# Практическая работа №3\_5

# Основы Python, часть 2

## Цель работы

Продолжение изучения базового синтаксиса Python.

## Задачи работы

1. Изучить итеративные типы данных Python: списки, кортежи, словари.
2. Изучать основные операции с итеративными типами данных.

## Перечень обеспечивающих средств

1. ПК.
2. Учебно-методическая литература.
3. Задания для самостоятельного выполнения.

## Общие теоретические сведения

**Итеративные типы данных Python**

В дополнение в спискам, рассмотренным в предыдущей работе, имеет смысл рассмотреть ещё два важных итеративных типа данных: кортежи и словари.

*Кортеж* (tuple) — заключенная в круглые скобки последовательность значений любых типов, разделенных запятыми.

Примеры кортежей:

|  |
| --- |
| (1, 'текст', 2, -0.19)  ((0, 1), 2, 'три') |

Основное отличие кортежей и списков: кортежи нельзя изменять и дополнять, списки — можно.

Список можно преобразовать в кортеж с помощью метода tuple() и, наоборот, котреж в список с помощью метода list().

*Словарь* (dict) — заключенная в фигурные списки последовательность пар ключ-значение. Ключ и значение разделяются двоеточием, пары между собой — запятыми.

Примеры словарей:

|  |
| --- |
| {1 : 1, 2 : 'два', 'три' : 3, 'четыре' : 'четыре'}  {'Арсенал' : 'Тула', 'Зенит' : 'Санкт-Петербург', 'Спартак' : 'Москва'} |

Ключи в словаре не могут повторяться, значения — могут.

Ключами могут выступать только неизменяемые значения, например, числа, строки, кортежи. Значения могут быть любыми, в том числе другими словарями.

**Основные операции со списками, кортежами и словарями**

Рассмотрим основные операции на следующих примерах:

|  |
| --- |
| my\_list = [1, 2, 4, 8, 16]  my\_tuple = (10, 'десять', 'X')  my\_dict = {3 : 'три', 2 : 'два', 1 : 'один'} |

*Индексация:*

|  |  |  |
| --- | --- | --- |
| Список | Кортеж | Словарь |
| Можно извлечь элемент по его номеру, начиная с нуля:  my\_list[1] равно 2.  Либо по номеру с конца:  my\_list[-1] равно 16. | Можно извлечь элемент по его номеру, начиная с нуля:  my\_tuple[1] равно 'десять'.  Либо по номеру с конца:  my\_tuple[-1] равно 'X'. | Можно извлечь значение по ключу:  my\_dict[1] равно 'один'. |

*Добавление элемента:*

|  |  |  |
| --- | --- | --- |
| Список | Кортеж | Словарь |
| Можно добавить новый элемент в конец списка:  my\_list.append(32)  Теперь my\_list равен  [1, 2, 4, 8, 16, 32]  Можно добавить новый элемент в любую позицию списка:  my\_list.insert(1, 32)  После этого my\_list будет равен  [1, 32, 2, 4, 8, 16] | Нельзя изменить. | Можно добавить новый элемент, указав пару ключ-значение:  my\_dict[0] = 'ноль'  После этого my\_dict будет равен {3 : 'три', 2 : 'два', 1 : 'один', 0 : 'ноль'} |

*Удаление элемента:*

|  |  |  |
| --- | --- | --- |
| Список | Кортеж | Словарь |
| Метод pop() возвращает элемент с указанными индексом, удаляя его из списка. Если индекс не указан, то возвращается и удаляется последний элемент.  Например, my\_list.pop(2) вернет 4, а my\_list после этого будет равен [1, 2, 8, 16].  Метод remove() удаляет из списка указанный элемент.  Например, после выполнения my\_list.remove(8) my\_list будет равен [1, 2, 4, 16]. | Нельзя изменить. | Метод pop() возвращает значение с указанными ключом, удаляя пару ключ-значение из словаря.  Например, my\_dict.pop(3) вернет 'три', а my\_dict после этого будет равен {2 : 'два', 1 : 'один'}. |

*Поиск элемента:*

|  |  |  |
| --- | --- | --- |
| Список | Кортеж | Словарь |
| Оператор in позволяет проверить, содержится ли элемент в списке.  Например,  2 in my\_list вернёт True (истина),  5 in my\_list вернёт False (ложь). | Оператор in позволяет проверить, содержится ли элемент в кортеже.  Например,  10 in my\_tuple вернёт True (истина),  'XI' in my\_tuple вернёт False (ложь). | Оператор in позволяет проверить, содержится ли ключ в словаре.  Например,  3 in my\_dict вернёт True (истина),  'три' in my\_ dict вернёт False (ложь). |

*Сортировка:*

|  |  |  |
| --- | --- | --- |
| Список | Кортеж | Словарь |
| Метод sort() сортирует содержимое списка.  Пример вызова: my\_list.sort().  Метод sorted() возвращает новый отсортированный список.  Пример вызова: sorted(my\_list). | Метод sorted() возвращает новый отсортированный кортеж.  Пример вызова: sorted(my\_tuple). | Метод sorted() возвращает новый отсортированный список ключей словаря.  Пример вызова: sorted(my\_dict). |

*Слияние:*

|  |  |  |
| --- | --- | --- |
| Список | Кортеж | Словарь |
| Метод extend() добавляет к первому списку элементы второго списка.  Например, для l1 = [0,1] и l2 = [2, 3] после выполнения l1.extend(l2) l1 равен [0,1,2,3].  Оператор + создает новый список, состоящий из элементов первого и второго списков.  Например, для l1 = [0,1] и l2 = [2, 3] l1+l2 равно [0,1,2,3]. Сами l1 и l2 при этом не изменяются. | Оператор + создает новый кортеж, состоящий из элементов первого и второго кортежей.  Например, для t1 = (0, 1) и t2 = (2, 3) t1+t2 равно (0,1,2,3). Сами t1 и t2 при этом не изменяются. | Метод update() добавляет переданные пары ключ‑значение к словарю.  Например, для d1 = {0:0, 1:1} и d2 = {2:2, 3:3} после выполнения d1.update(d2) d1 будет равен {0:0, 1:1, 2:2, 3:3}. |

*Получение длины:*

|  |  |  |
| --- | --- | --- |
| Список | Кортеж | Словарь |
| Метод len возвращает количество элементов в списке.  Например, len(my\_list) вернет 5. | Метод len возвращает количество элементов в кортеже.  Например, len(my\_tuple) вернет 3. | Метод len возвращает количество элементов в словаре.  Например, len(my\_dict) вернет 3. |

*Копирование:*

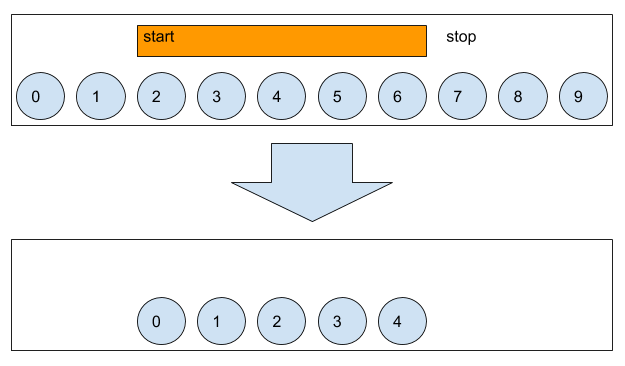
|  |  |  |
| --- | --- | --- |
| Список | Кортеж | Словарь |
| Метод copy() создает копию списка.  Пример вызова: my\_list.copy(). | Не имеет особого смысла, т. к. кортеж неизменяем. Два кортежа с одними и теми же значениями — по сути один и тот же кортеж. | Метод copy() создает копию словаря.  Пример вызова: my\_dict.copy(). |

*Слайсинг:*

Слайсинг — это более мощный аналог индексирования. Слайсинг применим к спискам, кортежам и строкам.

Нотация слайсинга аналогична нотации функции range():

start : stop : step – начало : остановка : шаг.



Start (начало):

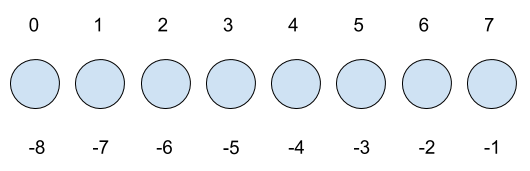
Если не указать, то считается равным 0.

Если указать отрицательное число, то элемент берется с конца списка или кортежа.

Stop (остановка):

Если не указать, то считается равным длине списка или кортежа.

Если указать отрицательное число, то элемент берется с конца списка или кортежа.



Step (шаг):

Число, на которое увеличивается индекс при проходе по списку или кортежу.

Если не указать, то считается равным 1.

Если указать отрицательное число, то отсчёт идет от конца к началу.

Слайсы можно использовать в большинстве методов или операторов, требующих указания индекса.

## Задание

**Часть 1**

* Сделайте форк репозитория

<https://github.com/mosalov/EmptyJupyterNoteebookForBinder>

* Откройте сайт Binder: <https://mybinder.org/>.
* В поле «GitHub repository name or URL» укажите ссылку на свой репозиторий. Нажмите кнопку «launch», дождитесь открытия репозитория.
* Откройте (кликните) файл «[empty\_notebook.ipynb](https://hub.gke.mybinder.org/user/mosalov-emptyju-eebookforbinder-wu4mdwz8/notebooks/empty_notebook.ipynb#_blank)».
* Напишите код, который создаст и выведет список квадратов натуральных чисел от 1 до 1000.
* Сохраните файл Jupyter notebook с названием «*Фамилия*\_Задание 3\_5\_1.ipynb» и загрузите его в созданный репозиторий.

**Часть 2**

* Вернитесь к файлу «[empty\_notebook.ipynb](https://hub.gke.mybinder.org/user/mosalov-emptyju-eebookforbinder-wu4mdwz8/notebooks/empty_notebook.ipynb#_blank)», открытому в Binder.
* Создайте кортеж из натуральных чисел от 1 до 20.
* С помощью слайсинга создайте кортеж, содержащий числа с 5 по 15 включительно.
* С помощью слайсинга создайте кортеж, содержащий последние 7 элементов первоначального кортежа.
* С помощью слайсинга создайте кортеж (7, 9, 11, 13).
* Сохраните файл Jupyter notebook с названием «*Фамилия*\_Задание 3\_5\_2.ipynb» и загрузите его в созданный репозиторий.

**Часть 3**

* Вернитесь к файлу «[empty\_notebook.ipynb](https://hub.gke.mybinder.org/user/mosalov-emptyju-eebookforbinder-wu4mdwz8/notebooks/empty_notebook.ipynb#_blank)», открытому в Binder.
* Создайте список, состоящий из букв вашего имени (с помощью метода list('строка') или любым другим удобным способом).
* Создайте копию списка (второй список).
* Отсортируйте второй список.
* Создайте третий список, который состоит из элементов первого и второго списков.
* Из третьего списка удалите первые 3 элемента.
* Выведите получившиеся первый, второй и третий списки.
* Сохраните файл Jupyter notebook с названием «*Фамилия*\_Задание 3\_5\_3.ipynb» и загрузите его в созданный репозиторий.

**Часть 4**

* Вернитесь к файлу «[empty\_notebook.ipynb](https://hub.gke.mybinder.org/user/mosalov-emptyju-eebookforbinder-wu4mdwz8/notebooks/empty_notebook.ipynb#_blank)», открытому в Binder.
* Создайте пустой словарь.
* Добавьте в словарь элементы «1 : 10», «'abc' : 'def'», «0 : [0, 1, 2]», «(0, 1) : 'а'».
* Прибавьте 5 к значению 10.
* Добавьте к строке 'def' ее копию (должно получиться 'defdef').
* В список [0, 1, 2] добавьте элемент 3, а затем удалите элемент 0.
* Удалите элемент со значением 'a'.
* Выведите получившийся словарь.
* Сохраните файл Jupyter notebook с названием «*Фамилия*\_Задание 3\_5\_4.ipynb» и загрузите его в созданный репозиторий.

## Контрольные вопросы

1. Какой тип данных: список, кортеж или словарь — вы считаете правильным использовать для указанных ниже данных и почему:
   1. список студентов в группе,
   2. список оценок студента,
   3. данные студента: ФИО, адрес, дата рождения.
2. Напишите функцию, которая принимает на вход список натуральных чисел и возвращает словарь, в котором ключами являются элементы входного списка, а значениями — кортежи простых чисел, меньших или равных ключу.

## Требования к отчету

Все файлы загрузите в свой репозиторий, созданный в практическом задании №1, по пути: «Notebook\_For\_AI\_Main/2021 Осенний семестр/Практическое задание 3\_5/» и сделайте пул-реквест.

## Литература

1. <https://pythonworld.ru/samouchitel-python>
2. <https://www.coursera.org/learn/diving-in-python>
3. <https://stepik.org/course/67/promo>
4. <http://pythontutor.ru/>