Последовательности

№ урока: 7 **Курс:** Python Essential

Средства обучения: PyCharm

Обзор, цель и назначение урока

После завершения урока обучающиеся будут иметь представление о последовательностях в Python и основных стандартных последовательностях, их назначении и использовании, смогут реализовывать собственные классы последовательностей.

Изучив материал данного занятия, учащийся сможет:

- Иметь представление о последовательностях
- Определять, какие контейнеры в Python считаются последовательностями и создавать свои последовательности
- Знать основные операции, общие для всех последовательностей и изменяемых последовательностей
- Работать со списками
- Работать с кортежами
- Работать с диапазонами
- Работать со строками

Содержание урока

1. Что такое последовательности?
2. Операции над последовательностями
3. Списки (list)
4. Строки (str)
5. Диапазон (range)
6. Проверка вхождения элемента

Резюме

Последовательность – это упорядоченная коллекция, поддерживающая индексированный доступ к элементам.
Некоторые последовательности в Python в отличие от традиционных массивов (например, в Си) могут хранить элементы различного типа (в том числе и коллекции различных видов). В языке Python имеется четыре встроенных типов последовательностей: list, range, str, tuple.

Операции над последовательностями
Некоторые итерабельные объекты обладают определёнными общими свойствами.
Итерабельные объекты, которые поддерживают эффективный доступ к элементам с использованием целочисленных индексов через специальный метод __getitem__() и поддерживают метод __len__(), который возвращает количество элементов, называются последовательностями.
Длина len(s): функция len() возвращает длину (количество элементов в последовательности) s.
Конкатенация (**склеивание**): s + t — возвращает новый объект-склейку s и t. Дублирование s * n`n * s— возвращает последовательность, повторяющуюся n` раз.
Индексация и срезы. Получить доступ к отдельному или группе элементов последовательности возможно с помощью оператора []. Индексацию (получение отдельного элемента) можно считать частным случаем получения среза (**слайсинга**).
Оператор получения среза имеет три формы записи:

- s[start] - индексация (с 0);
- s[start:end] - срез **[start; end)**

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- s[start:end:step] - срез [start; end) с шагом step.

В ряде случаев целочисленные параметры start, end и step могут быть опущены. Элемент с индексом end **не включается** в результат при взятии срезов.
Минимальное и максимальное значения min(s) и max(s) — возвращает минимальный и максимальный элементы последовательности s соответственно. Проверка на вхождение x in s — возвращает True, если x входит в последовательность s и False в противном случае.
Индекс (положение) элемента s.index(x[, start[, end]]) --> int — возвращает первое вхождение элемента x в последовательность s (между индексами start и end, если они заданы).
Количество повторений s.count(x) — возвращает количество вхождений элементов x в последовательность s.
Сортировка sorted(iterable, key=None, reverse=False) — возвращает отсортированный объект в виде списка. Исходный объект при этом не изменяется.
Параметры:
key – функция сортировки (по умолчанию не учитывается, сортировка осуществляется поэлементно)
reverse – если равен True, сортировка осуществляется в обратном порядке.
Ранее уже были рассмотрены три типа данных, которые являются последовательностями.

Списки
Список (list) - это упорядоченная изменяемая последовательность элементов. Особенности:

- может содержать элементы разного типа;
- поддерживает операторы сравнения: при этом сравнивание производится поэлементно (и рекурсивно, при наличии вложенных элементов).

С помощью цикла for мы можем итерировать список:

```
my_list = [1, 2, 3]

for element in my_list:
    print(element)
```

Индексирование позволяет получить один конкретный элемент со списка:

```
my_list = [1, 2, 3]

print(my_list[0])
print(my_list[2])
print(my_list[-1])
print(my_list.__getitem__(-1))
```

С помощью встроенной функции len() можно получить длину последовательности:

```
my_list = [1, 2, 3]

print('Length:', len(my_list))
print('Length:', my_list.__len__())
```

Строки
Строка (str) - это упорядоченная неизменяемая последовательность символов Юникода.
Литералы строк создаются с использованием кавычек или апострофов, при этом важно, чтобы с обоих концов литерала использовались кавычки одного и того же типа. Также можно использовать строки, которые начинаются и заканчиваются тремя символами кавычки.
Важным для строкового типа является понятие кодировки символов, что в частности, влияет на правила сравнения строк. По умолчанию Python хранит строки в кодировке UTF-8.
Если в строке необходимо использовать специальные символы (например, перенос или одноименные кавычки), можно воспользоваться механизмом экранирования символов, для чего используется специальный символ \. В Таблице приведены некоторые из экранированных последовательностей:

CyberBionic systematics
CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Специальные символы

Последовательность	Значение
****	Обратный слеш ()
\'	Апостроф (')
"\""	"Кавычка ("")"
\n	Символ «Перевод строки»
\t	Символ «Табуляция»

Повторим операции выше для строк:

```
string = "Lorem ipsum dolor sit amet, consectetur adipisicing elit."

# Итерирование
for character in string:
    print(character)

# Получение доступа к элементам при помощи целочисленных ключей (индексация)
print(string[0])
print(string[2])
print(string[-1])

# Длина последовательности
print('Length:', len(string))
```

Диапазоны

Числовой диапазон (range) - это упорядоченная неизменяемая последовательность элементов - целых чисел.

```
my_range = range(10)

# Итерирование
for element in my_range:
    print(element)

# Получение доступа к элементам при помощи целочисленных ключей (индексация)
print(my_range[0])
print(my_range[2])
print(my_range[-1])

# Длина последовательности
print('Length:', len(my_range))
```

Числовые диапазоны поддерживают те же операции, что и кортежи.

Кортежи

Кортеж (tuple) – это упорядоченная **неизменяемая** последовательность элементов.
Особенности:

- Умеет все, что умеет список, за исключением операций, приводящих к изменению кортежа.
- Применяется в случаях, когда известно, что последовательность не будет меняться после создания.

Проверка вхождения элемента

Большинство последовательностей поддерживают операции проверки вхождения элемента in и not in:

```
>>> 4 in [1, 5, 4]
True
```


CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

```
>>> 40 in range(30, 100)
True
>>> "i" in "ispum"
True
>>> 4 not in [3, 4, 5]
False
>>>
```

Для поддержки данной операции необходимо реализовать специальный метод __contains__:

```
# Реализация поддержки операции
class Container(object):
    def __contains__(self, element):
        return element == 3

container = Container()
print(3 in container)
print(5 in container)
```

Закрепление материала

- Что такое последовательность?
- Какие методы обязана реализовать любая последовательность?
- Какие методы реализуют большинство последовательностей?
- Какие методы реализуют большинство изменяемых последовательностей?
- Какие вы знаете встроенные неизменяемые последовательности в Python?
- Какие вы знаете встроенные изменяемые последовательности в Python?
- Что такое кортежи?
- Как реализуется передача произвольного количества аргументов функции в Python?

Дополнительное задание

Задание
Напишите программу, которая вводит с клавиатуры последовательность чисел и выводит её отсортированной в порядке возрастания.

Самостоятельная деятельность учащегося

Задание 1
Создайте функцию от произвольного количества аргументов, которая вычисляет среднее арифметическое данных чисел. Вычислите при помощи неё среднее арифметическое двух заданных чисел и среднее арифметическое чисел из заданного диапазона.

Задание 2
Используя документацию, ознакомьтесь с методами класса str.

Задание 3
Напишите программу, которая вводит с клавиатуры текст и выводит отсортированные по алфавиту слова данного текста.

Задание 4
Ознакомьтесь при помощи документации с классами namedtuple и deque модуля collections.

Рекомендуемые ресурсы

Документация Python
https://docs.python.org/3/glossary.html#term-sequence

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

https://docs.python.org/3/tutorial/datastructures.html
https://docs.python.org/3/library/stdtypes.html#sequence-types-list-tuple-range
https://docs.python.org/3/library/stdtypes.html#common-sequence-operations
https://docs.python.org/3/library/stdtypes.html#immutable-sequence-types
https://docs.python.org/3/library/stdtypes.html#mutable-sequence-types
https://docs.python.org/3/library/stdtypes.html#lists
https://docs.python.org/3/library/stdtypes.html#tuples
https://docs.python.org/3/library/stdtypes.html#ranges
https://docs.python.org/3/library/stdtypes.html#text-sequence-type-str
https://docs.python.org/3/library/stdtypes.html#string-methods
https://docs.python.org/3/tutorial/controlflow.html#arbitrary-argument-lists
https://docs.python.org/3/tutorial/controlflow.html#unpacking-argument-lists
https://docs.python.org/3/library/collections.html

Статьи в Википедии о ключевых понятиях, рассмотренных на этом уроке
https://ru.wikipedia.org/wiki/Список_(информатика)
https://ru.wikipedia.org/wiki/Строковый_тип
https://ru.wikipedia.org/wiki/Списковое_включение

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com