Символ в его код ASCII
chr(число)	Код ASCII в символ
S.capitalize()	Переводит первый символ строки в верхний регистр, а все остальные в нижний
S.center(width, [fill])	Возвращает отцентрованную строку, по краям которой стоит символ fill (пробел по умолчанию)
S.count(str, [start],[end])	Возвращает количество непересекающихся вхождений подстроки в диапазоне [начало, конец] (0 и длина строки по умолчанию)
S.expandtabs([tabsize])	Возвращает копию строки, в которой все символы табуляции заменяются одним или несколькими пробелами, в зависимости от текущего столбца. Если TabSize не указан, размер табуляции полагается равным 8 пробелам

МЕТОДЫ РАБОТЫ СО СТРОКАМИ

S.lstrip([chars])	Удаление пробельных символов в начале строки
S.rstrip([chars])	Удаление пробельных символов в конце строки
S.strip([chars])	Удаление пробельных символов в начале и в конце строки
S.partition(шаблон)	Возвращает кортеж, содержащий часть перед первым шаблоном, сам шаблон, и часть после шаблона. Если шаблон не найден, возвращается кортеж, содержащий саму строку, а затем две пустых строки
S.rpartition(sep)	Возвращает кортеж, содержащий часть перед последним шаблоном, сам шаблон, и часть после шаблона. Если шаблон не найден, возвращается кортеж, содержащий две пустых строки, а затем саму строку
S.swapcase()	Переводит символы нижнего регистра в верхний, а верхнего – в нижний
S.title()	Первую букву каждого слова переводит в верхний регистр, а все остальные в нижний
S.zfill(width)	Делает длину строки не меньшей width, по необходимости заполняя первые символы нулями
S.ljust(width, fillchar=" ")	Делает длину строки не меньшей width, по необходимости заполняя последние символы символом fillchar
S.rjust(width, fillchar=" ")	Делает длину строки не меньшей width, по необходимости заполняя первые символы символом fillchar
S.format(*args, **kwargs)	Форматирование строки

ЗАДАНИЕ

1. ОПРЕДЕЛИТЬ ПРОЦЕНТ СТРОЧНЫХ И ПРОПИСНЫХ БУКВ В СТРОКЕ.
2. В СТРОКЕ ЗАМЕНИТЬ ПРОБЕЛЬНЫЕ СИМВОЛЫ ЗНАКОМ ЗВЕЗДОЧКИ ("*"). ЕСЛИ ВСТРЕЧАЕТСЯ ПОДРЯД НЕСКОЛЬКО ПРОБЕЛОВ, ТО ИХ СЛЕДУЕТ ЗАМЕНИТЬ ОДНИМ ЗНАКОМ "*", ПРОБЕЛЫ В НАЧАЛЕ И КОНЦЕ СТРОКИ УДАЛИТЬ.