

Задачи Знакомство с индексами и срезами строк

Как вы уже знаете, в Python строку можно задавать, например, так:

`s = "hello python"`

индексы:	0	1	2	3	4	5	6	7	8	9	10	11
s =	h	e	l	l	o		p	y	t	h	o	n
	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

и она представляет собой упорядоченный набор символов. Что значит упорядоченный? Смотрите, каждый символ строки имеет свой уникальный, порядковый номер. Эти номера называются индексами. Первый – 0, второй – 1 и так до конца. Мы можем использовать эти индексы для обращения к отдельному символу строки. Для этого, записывается имя переменной и в квадратных скобках указывается номер символа:

`s[0]` Но будьте внимательны, если указать несуществующий индекс:
`s[1]`

`s[12]`

то получим ошибку выхода за пределы диапазона. В нашем случае последний индекс равен 11, а мы записали 12. Чтобы взять последний символ нам следовало бы записать: `s[11]`

То есть, последний индекс равен длине строки минус один: `len(s) - 1`

И, по идее, мы могли бы его так и указать в квадратных скобках: `s[len(s) - 1]`

Эта конструкция работает для строк произвольной длины (кроме нулевой, когда символов нет). Но в Python то же самое можно записать проще, используя отрицательные индексы:

`s[-1]`

То есть, длину строки `len(s)` можно опускать и записывать отрицательные значения. Получаем способ индексации с конца строки.

Причем индексацию можно выполнять также и непосредственно над строками, например: `"panda"[3]`

вернет 4-й символ этой строки. Иногда это тоже может быть полезно. Фактически, когда мы выполняем индексацию, то возвращается новая строка из одного выделенного символа. Но мы можем

выделять и сразу несколько символов, используя следующий синтаксис:

строка[start:stop]

Выделенная последовательность символов называется срезом строки.

Например, можно сделать так: `s[1:3]`

вернет два символа с индексами 1 и 2. Последнее значение 3 не включается в срез. В срезах можно не указывать последнее значение: `s[4:]`

или первое: `s[:5]` или оба: `s[:]`

В последнем случае возвращается та же самая строка. Убедиться в этом можно, следующим образом:

```
a = s[:]
id(a)
id(s)
```

В срезах также можно использовать отрицательные индексы, например: `s[2:-2]`

Но если записать: `s[-2:2]`

то получим пустую строку, так как эти индексы не образуют диапазон значений.

Наконец, в срезах дополнительно можно указывать еще и шаг перебора символов, согласно синтаксису: `строка[start:stop:step]`

Например: `s[2:10:2]`
`s[2::3]`
`s[:5:3]`
`s[::2]`

И использовать отрицательный шаг: `s[:: -1]`

тогда все символы будут перебираться в обратном порядке, начиная с последнего. Если же указать: то будут выбираться символы с конца через один. `s[:: -2]`

Изменение строк

При работе со строками следует иметь в виду, что она относится к неизменяемым типам данных, то есть, существующую строку изменить нельзя. В частности, из-за этого попытка присвоить строке какой-либо символ:

```
s[0] = 'H'
```

приведет к ошибке. Чтобы изменить строку, нужно создать новую с другим содержимым:

```
s2 = 'H' + s[1:]
```

И так происходит каждый раз, когда нужно изменить что-то в уже существующей строке.

Итак, из этого занятия вы должны хорошо себе представлять:

- строка – упорядоченный набор символов;
- как выполняется индексация к отдельным символам строки `str[index]`;
- как выделять из строк наборы символов – срезы;
- строка – неизменяемый объект;
- способ модификации (изменения) строк через индексы и срезы.