нетология

Составные типы данных, циклы

Гежин Олег

Олег Гежин

Python - разработчик, специалист SQL, фрилансер.

План занятия

1. Простые типы данных
2. Списки
3. Кортежи
4. Циклы

Простые типы данных

Простые типы данных

int / integer	float	str / string	bool / boolean
целые числа	действительные числа	строки	логический тип
`number = 10`	`q = 9.8`	`name = 'Коля'`	`sun = True`

Простые типы данных

int / integer	float	str / string	bool / boolean
целые числа	действительные числа	строки	логический тип
`number = 10`	`q = 9.8`	`name = 'Коля'`	`sun = True`

Тип объекта можно узнать при помощи функции `type()`.

Тип данных можно принудительно изменить функциями `int()`, `float()`, `bool()`, `str()` и т.д.

Индексация и срезы строк

Доступ к элементам объекта по их порядковому номеру в нем.

Индексация элементов начинается с нуля

Индексация и срезы строк

Доступ к элементам объекта по их порядковому номеру в нем.

Индексация элементов начинается с нуля.

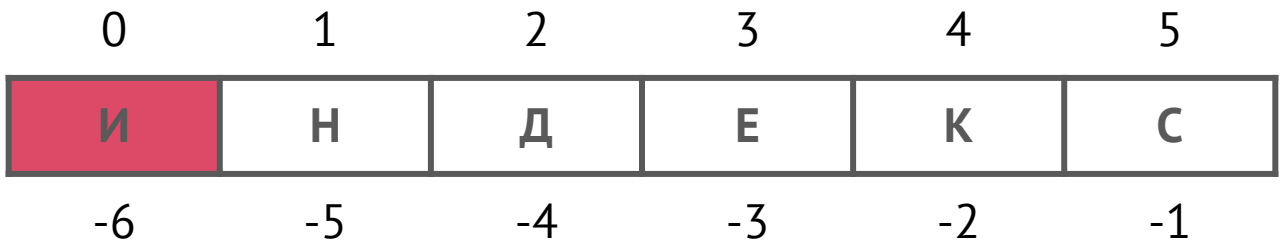
Поиск символов

Получить значение элемента по индексу можно при помощи `[ ]`.

`my_string[0]`

или

`my_string[-6]`



Срез строк

Можно извлечь из строки несколько элементов при помощи “срезов” (slicing). Для указания интервала среза используется `:`

Срез строк

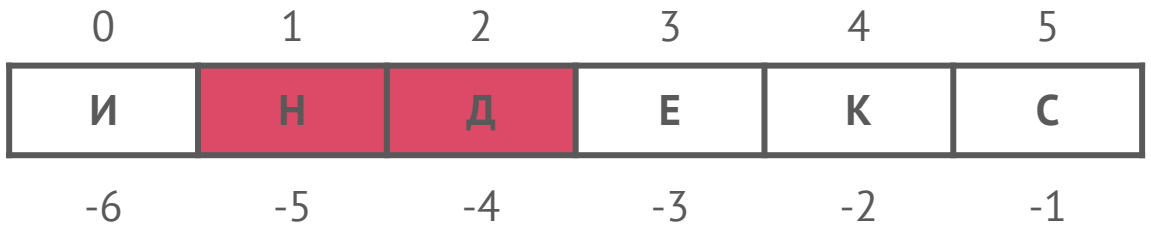
Можно извлечь из строки несколько элементов при помощи “срезов” (slicing). Для указания интервала среза используется `:`.

Синтаксис: `string[start:stop]`

`start` – индекс первого элемента в списке

`stop` – индекс списка, перед которым срез должен закончиться

`my_string[1:3]`



Срез строк

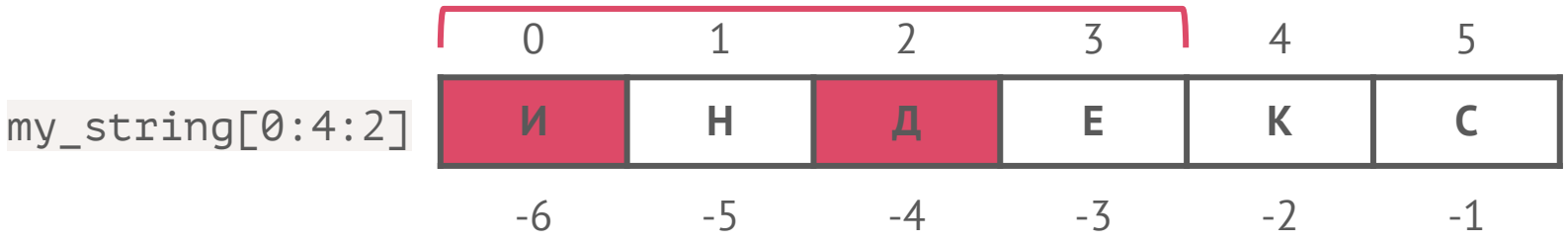
Срез с шагом. Шаг указывает, на сколько символов нужно подвинуться после взятия первого символа

Срез строк

Срез с шагом. Шаг указывает, на сколько символов нужно подвинуться после взятия первого символа.

Синтаксис: `string[start:stop:step]`, где

`step` – шаг прироста выбираемых индексов.



Срез строк

`string[start:stop:step]`

Любой из параметров может быть опущен. Тогда вместо соответствующего параметра будет выбрано значение по умолчанию

Срез строк

`string[start:stop:step]`

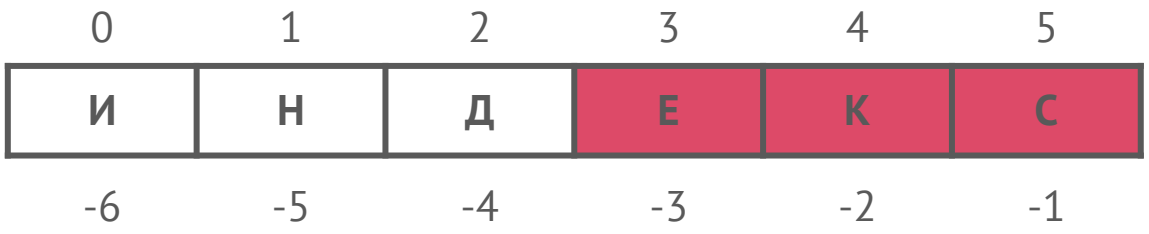
Любой из параметров может быть опущен. Тогда вместо соответствующего параметра будет выбрано значение по умолчанию

Значения по-умолчанию:

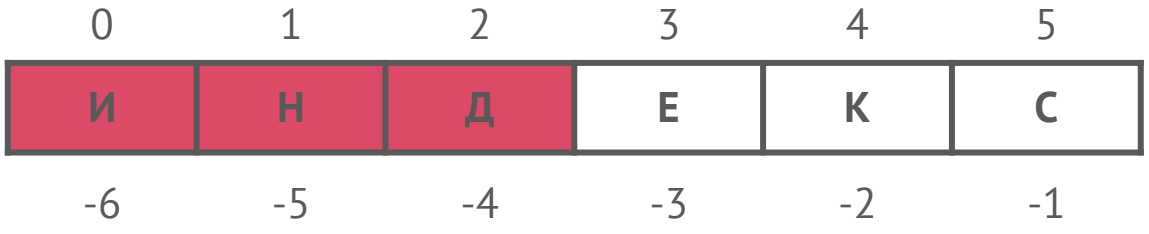
- `start` — «от начала списка»
- `stop` — «до конца списка» (включительно)
- `step` — «брать каждый элемент».

Срез строк

`my_string[3:]`



`my_string[:3]`



Списки

Списки (list)

Структура данных для **упорядоченного** хранения объектов **различных** типов.

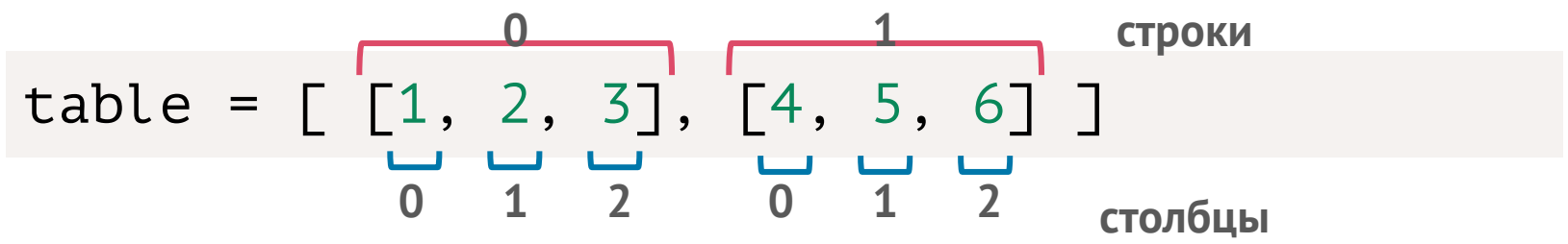
Последовательность элементов в списке начинается с 0

0	1	2	3
‘Петров’	**‘Николай’**	**‘Иванович’**	**25**

Список “Данные пользователя”

Многомерные списки

Внутри одного списка могут быть другие списки.



Многомерные списки

Многомерный список в виде таблицы:

	0	1	2
0	**1**	**2**	**3**
1	**4**	**5**	**6**

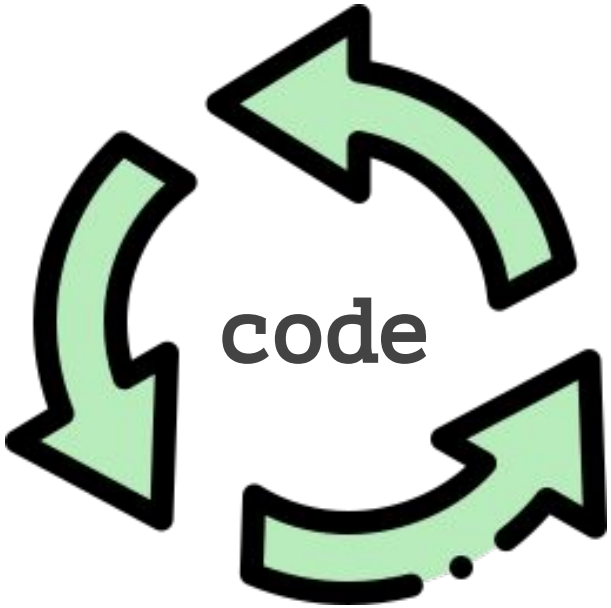
Кортежи

Циклы

Циклы

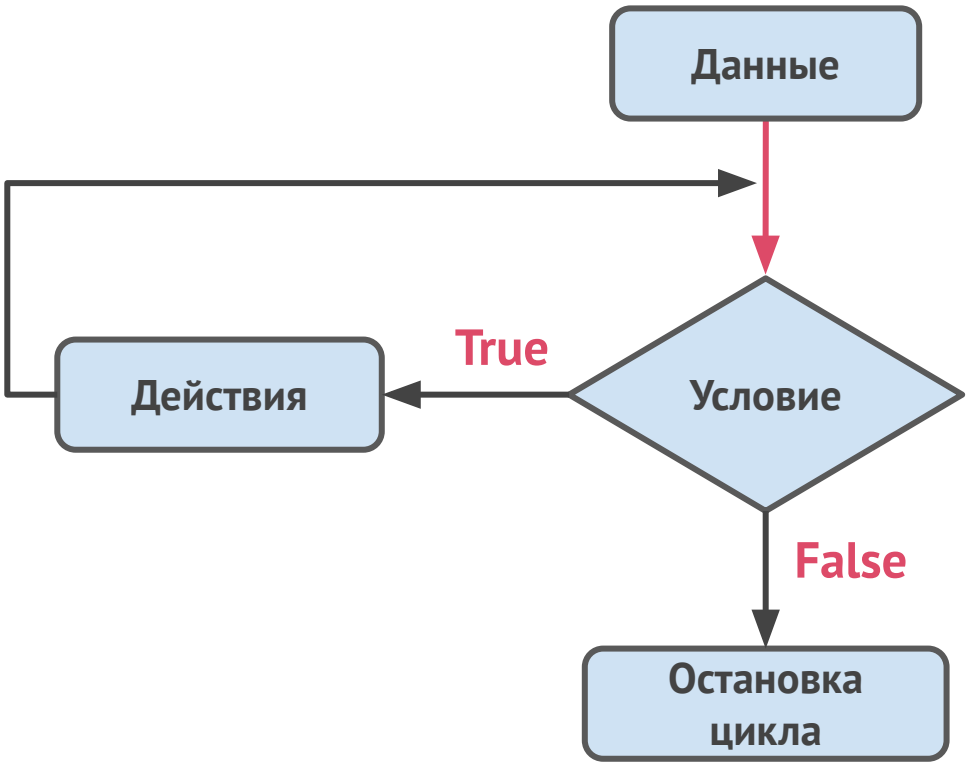
Циклы позволяют организовать повторение выполнения участков кода.

В Python существует два типа циклов: цикл `while` и цикл `for`.



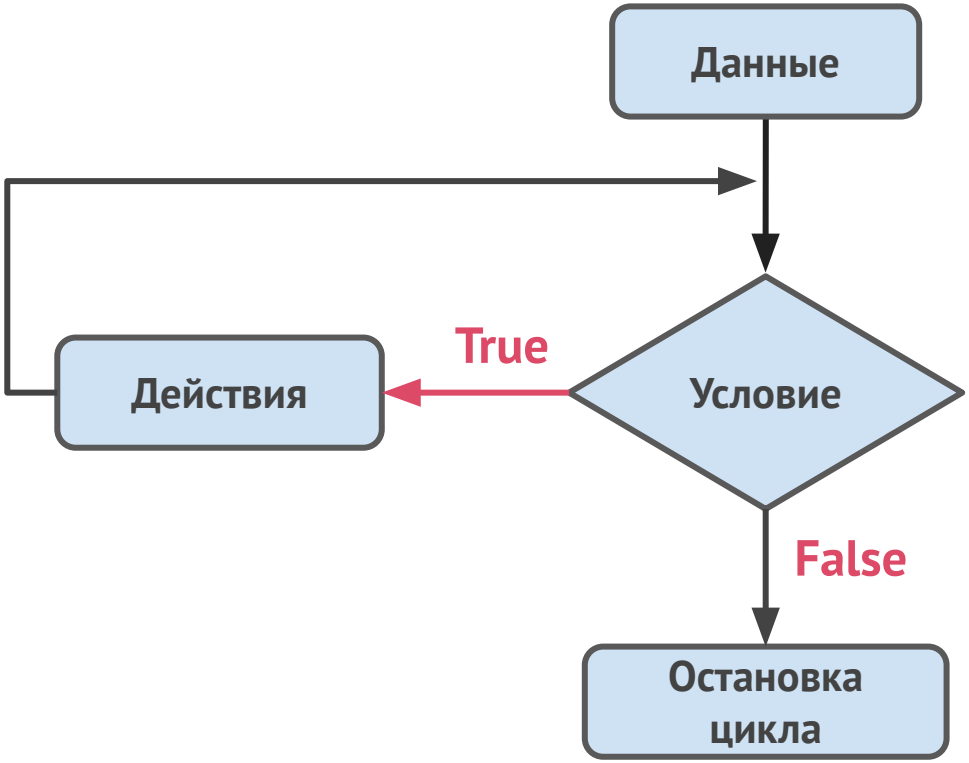
Цикл `while`

Позволяет выполнить одну и ту же последовательность действий, пока проверяемое условие истинно.



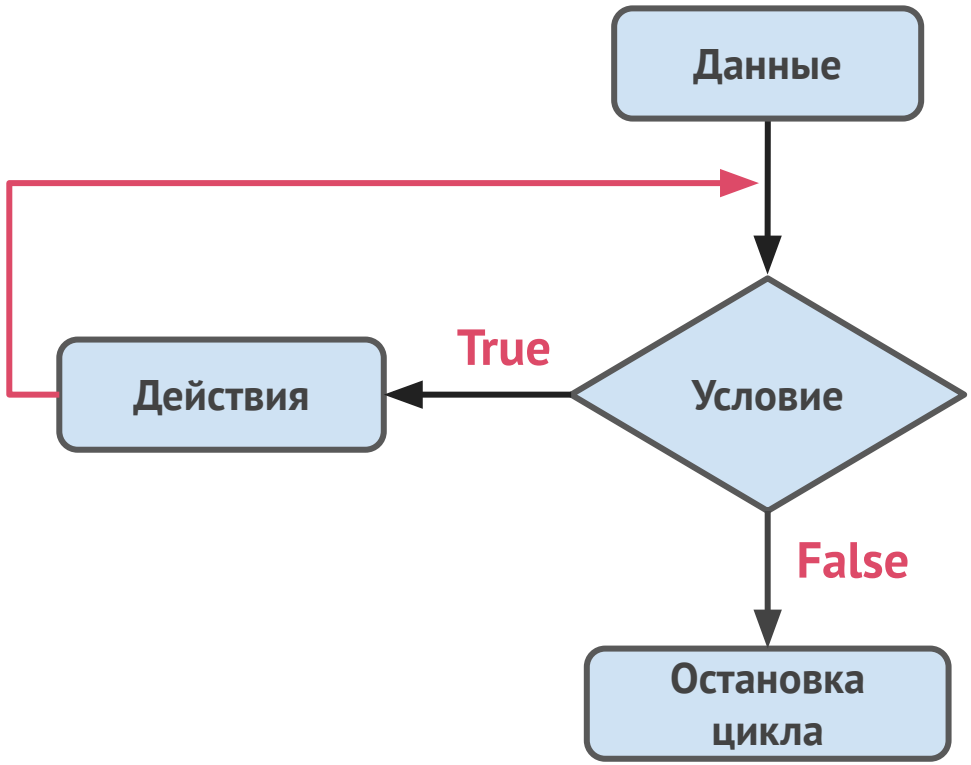
Цикл `while`

Позволяет выполнить одну и ту же последовательность действий, пока проверяемое условие истинно.



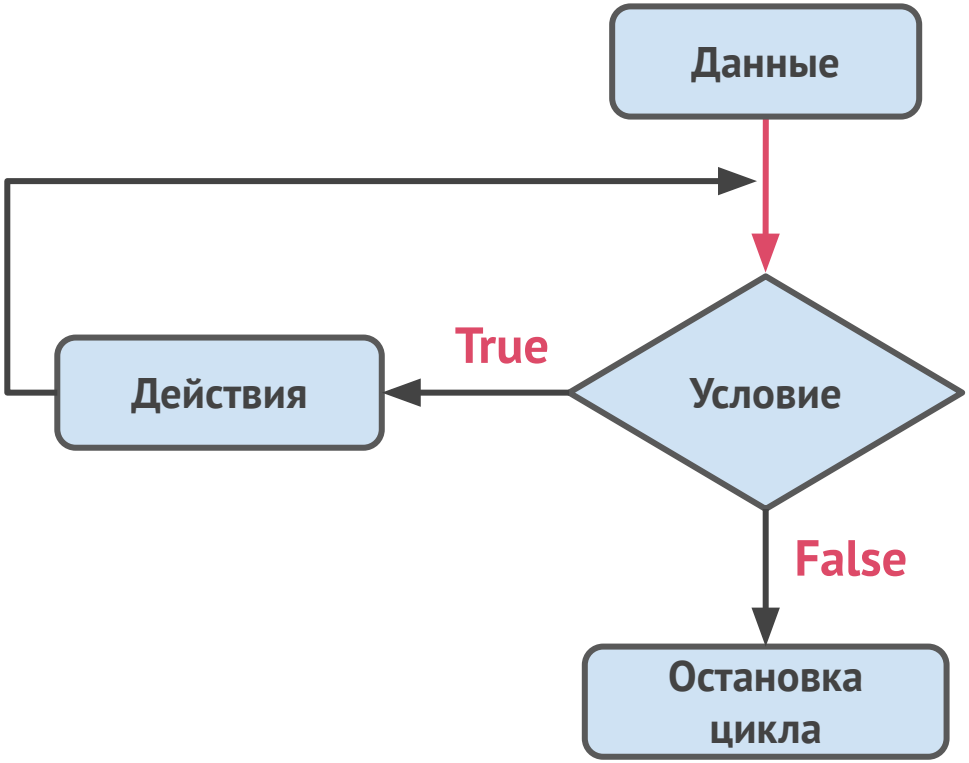
Цикл `while`

Позволяет выполнить одну и ту же последовательность действий, пока проверяемое условие истинно.



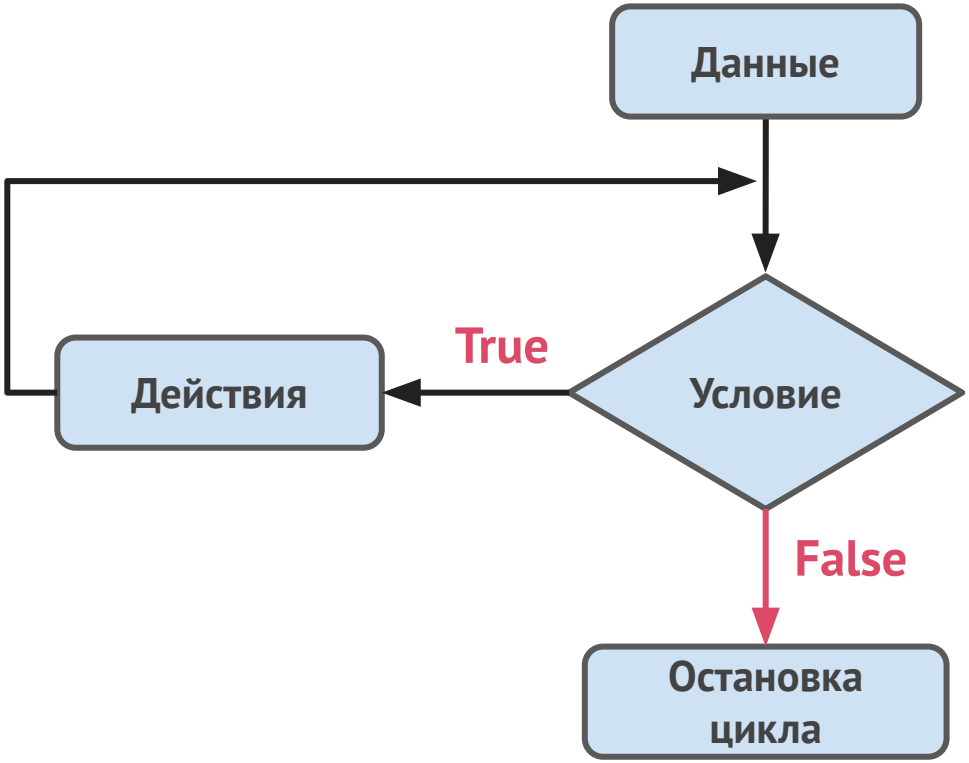
Цикл `while`

Позволяет выполнить одну и ту же последовательность действий, пока проверяемое условие истинно.



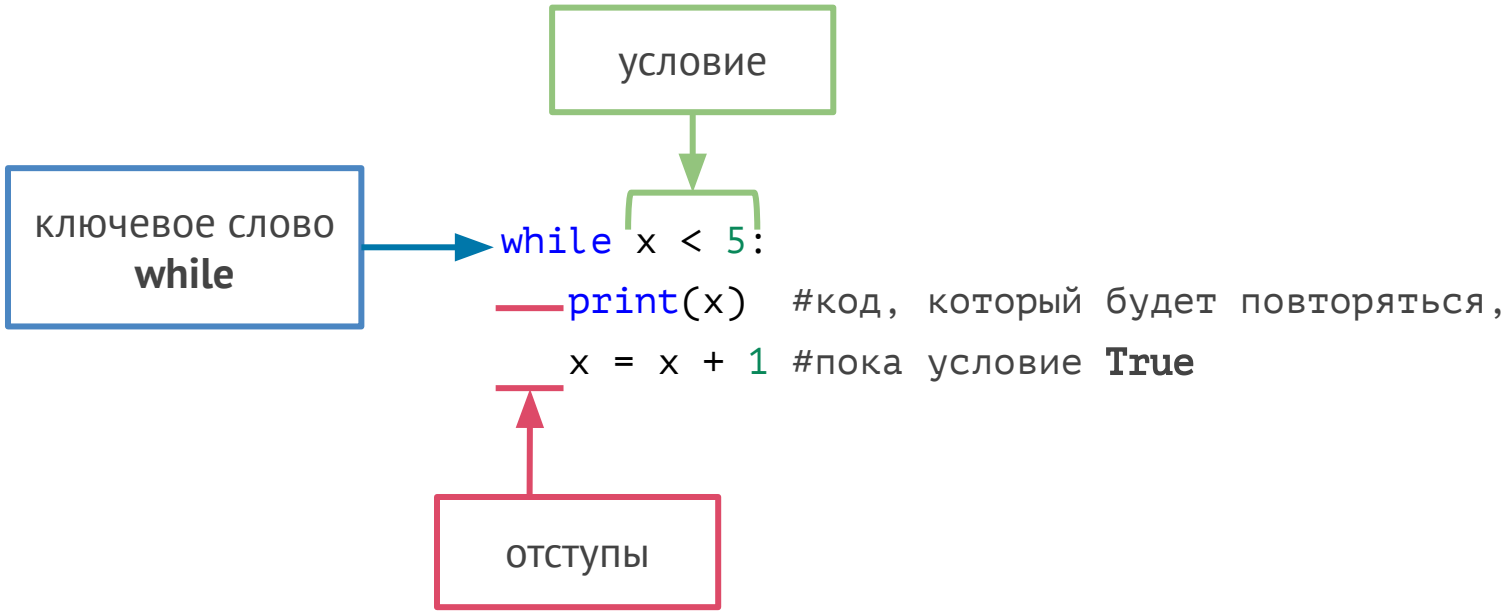
Цикл `while`

Позволяет выполнить одну и ту же последовательность действий, пока проверяемое условие истинно.



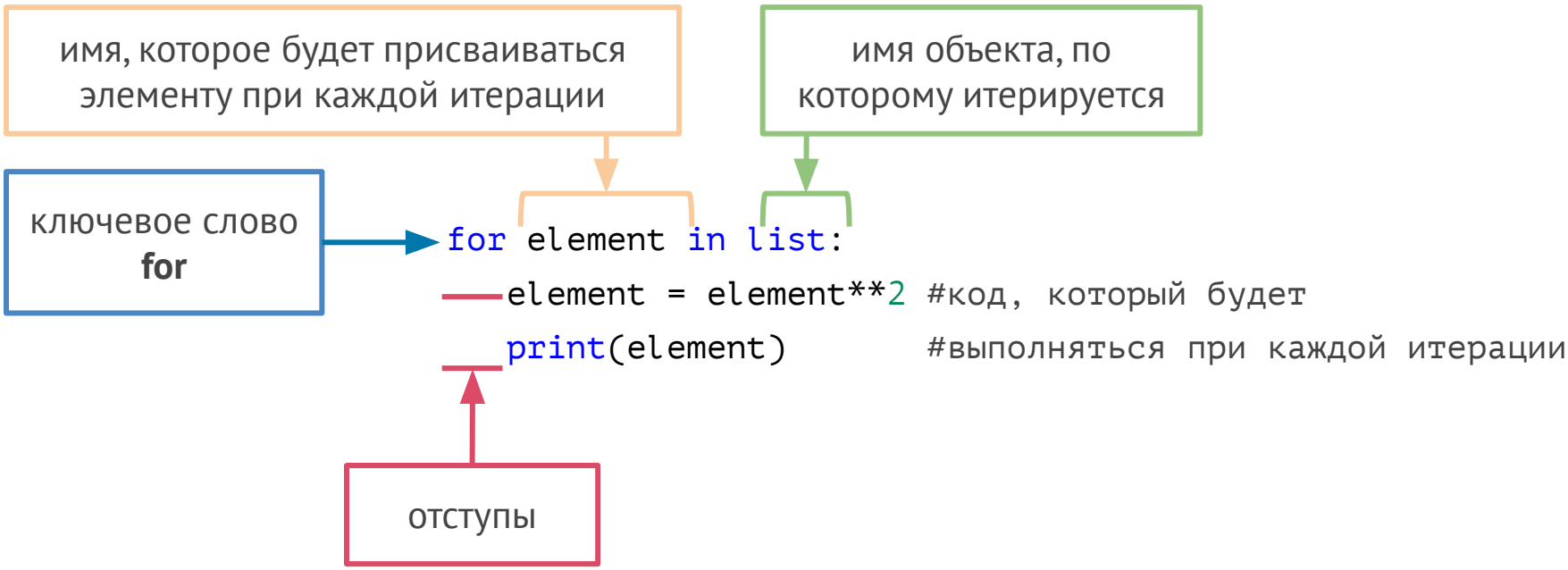
Цикл `while`

Используется, когда невозможно заранее определить точное значение количества проходов исполнения цикла.



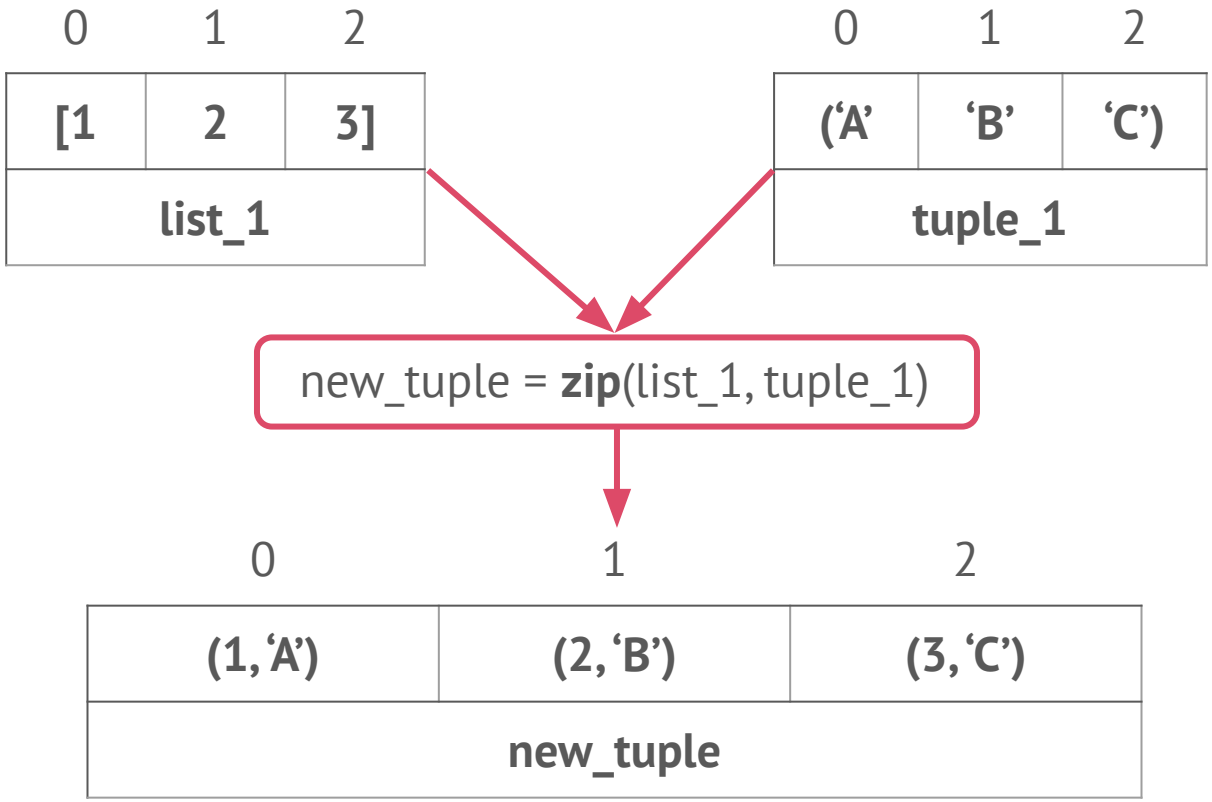
Цикл `for`

Проходится по элементам любого итерируемого объекта (строки, списка и т.д.) и во время каждого прохода выполняет заданную последовательность действий.



Функция `zip`

Функция `zip(list_1, list_2, ...)` берёт на вход несколько списков/кортежей и создаёт из них специальный zip-объект, состоящий из кортежей, такой, что первый элемент полученного объекта содержит кортеж из первых элементов всех списков-аргументов.



Домашнее задание

Задача №1

Мы делаем MVP dating-сервиса, и у нас есть список парней и девушек (их число может варьироваться):

```
boys = ['Peter', 'Alex', 'John', 'Arthur', 'Richard']
girls = ['Kate', 'Liza', 'Kira', 'Emma', 'Trisha']
```

Выдвигаем гипотезу: лучшие рекомендации мы получим, если просто отсортируем имена по алфавиту и познакомим людей с одинаковыми индексами после сортировки! “Познакомить” пары нам поможет функция zip, а в цикле распакуем zip-объект и выведем информацию в виде:

```
Идеальные пары:
Alex и Emma
Arthur и Kate
John и Kira
Peter и Liza
Richard и Trisha
```

Внимание! Если количество людей в списках будет не совпадать, то мы никого знакомить не будем и выведем пользователю предупреждение, что кто-то может остаться без пары!

Задача №2

Имеется структура данных `cook_book`, в которой хранится информация об ингредиентах блюд и их количестве в расчете на одну порцию:

```
cook_book = [
  ['салат',
      [
        ['картофель', 100, 'гр.'],
        ['морковь', 50, 'гр.'],
        ['огурцы', 50, 'гр.'],
        ['горошек', 30, 'гр.'],
        ['майонез', 70, 'мл.'],
      ]
  ],
  ['пицца',
      [
        ['сыр', 50, 'гр.'],
        ['томаты', 50, 'гр.'],
        ['тесто', 100, 'гр.'],
        ['бекон', 30, 'гр.'],
        ['колбаса', 30, 'гр.'],
        ['грибы', 20, 'гр.'],
      ],
  ],
  ['фруктовый десерт',
      [
        ['хурма', 60, 'гр.'],
        ['киви', 60, 'гр.'],
        ['творог', 60, 'гр.'],
        ['сахар', 10, 'гр.'],
        ['мед', 50, 'мл.'],
      ]
  ]
]
```

и переменная, в которой хранится количество людей, на которых необходимо приготовить данные блюда:

```
person = 5
```

Необходимо вывести пользователю список покупок необходимого количества ингредиентов для приготовления блюд на определенное число персон в следующем виде:

```
Салат:
картофель, 500гр.
морковь, 250гр.
огурцы, 250гр.
горошек, 150гр.
майонез, 350мл.

Пицца:
сыр, 250гр.
томаты, 250гр.
тесто, 500гр.
бекон, 150гр.
колбаса, 150гр.
грибы, 100гр.

Фруктовый десерт:
хурма, 300гр.
киви, 300гр.
творог, 300гр.
сахар, 50гр.
мед, 250мл.
```

Внимание! Реализация не должна зависеть от количества блюд, их названий и количества ингредиентов в них!

Задание №3

К следующей лекции прочитать про типы данных

Инструкция по выполнению домашнего задания:

1. Зарегистрируйтесь на сайте Repl.IT.
2. Перейдите в раздел **my repls**.
3. Нажмите кнопку **Start coding now!**, если приступаете впервые, или **New Repl**, если у вас уже есть работы.
4. В списке языков выберите Python.
5. Код пишите в левой части окна.
6. Посмотреть результат выполнения файла можно, нажав на кнопку **Run**. Результат появится в правой части окна.
7. После окончания работы нажмите кнопку **Share** и скопируйте ссылку из поля *Share link*.
8. В личном кабинете на сайте netology.ru в поле комментария к домашней работе вставьте скопированную ссылку и отправьте работу на проверку.

Помимо ссылки, пожалуйста, прикрепите к вашему решению любой файл (например, скриншот любой части вашего кода).

нетология

Задавайте вопросы и пишите отзыв о лекции!

Олег Гежин