

# PYTHON | БЕЗ ВОДЫ

Практический курс по Python

# **#Блок 4 | Списки**

Для быстрого старта

# О курсе

- ▶ Это ПРАКТИЧЕСКИЙ КУРС по программированию
- максимальная польза от курса в самостоятельном решении наибольшего числа задач
- вопросы на занятиях – самое ценное для учеников
- разбор задач учеников НА ЗАНЯТИЯХ
- разбор вопросов по задачам НА ЗАНЯТИЯХ

# Регламент

- ▶ Длительность одного занятия 90 минут (может +)
- ▶ 1 занятие - теория | 3 занятия - практика
- ▶ Регулярность занятий 1 раз в неделю
- ▶ Проходим блоками
- ▶ Занятия по скайпу
- ▶ Запись занятий
- ▶ Видео доступно на youtube в закрытом плейлисте

# #1 | Откуда потребность

Для быстрого старта

# Откуда потребность

- ▶ Нужно много переменных
  - в частности переменных разного типа

```
1   a1 = 3
2   a2 = 4
3   a3 = 1
4   a4 = 0
5   a5 = 9
6   a6 = 10
7   a7 = 7
8   a8 = 5
9   a9 = 9
10  a10 = 3
11  a11 = 34
12  a12 = 34
```



# Откуда потребность

- ▶ Нужно собрать данные
- ▶ Нужно хранить данные
- ▶ Нужно отфильтровать данные
- ▶ Нужно отсортировать данные
- ▶ Нужно обрабатывать данные
- ▶ Нужно преобразовать данные



# **#2 | Что такое списки**

Для быстрого старта

# Что такое списки?

## ▶ Списки:

- упорядоченный изменяемый набор объектов произвольных типов, пронумерованных от 0.

## ▶ Используются для:

- сбора и хранения данных
- фильтрации и сортировки данных
- преобразования и обработки данные



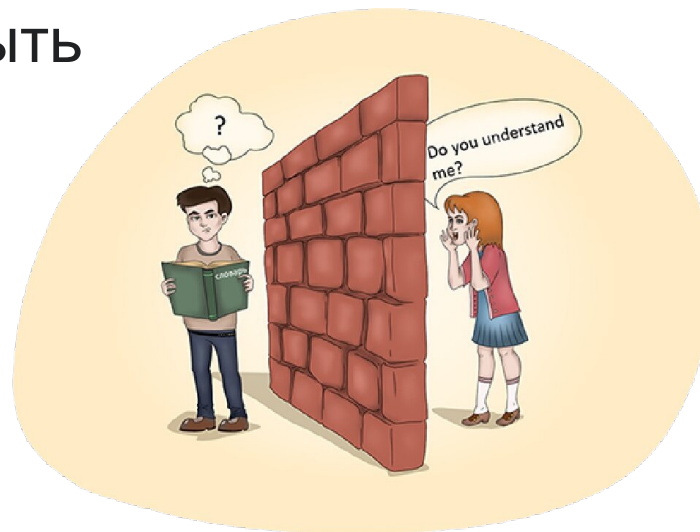
# На пальцах



# Ещё раз про объекты

## ► Объект в программировании:

- сущность в цифровом пространстве, обладающая состоянием и поведением, имеющая поля и методы
- с каждым объектом можно взаимодействовать строго определенным образом через методы объектов
- для одних объектов методы одни – для других другие
- для разных объектов могут быть одинаковые методы или действия, но срабатывать будут по-разному



# Итак списки

## ▶ Список - переменная, в которой:

- находятся разные значения (или ни одного значения)
- у каждого значения есть порядковый номер
- порядковый номер начинается строго с 0 и увеличивается на 1
- переменная заполняется строго с начала по возрастанию порядкового номера

```
a = [1, 6, 'питон', True, [1,2,3], (7, 9, 'a'), {1,8,20,'a'}]  
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py  
[1, 6, 'питон', True, [1, 2, 3], (7, 9, 'a'), {8, 1, 20, 'a'}]
```

# Списки - частность

- ▶ Списки
- ▶ Кортежи
- ▶ SET - множества
- ▶ Словари
- ▶ Другие итерируемые объекты



# Вспомним генераторы

```
a = [i for i in range(20)]  
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py  
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
```

```
a = [x for x in 'Азбука Морзе']  
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py  
['A', 'з', 'б', 'у', 'к', 'а', ' ', 'М', 'о', 'р', 'з', 'е']
```

```
a = [x for x in [1,2,3,4,5,6,7,8]]  
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py  
[1, 2, 3, 4, 5, 6, 7, 8]
```



# Доступ к элементам списка

## ▶ Каждое значение в списке = элемент

- у каждого элемента есть свой порядковый номер (начиная с 0)
- каждый элемент списка = переменная
- изменение элемента списка = изменение переменной

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
print(a)
a[0] = 111
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/cource/bin/python /Users/raybin/PycharmProjects/cource/block_4.py
[1, 4, 9, 10, 11, 22, 23, 3]
[111, 4, 9, 10, 11, 22, 23, 3]
```

# Ликбез

## ▶ Функция `len()`

- считает количество элементов итерируемой последовательности
- работает со списками, строками, кортежами, `set` множествами, словарями
- работает не со всеми итерируемыми последовательностями
- может вернуть 0

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
print(len(a))
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
8
```

# Преобразование списка генератором

## ▶ Генераторы идеально работают со списками

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
print(a)
a = [x**2 for x in a]
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
[1, 4, 9, 10, 11, 22, 23, 3]
[1, 16, 81, 100, 121, 484, 529, 9]
```

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
print(a)
a = [a[i]**2 for i in range(len(a))]
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
[1, 4, 9, 10, 11, 22, 23, 3]
[1, 16, 81, 100, 121, 484, 529, 9]
```

# **#3 | Циклы и списки**

Для быстрого старта

# Использование циклов для работы со списками

## ► Основной цикл для работы со списками - **FOR**:

- Проход по индексам

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
s = 0
for i in range(len(a)):
    s = s + a[i]
print(s)
```

```
block_4 ×
/Users/raybin/PycharmProjects/course/bin/python
83
```

- Проход по элементам

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
s = 0
for x in a:
    s = s + x
print(s)
```

```
block_4 ×
/Users/raybin/PycharmProjects/course/bin/python
83
```

- Проход по индексам и по элементам

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
s = 0
for i,x in enumerate(a):
    print(i,end=",")
    s = s + x
print(f'\n{s}')

```

```
block_4 ×
/Users/raybin/PycharmProjects/course/bin/python
0,1,2,3,4,5,6,7,
83
```

# Тоже самое функцией

## ► Функция `sum()`

- суммирует элементы списков, кортежей, set множеств

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
print(sum(a))
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python
```

```
83
```

# Заполнение списка данными

## ▶ Циклом – метод `append()`

```
a = []
for i in range(3):
    x = int(input('Введите элемент списка: '))
    a.append(x)
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
Введите элемент списка: 4
Введите элемент списка: 7
Введите элемент списка: 23
[4, 7, 23]
```

## ▶ Генератором

```
a = [int(input('Введите элемент списка: ')) for i in range(3)]
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
Введите элемент списка: 4
Введите элемент списка: 5
Введите элемент списка: 6
[4, 5, 6]
```

# Заполнение списка данными

## ▶ С помощью метода split()

```
a = list(map(int,input('Введите числа через пробел: ').split()))  
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/cource/bin/python /Users/raybin/PycharmProjects/cource/block_4.py
```

```
Введите числа через пробел: 5 43 3 2 7 5 4 3
```

```
[5, 43, 3, 2, 7, 5, 4, 3]
```

# Поиск индекса максимального элемента списка

## ▶ Циклом

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
imax, amax = 0, 0
for i in range(len(a)):
    if a[i] > amax:
        amax = a[i]
        imax = i
print(imax)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
6
```

## ▶ Генератором

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
imax = [i for i in range(len(a)) if a[i] == max(a)][0]
print(imax)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
6
```

# Сложение списков

## ▶ Через оператор суммирования

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
b = [4, 7, 9, 23, 9]
print(a+b)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
[1, 4, 9, 10, 11, 22, 23, 3, 4, 7, 9, 23, 9]
```

## ▶ Добавление одного элемента

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
a = a + [11111111]
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
[1, 4, 9, 10, 11, 22, 23, 3, 11111111]
```

# Добавление элемента в произвольное место списка

- ▶ Метод `insert(индекс_куда_вставляем, что_вставляем)`

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
a.insert(2, 2222222)
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
```

```
[1, 4, 2222222, 9, 10, 11, 22, 23, 3]
```

```
Process finished with exit code 0
```

# Добавление элемента в произвольное место списка

## ► Метод `pop(индекс_удаляемого_элемента)`

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
a.pop()
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
[1, 4, 9, 10, 11, 22, 23]
```

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
a.pop(2)
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
[1, 4, 10, 11, 22, 23, 3]
```

- Метод `pop` возвращает элемент, который был извлечен из списка
- Вернет ошибку, если список пустой или удаляемого индекса нет в списке

# Удаление элемента из списка

- ▶ Метод `pop(индекс_удаляемого_элемента)`

```
a = [1,5,4,3,7,6,9,8]
a.pop()
print(a)
print(a.pop())
```

```
block_1 x
/Users/raybin/PycharmProjects/course/bin/python
[1, 5, 4, 3, 7, 6, 9]
9
```

# Добавление элемента в произвольное место списка

## ► Метод `remove`(элемент для удаления)

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
a.remove(22)
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
[1, 4, 9, 10, 11, 23, 3]
```

- Вернет ошибку, если список в списке нет удаляемого элемента

# Сортировка списков

## ▶ Метод `sort`(список для сортировки, `reverse = False`)

```
a = [1, 4, 9, 10, 11, 22, 23, 3]
a.sort(reverse=True)
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
[23, 22, 11, 10, 9, 4, 3, 1]
```

- По умолчанию сортирует по возрастанию
- Возвращает `None` object если попытаться напечатать

# Сортировка списков

## ► Функция `sorted`(список для сортировки, `reverse=False`)

```
a = (1, 4, 9, 10, 11, 22, 23, 3)
a = sorted(a)
print(type(a))
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
```

```
<class 'list'>
```

```
[1, 3, 4, 9, 10, 11, 22, 23]
```

- По умолчанию сортирует по возрастанию
- Возвращает список
- Преобразует итерируемую последовательность в список (!)

# Сортировка списков

## ► Функция `sorted`(список для сортировки, `reverse=False`)

```
a = [(1, 4, 9), (10, 11, 22), (23, 3, 44)]  
a = sorted(a, key=lambda x:x[1], reverse=True)  
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py  
[(10, 11, 22), (1, 4, 9), (23, 3, 44)]
```

- Безымянная функция `lambda`
- Вместо `lambda` можно реализовать свою функцию
- Возможно реализовать различные критерии сортировки

# Поиск в списке

- ▶ Оператор `in` проверяет есть ли элемент в списке

```
a = [1, 4, 9, 10, 11, 22, 23, 3, 44]
if 11 in a:
    print('На борту')
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
На борту
```

- ▶ Метод `index(элемент_для_поиска)` определяет позицию первого вхождения элемента в список

```
a = [1, 4, 9, 10, 11, 22, 23, 3, 44]
print(a.index(11))
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
4
```

# Приведение к спискам

## ▶ Превращение строки в список

```
a = list('Это Python')
print(a)
```

block\_1 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_1.py
['Э', 'т', 'о', ' ', 'P', 'y', 't', 'h', 'o', 'n']
```

## ▶ Итерируемых последовательностей в список

```
a = list(range(20))
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python /Users/raybin/PycharmProjects/course/block_4.py
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
```

# Срезы списков

## ▶ Срезать с элемента до конца списка

```
a = [1, 4, 9, 10, 11, 22, 23, 3, 44]
a = a[2:]
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python
[9, 10, 11, 22, 23, 3, 44]
```

# Срезы списков

## ▶ Срезать с начала до определенного индекса

```
a = [1, 4, 9, 10, 11, 22, 23, 3, 44]
a = a[:-2]
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python
[1, 4, 9, 10, 11, 22, 23]
```

# Срезы списков

## ► Срез от левого индекса до правого

```
a = [1, 4, 9, 10, 11, 22, 23, 3, 44]
a = a[2:-2]
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python
[9, 10, 11, 22, 23]
```

# Срезы списков

## ► Список наоборот

```
a = [1, 4, 9, 10, 11, 22, 23, 3, 44]
a = a[::-1]
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python
[44, 3, 23, 22, 11, 10, 9, 4, 1]
```

# Срезы списков

## ▶ Срез среза

```
a = [1, 4, 9, 10, 11, 22, 23, 3, 44]
a = a[1:7][:3]
print(a)
```

block\_4 ×

```
/Users/raybin/PycharmProjects/course/bin/python
[4, 9, 10]
```



# Про использование списков

## ▶ Списки = арсенал для решения огромного количества задач

- задачи на итерируемые объекты
- задачи на обработку данных
- задачи на обработку строк
- неожиданные решения разных задач

## ▶ Списки + генераторы арсенал для обработки данных

# Задача блока 4

## ▶ НАУЧИТЬСЯ ГРАМОТНО РАБОТАТЬ СО СПИСКАМИ

```
a = a + [1] + [2]
print(a)
a = a - [10]
print(a)
a = a/10
print(a)
a = a//10
print(a)
a = a.minus(10)
print(a)
```

