

Python Essential

Последовательности

Python Essential

После урока обязательно



Повторите этот урок в видео формате на [ITVDN.com](http://itvdn.com)



Проверьте как Вы усвоили данный материал на [TestProvider.com](http://testprovider.com)

Python Essential

Тема

Последовательности

Последовательности

Понятие последовательности

- **Последовательностью** в Python называется итерабельный объект, который поддерживает эффективный доступ к элементам с использованием целочисленных индексов через специальный метод `__getitem__()` и поддерживает метод `__len__()`, который возвращает длину последовательности. К основным встроенным типам последовательностей относятся `list`, `tuple`, `range`, `str` и `bytes`.
- Последовательности также опционально могут реализовывать методы `count()`, `index()`, `__contains__()` и `__reversed__()` и другие.



Последовательности

Общие операции

Операция	Описание
$x \text{ in } s, x \text{ not in } s$	находится ли элемент x в последовательности s
$s + t$	конкатенация последовательностей
$s * n, n * s$	конкатенация n неполных копий последовательности s
$s[i]$	i -й элемент последовательности s
$s[i:j], s[i:j:k]$	срез последовательности s от i до j с шагом k
$\text{len}(s)$	длина последовательности
$\text{min}(s)$	минимальный элемент последовательности
$\text{max}(s)$	максимальный элемент последовательности
$s.\text{index}(x[, i[, j]])$	индекс первого вхождения x (опционально – начиная с позиции i и до позиции j)
$s.\text{count}(x)$	общее количество вхождений x в s
$\text{sum}(s)$	$\text{sum}(s)$ – сумма элементов последовательности

Последовательности

Операции с изменяемыми последовательностями

Операция	Описание
<code>s[i] = x</code>	элемент с индексом <code>i</code> заменяется на <code>x</code>
<code>s[i:j] = t</code> , <code>s[i:j:k] = t</code>	элементы с индексами от <code>i</code> до <code>j</code> (с шагом <code>k</code>) заменяются содержимым итерабельного объекта <code>t</code>
<code>del s[i:j]</code> , <code>del s[i:j:k]</code>	удаление соответствующих элементов из последовательности
<code>s.append(x)</code>	добавление <code>x</code> в конец последовательности
<code>s.clear()</code>	удаление всех элементов последовательности
<code>s.copy()</code>	неполная копия последовательности
<code>s.extend(t)</code>	добавление всех элементов итерабельного объекта в конец последовательности
<code>s.insert(i, x)</code>	вставка элемента <code>x</code> по индексу <code>i</code>
<code>s.pop()</code> , <code>s.pop(i)</code>	возврат значения по индексу <code>i</code> (по умолчанию – последний) и удаление его из последовательности
<code>s.remove(x)</code>	удаление первого вхождения <code>x</code>
<code>s.reverse()</code>	разворот последовательности в обратном порядке

Последовательности

Списки

Списки – это изменяемые последовательности, обычно используемые для хранения однотипных данных (хотя Python не запрещает хранить в них данные разных типов). Представлены классом `list`

Создание списков:

```
my_list = []  
my_list = [0]  
my_list = [1, 2, 3, 5, 9, 0]  
my_list = [x ** 3 for x in range(10)]  
my_list = list(range(8))
```



Последовательности

Операции со списками

- Поддерживают все общие для всех последовательностей операции.
- Поддерживают общие для изменяемых последовательностей операции
- Реализуют один дополнительный метод:

`list.sort(self, *, key=None, reverse=None)`

Он сортирует список при помощи операции “<”. Опциональный параметр `key` – функция от одного аргумента, которая извлекает ключ для сортировки, `reverse` – сортировка в обратном порядке, если он равен `True`.



```
my_list.sort()  
my_list.sort(reverse=True)
```

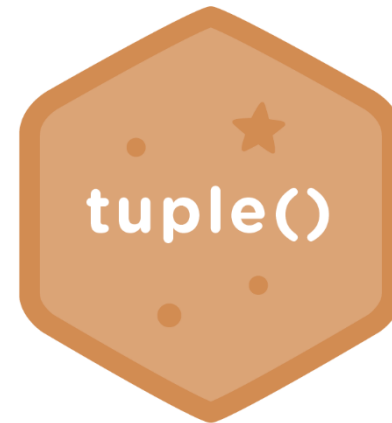

Последовательности

Кортежи

- **Кортежи** – это неизменяемые последовательности, обычно используемые, чтобы хранить разнотипные данные. Представлены классом `tuple`.
- Поддерживают все общие для последовательностей операции.

Создание кортежей:

```
my_tuple = ()  
my_tuple = (1,)   
my_tuple = 1,  
my_tuple = (1, 2, 'a string')  
my_tuple = 1, 2, 'a string'  
my_tuple = tuple(range(8))
```



Последовательности

Распаковка кортежей

```
a, b, c = 1, 2, 3
```

```
a, b = b, a
```

```
a, b, c = iterable
```

```
head, *tail = iterable
```

```
for index, value in enumerate(sequence):  
    pass
```



Последовательности

Функции с произвольным количеством аргументов. Распаковка аргументов функции

- Функция может иметь произвольное количество аргументов. После всех позиционных параметров функции или вместо них (но перед теми, которые предполагается использовать как именованные) в её сигнатуре можно указать специальный аргумент с символом * перед именем. Тогда оставшиеся фактические параметры сохраняются в кортеже с этим именем.
- Также существует и обратная возможность. Если при вызове функции перед именем итерабельного объекта поставить символ *, то его элементы распаковываются в позиционные аргументы.



Последовательности

Диапазоны

- **Диапазоны** – неизменяемые последовательности чисел, которые задаются началом, концом и шагом. Представлены классом `range` (в Python 2 – `xrange`; `range` в Python 2 – это функция, которая возвращает список).
- Начало по умолчанию равно нулю, шаг – единице. Если задать нулевой шаг, будет выброшено исключение `ValueError`.
- Параметры конструктора должны быть целыми числами (либо экземпляры класса `int`, либо любой объект с методом `__index__`).
- Элементы диапазона `r` определяются по формуле $r[i] = \text{start} + \text{step} * i$, где $i \geq 0$ и $r[i] < \text{stop}$ для $\text{step} > 0$ или $r[i] > \text{stop}$ для $\text{step} < 0$.
- Поддерживает все общие для последовательностей операции, кроме конкатенации и повторения, а также, в версиях Python до 3.2, срезов и отрицательных индексов.



Последовательности

Строки

- **Строки** – неизменяемые последовательности кодов символов (в Python 3 – в кодировке Unicode, в Python 2 – в ASCII). Представлены классом `str`. В Python 2 также есть класс `unicode`, который представляет Unicode-строки подобно `str` в Python 3.
- Строковые литералы выделяются одинарными или двойными кавычками. Можно использовать утроенные кавычки для создания многострочных строк. Если перед строковым литералом стоит префикс `r`, то большинство escape-последовательностей игнорируются. В Python 2 префикс `u` задаёт Unicode-литерал.
- Поддерживают все общие для последовательностей операции, а также реализуют огромное количество собственных методов.
- Функция `ord(char)` возвращает код символа `char`, а функция `chr(code)` возвращает символ с кодом `code`.



Последовательности

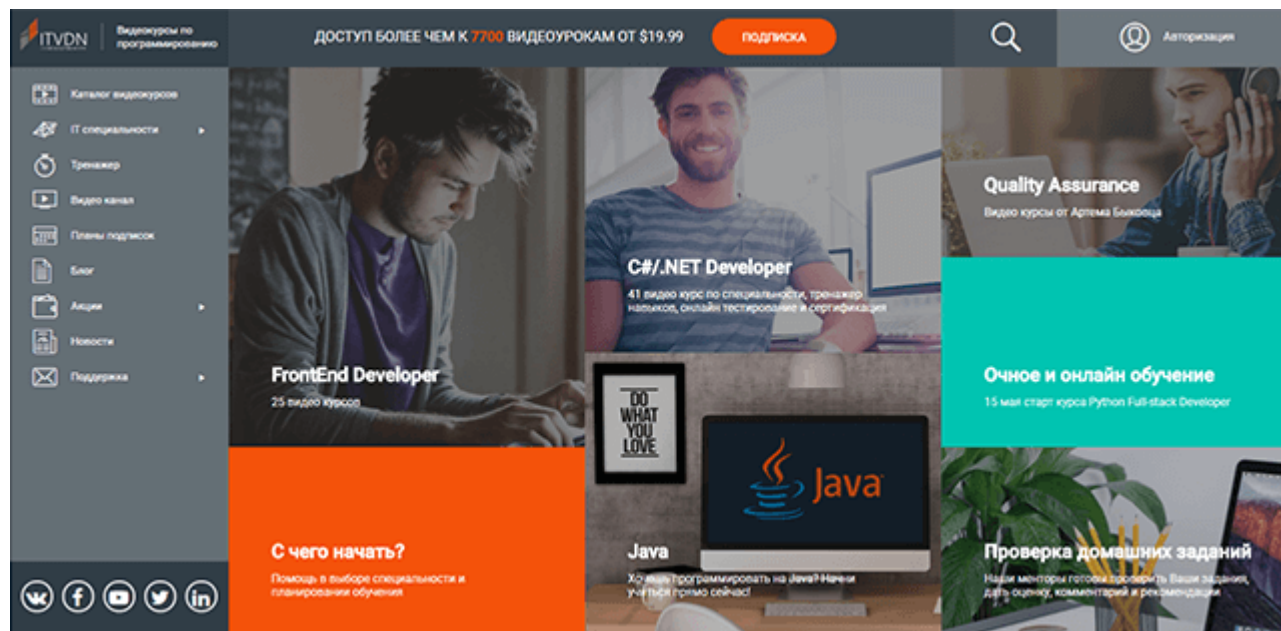
Сравнение последовательностей

- Две последовательности равны, если они имеют одинаковый тип, равную длину и соответствующие элементы обеих последовательностей равны.
- Последовательности одинаковых типов можно сравнивать. Сравнения происходят в лексикографическом порядке: последовательность меньшей длины меньше, чем последовательность большей длины, если же их длины равны, то результат сравнения равен результату сравнения первых отличающихся элементов.



Смотрите наши уроки в видео формате

ITVDN.com



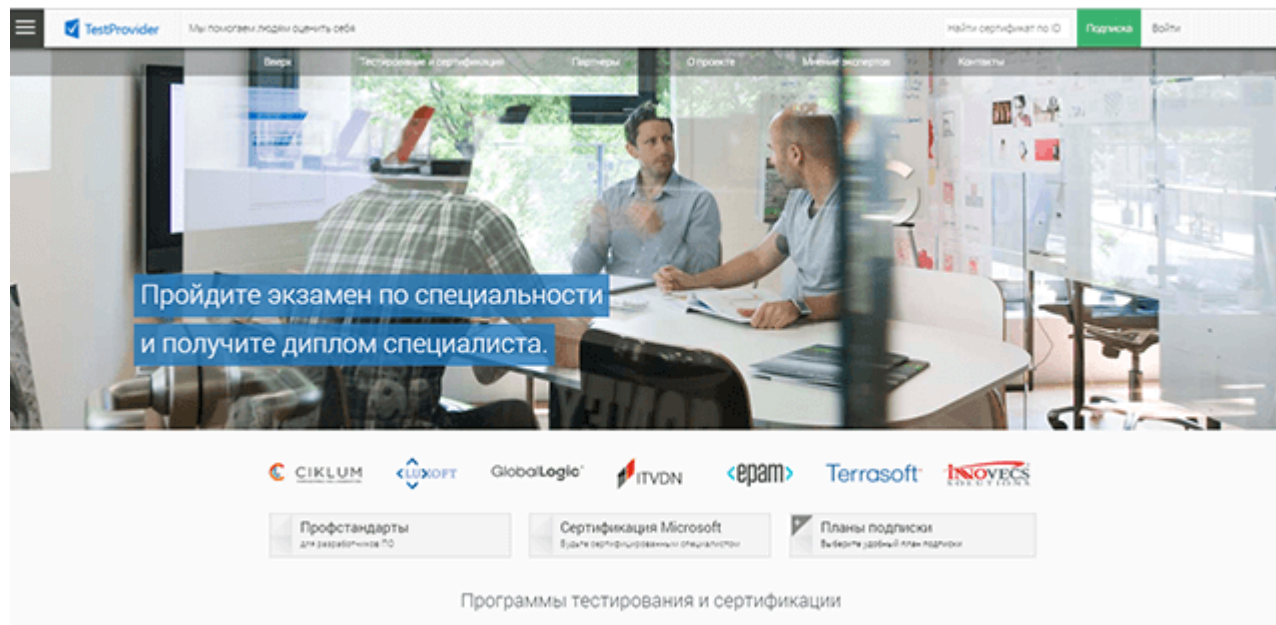
Посмотрите этот урок в видео формате на образовательном портале ITVDN.com для закрепления пройденного материала.

Курсы записаны сертифицированными тренерами, которые работают в учебном центре CyberBionic Systematics и другими высококвалифицированными разработчиками.



Проверка знаний

TestProvider.com



TestProvider – это online сервис проверки знаний по информационным технологиям. С его помощью Вы можете оценить Ваш уровень и выявить слабые места. Он будет полезен как в процессе изучения технологии, так и для общей оценки знаний IT специалиста.

После каждого урока проходите тестирование для проверки знаний на [TestProvider.com](https://testprovider.com)

Успешное прохождение финального тестирования позволит Вам получить соответствующий Сертификат.



Python Essential

Q&A

Информационный видеосервис для разработчиков программного обеспечения

