**Строки**

* <https://www.tutorialspoint.com/python/python_strings.htm> - Tutorial
* <https://docs.python.org/3/library/stdtypes.html#textseq>
* <https://docs.python.org/3/library/string.html>

**Создать строку**

* DEA! Строку можно написать в одинарных, двойных и тройных кавычках.
* \n \t и прочие - специальные значения (новая строка, табуляция, ...)
* DEA! r'строка' или R'строка' оставит \ без изменений.

Пустая строка

a = ''

b = ""

Не пустая строка

a = 'apple'

b = "banana"

c = '''текст на одной строке

и на другой строке'''

* a[0] - первый символ строки
* a[-1] - последний символ строки

**Срезы (Slice)**

* s[a:b:step] - часть строки с элементами от номера а (включительно) до номера b (не включительно), с шагом step

**Сравнение строк**

Для сравнения строк используются операторы =, !, <, <=, >=, <

Они выполняют побайтовое сравнение строк в памяти.

**Методы**

DEA! Нельзя изменить строку. Можно получить другую строку.

|  |  |  |  |
| --- | --- | --- | --- |
| [**Функция**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=1&up=0#sorted_table) | [**Что делает**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=1&up=0#sorted_table) | [**код**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=2&table=1&up=0#sorted_table) | [**результат**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=3&table=1&up=0#sorted_table) |
| a+b | строки вместе | "hel"+"lo" | "hello" |
| str\*n | повторить str n раз | "hi"\*3 | "hihihi" |
| **len**(str) | длина строки | len("hello") | 5 |
| str.**count**t(s) | сколько раз s встретилась в str | "alalabla".count("la")  "alalabla".count("ala") | 3  1 |
| str.**startswith**(s) | начинается str на s? | "hello".startswith("hel") "hello".startswith("el") | True False |
| str.**endswith**(s) | заканчивается str на s? | "hello".endswith("lo") "hello".endswith("el") | True False |
| s **in** str | строка s в str | "el" in "hello" "hel" in "hello" "Hel" in "hello" | True True False |
| str.**find**(s) | возвращает *первое* вхождение s в str или -1, если нет | "balalabla".find("la") "alalabla".find("zx") | 2 -1 |
| str.**index**(s) | возвращает *первое* вхождение s в str или ValueError[?](http://acm.mipt.ru/twiki/bin/edit/Cintro/ValueError?topicparent=Cintro.PythonString), если нет | "balalabla".index("la") | 2 |
| str.**rfind**(s) | возвращает *последнее* вхождение s в str или -1, если нет | "balalabla".rfind("la") "alalabla".rfind("zx") | 7 -1 |
| str.**rindex**(s) | возвращает *последнее* вхождение s в str или ValueError[?](http://acm.mipt.ru/twiki/bin/edit/Cintro/ValueError?topicparent=Cintro.PythonString), если нет | "balalabla".rindex("la") | 7 |
| str.**lstrip**(chars) | убирает символы из chars с начала строки | ' spacious '.lstrip() 'www.example.com'.lstrip('cmowz.') | 'spacious ' 'example.com' |
| str.**rstrip**(chars) | убирает символы из chars с конца строки | ' spacious '.rstrip() 'mississippi'.rstrip('ipz') | ' spacious' 'mississ' |
| str.**strip**(chars) | убирает символы из chars во всей строке | ' spacious '.strip() 'www.example.com'.strip('cmowz.') | 'spacious' 'example' |
| str.**replace**(old, new) str.replace(old, new, count) | Заменяет в str строку old на строку new (не более count раз) | "balalabla".replace("la", "xyz") "balalabla".replace("la", "xyz", 2) | "baxyzxyzbxyz" "baxyzxyzbla" |
| str.**partition**(sep) | возвращает 3 строки: до, sep, после |  |  |
| str.**rpartition**(sep) | возвращает 3 строки: до, sep, после |  |  |
| a = str.**split**(delimiter) a = str.split(delimiter, count) | Сделать список а, разбив строку str на элементы по разделителям delimeter; по умолчанию делится по пробелам |  |  |
| a = str.**rsplit**(delimiter) a = str.split(delimiter, count) | Сделать список а, разбив строку str на элементы по разделителям delimeter; по умолчанию делится по пробелам |  |  |
| s = str.join(a) | сделать из списка a строку, между элементами вставлять подстроку str |  |  |
| s = str.lower() | возвращает строку, преобразованную к нижнему регистру |  |  |
| s = str.upper() | возвращает строку, преобразованную к ВЕРХНЕМУ регистру |  |  |

Это не все возможные функции. О других функциях смотрите в [документации](https://docs.python.org/3/library/stdtypes.html#text-sequence-type-str).

Примеры:

before, s, after = "I have 2 small bombs and big bomb".partition("bomb")

print(before) # "I have 2 small "

print(s) # "bomb"

print(after) # "s and big bomb"

before, s, after = "I have 2 small bombs and big bomb".partition("dog")

print(before) # "I have 2 small bombs and big bomb"

print(s) # ""

print(after) # ""

>>> '1 2 3'.split()

['1', '2', '3']

>>> '1,2,3'.split(',')

['1', '2', '3']

>>> '1,2,3'.split(',', maxsplit=1)

['1', '2,3']

>>> '1,2,,3,'.split(',')

['1', '2', '', '3', '']

>>> '1 2 3'.split()

['1', '2', '3']

>>> '1 2 3'.split(maxsplit=1)

['1', '2 3']

>>> ' 1 2 3 '.split()

['1', '2', '3']

s = ["abc", "de", "xyz"].join("-")

print(s) # "abc-de-xyz"

**Задачи на строки**

**1 - Срезы строки**

Дана строка 'Abrucadabra'. Последовательно на разных строках выведите:

|  |  |
| --- | --- |
| [**Что вывести**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=2&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=2&up=0#sorted_table) |
| третий символ этой строки; | u |
| предпоследний символ этой строки; | a |
| первые пять символов этой строки; | Abruc |
| всю строку, кроме последних двух символов; | Abrucadab |
| все символы с четными номерами (считая, что индексация начинается с 0); | Arcdba |
| все символы с нечетными номерами; | buaar |
| все символы в обратном порядке. | arbadacurbA |
| все символы строки через один в обратном порядке, начиная с последнего; | abdcrA |
| длину данной строки. | 11 |

**2 - Слов в строке**

Дана строка. Подсчитать, сколько в ней слов.

**4 - Замените буквы, но не все**

Замените в строке все появления буквы h на букву H, кроме первого и последнего вхождения. Пример:

|  |  |
| --- | --- |
| [**Input**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=3&up=0#sorted_table) | [**Output**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=3&up=0#sorted_table) |
| aahhhhhbb | aahHHHhbb |

**Капитан Флинт (int)**

Капитан Флинт зарыл клад на Острове сокровищ. Он оставил описание, как найти клад. Описание состоит из строк вида: "0 5", где первое число – одно из 0 (North, север), 2 (Sout, юг), 1 (East, восток), 3 (West, запад), а второе число – количество шагов, необходимое пройти в этом направлении.

Напишите программу, которая по описанию пути к кладу определяет точные координаты клада, считая, что начало координат находится в начале пути, ось OX направлена на восток, ось OY – на север.

Программа получает на вход последовательность строк указанного вида, завершающуюся строкой со словом "Treasure!". Программа должна вывести два целых числа: координаты клада.

Путь 5 шагов на север, 3 шага на восток, 1 шаг на юг мы запишем как:

0 5   
1 3   
2 1   
Treasure!

|  |  |
| --- | --- |
| [**Вход**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=4&up=0#sorted_table) | [**Выход**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=4&up=0#sorted_table) |
| 0 5  1 3  2 1  Treasure! | 3 4 |

# -\*- coding: utf-8 -\*-

x = 0

y = 0

while True :

# прочитали одну строку в line

line = raw\_input()

# если Treasure! надо напечатать координаты и остановиться

if line == 'Treasure!' :

print x, y

break

# line разбили на части и каждую часть сделали int

pair = line.split()

direction = int(pair[0])

d = int(pair[1])

if direction == 0 : # север

y = y + d

# тут нужно дописать решение для юг, восток, запад

**Капитан Флинт (string)**

Капитан Флинт зарыл клад на Острове сокровищ. Он оставил описание, как найти клад. Описание состоит из строк вида: "куда 5", где первое слово *куда* – одно из: north (север), south (юг), east (восток), west (запад), а второе число – количество шагов, необходимое пройти в этом направлении.

Напишите программу, которая по описанию пути к кладу определяет точные координаты клада, считая, что начало координат находится в начале пути, ось OX направлена на восток, ось OY – на север.

Программа получает на вход последовательность строк указанного вида, завершающуюся строкой со словом "Treasure!". Программа должна вывести два целых числа: координаты клада.

Путь 5 шагов на север, 3 шага на восток, 1 шаг на юг мы запишем как:

north 5   
east 3   
south 1   
Treasure!

|  |  |
| --- | --- |
| [**Вход**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=5&up=0#sorted_table) | [**Выход**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=5&up=0#sorted_table) |
| north 5  east 3  south 1  Treasure! | 3 4 |

**1str\_bomb**

Напечатать YES, если в тексте есть слово bomb, иначе напечатать NO.

|  |  |
| --- | --- |
| [**Текст**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=6&up=0#sorted_table) | [**Результат**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=6&up=0#sorted_table) |
| a high explosive bomb is one that employs a process called detonation to rapidly release its chemical energy | YES |
| the simplest and oldest type of bombs store energy in the form of a low explosive | NO |
| mumbai also known as bombay is the capital city of the indian state of maharashtra | NO |
| the seven islands that came to constitute mumbai were home to communities of fishing colonies | NO |

**1str\_len**

Напечатать самое длинное слово и его длину. Если несколько слов одинаковой длины, то печатать первое из них.

|  |  |
| --- | --- |
| [**Текст**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=7&up=0#sorted_table) | [**Результат**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=7&up=0#sorted_table) |
| a high explosive bomb is one that employs a process called detonation to rapidly release its chemical energy | detonation 10 |

**2str\_bomb**

Напечатать YES, если в тексте есть подстрока bomb, иначе напечатать NO.

|  |  |
| --- | --- |
| [**Текст**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=8&up=0#sorted_table) | [**Результат**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=8&up=0#sorted_table) |
| a high explosive bomb is one that employs a process called detonation to rapidly release its chemical energy | YES |
| the simplest and oldest type of bombs store energy in the form of a low explosive | YES |
| mumbai also known as bombay is the capital city of the indian state of maharashtra | YES |
| BOMBS | NO |
| the seven islands that came to constitute mumbai were home to communities of fishing colonies | NO |

**3str\_bomb**

Напечатать YES, если в тексте есть подстрока bomb (без учета регистра, то есть находить слова BOMB, bOmB и так далее). Иначе напечатать NO.

|  |  |
| --- | --- |
| [**Текст**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=9&up=0#sorted_table) | [**Результат**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=9&up=0#sorted_table) |
| a high explosive bomb is one that employs a process called detonation to rapidly release its chemical energy | YES |
| the simplest and oldest type of bombs store energy in the form of a low explosive | YES |
| mumbai also known as bombay is the capital city of the indian state of maharashtra | YES |
| BOMBS | YES |
| the seven islands that came to constitute mumbai were home to communities of fishing colonies | NO |

**4str\_bomb**

Напечатать сколько раз в тексте встретилась подстрока bomb (маленькими буквами; дополнительно - без учета регистра).

|  |  |
| --- | --- |
| [**Текст**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=10&up=0#sorted_table) | [**Результат**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=10&up=0#sorted_table) |
| i have a bomb. you have a bomb | 2 |
| a high explosive bomb is one that employs a process called detonation to rapidly release its chemical energy | 1 |
| the simplest and oldest type of bombs store energy in the form of a low explosive | 1 |
| mumbai also known as bombay is the capital city of the indian state of maharashtra | 1 |
| BOMBS |  |
| the seven islands that came to constitute mumbai were home to communities of fishing colonies |  |

**5str\_bomb**

Напечатать текст, заменив все подстроки bomb на watermelon.

|  |  |
| --- | --- |
| [**Текст**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=11&up=0#sorted_table) | [**Результат**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=11&up=0#sorted_table) |
| i have a bomb. you have bombs. | i have a watermelon. you have watermelon. |

**Таблица в HTML (дополнительная задача)**

Есть данные в виде csv файла

"COUNTRY","2000","2001",2002,2003,2004

"ANTIGUA AND BARBUDA",0,0,0,0,0

"ARGENTINA",37,35,33,36,39

"BAHAMAS, THE",1,1,1,1,1

"BAHRAIN",5,6,6,6,6

(Можете положить их в файл. Например, t.csv)

Нужно преобразовать данные из csv в html.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Country | 2000 | 2001 | 2002 | 2003 | 2004 |
| ANTIGUA AND BARBUDA | 0 | 0 | 0 | 0 | 0 |
| Argentina | 37 | 35 | 33 | 36 | 39 |
| BAHAMAS, THE | 1 | 1 | 1 | 1 | 1 |
| BAHRAIN | 5 | 6 | 6 | 6 | 6 |

<table border='1'>

<tr bgcolor='lightgreen'>

<td>Country</td><td align='right'>2000</td><td align='right'>2001</td>

<td align='right'>2002</td><td align='right'>2003</td>

<td align='right'>2004</td>

</tr>

<tr bgcolor='lightyellow'><td>ANTIGUA AND BARBUDA</td>

<td align='right'>0</td><td align='right'>0</td>

<td align='right'>0</td><td align='right'>0</td>

<td align='right'>0</td></tr>

<tr bgcolor='white'><td>Argentina</td>

<td align='right'>37</td><td align='right'>35</td>

<td align='right'>33</td><td align='right'>36</td>

<td align='right'>39</td></tr>

<tr bgcolor='lightyellow'><td>BAHAMAS, THE</td>

<td align='right'>1</td><td align='right'>1</td>

<td align='right'>1</td><td align='right'>1</td>

<td align='right'>1</td></tr>

<tr bgcolor='white'><td>BAHRAIN</td>

<td align='right'>5</td><td align='right'>6</td>

<td align='right'>6</td><td align='right'>6</td>

<td align='right'>6</td></tr>

</table>

Если исходные данные в файле t.csv, а программа написана в файле t.py, то можно сохранить полученную таблицу в файл t.html:

python t.py < t.csv > t.html

А потом открыть файл t.html в браузере.

Кому этой задачи мало, берут ее модификацию из Лутц, глава 2, Упражнения.

**Дополнительно**

В Python 3.0 существует три строковых типа: str – для пред- ставления текста Юникода (содержащего символы в кодировке ASCII и символы в других кодировках), bytes – для представле- ния двоичных данных (включая кодированный текст) и bytearray – изменяемый вариант типа bytes

**Кодировки символов**

[Unicode HowTo](https://docs.python.org/3/howto/unicode.html) (En)

Для хранения информации в компьютерах используются байты. (Обычно 8 бит, в них можно хранить числа от 0 до 255.)

Чтобы хранить текст в виде чисел, каждый символ (буква, цифра, знак препинания, другой символ) обозначают своим числом.

Пары "символ - число" образуют таблицу кодировки. Таблицы кодировки бывают разными. Чтобы английский текст в разных кодировках выглядел одинаково, принято одинаково кодировать в разных кодировках первые 127 символов (латинские буквы, арабские цифры и знаки препинания).

Расширенная [таблица символов ASCII](http://acm.mipt.ru/twiki/bin/view/Cintro/CodeASCII) (1 символ хранится в 1 байте).

Использовать разные таблицы кодировок неудобно. Для русского языка было несколько разных таблиц. Решили сделать одну общую таблицу кодировки Unicode, где символы сгруппированы по страницам. В таблице около 1 000 000 символов. Для хранения каждого символа хватит 32 бит. Первые 127 символов Unicode совпадают с ASCII.

Если текст на английском, то для кодирования каждого символа хватит 1 байта. Если на каждый символ отводить 32 бита, то в кодировке текста 3/4 будут 0. Поэтому первые 127 символов в кодировке UTF-8 кодируются 1-байтовыми значениями, а остальными 2 или 3 байтовыми. Она удобна для кодирования языков с небольшим количеством букв (не иероглифов). В кодировке UTF-16 большинство символов кодируется 2 байтами, остальные - 4 байтами. Ей удобнее кодировать тексты с иероглифами.

DEA! По умолчанию в python3 считается, что программа написана в кодировке UFT-8.

Можно написать в другой кодировке, указав ее в программе. Так мы можем указать, что используется кодировка UTF-8 (если мы хотим писать в питоне 2.x и использовать русские буквы):

# -\*- coding: utf-8 -\*-

|  |  |
| --- | --- |
| [**Функция**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=0&table=12&up=0#sorted_table) | [**Что делает**](http://acm.mipt.ru/twiki/bin/view/Cintro/PythonString?sortcol=1&table=12&up=0#sorted_table) |
| [str.encode(encoding="utf-8", errors="strict")](https://docs.python.org/3/library/stdtypes.html#str.encode) | представляет строку str как последовательность байт |
| [bytes.decode(encoding="utf-8", errors="strict") bytearray.decode(encoding="utf-8", errors="strict")](https://docs.python.org/3/library/stdtypes.html#bytes.decode) | представляет последовательность байт bytes или bytearray как строку в выбранной кодировке |

*Wiki не позволяет правильно показать вам символы. Попробуйте выполнить код со строками у себя на компьютере.*

>>> artist = "Tage °Ase&#1026;n"

>>> artist.encode("Latin1")

b'Tage \xc5s\xe9n'

>>> artist.encode("CP850")

b'Tage \x8fs\x82n'

>>> artist.encode("utf8")

b'Tage \xc3\x85s\xc3\xa9n'

>>> artist.encode("utf16")

b'\xff\xfeT\x00a\x00g\x00e\x00 \x00\xc5\x00s\x00\xe9\x00n\x00'

>>> print(b"Tage \xc3\x85s\xc3\xa9n".decode("utf8"))

Tage °Ase&#1026;n

>>> print(b"Tage \xc5s\xe9n".decode("latin1"))

Tage °Ase&#1026;n

DEA! Чтобы создать строку в нужной кодировке, можно обойтись без создания промежуточного массива байт, указав кодировку в конструкторе [class str(object=b”, encoding='utf-8', errors='strict')](https://docs.python.org/3/library/stdtypes.html) вместо str(object='')

**Сравнения строк и кодировки**

При сравнении строки сравниваются побайтово. Это может привести к проблемам:

Символы Юникода могут быть представлены двумя и более последовательностями байтов.

Например, символ Юникода с кодом 0x00C5 в кодировке UTF-8 может быть представлен тремя различными способами: [0xE2, 0x84, 0xAB], [0xC3, 0x85] и [0x41, 0xCC, 0x8A].

Используйте unicodedata.normalize() со значением "NFKD" в первом аргументе (эта аббревиатура определяет способ нормализации "Normalization Form Compatibility Decomposition" - нормализация в форме совместимой декомпозиции)

Передав ей строку, содержащую символ 0x00C5, представленный любой из допустимых последовательностей байтов, мы получим строку с символами в кодировке UTF-8, где 0x00C5 всегда будет представлен последовательностью [0x41, 0xCC, 0x8A].

Нормализация редко когда требуется (даже если читаем из файлов / сокетов). О проблеме сортировки юникодных строк <http://www.unicode.org/reports/tr10/>

-- [TatyanaDerbysheva](http://acm.mipt.ru/twiki/bin/view/Main/TatyanaDerbysheva) - 21 Oct 2015