# 2 занятие

Школа::Кода

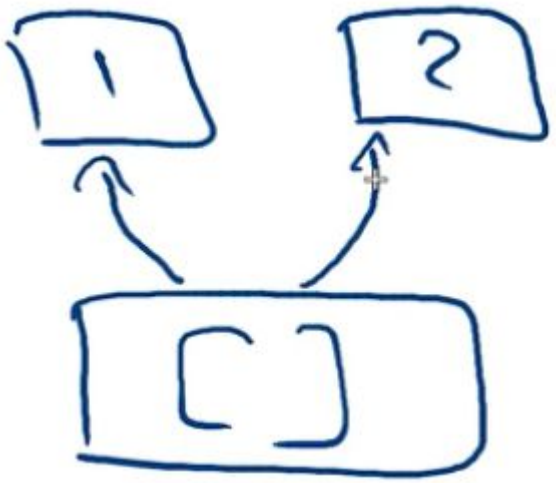
«Основы программирования на языке Python»

# Проверка присутствия

# Представление данных в языке

**Всё в Python - есть объекты**

Объект - это абстракция для данных. Объект - это некоторый контейнер в памяти, который содержит данные.




Все данные в языке Python представлены объектами и отношениями между объектами:

```
a = 1
b = 2
c = [a, b]
```

В языка Python у любого объекта есть три обязательные вещи: идентификатор, тип и значение. Тип и идентификатор объекта не изменяются в течение жизни объекта.

Никакие два объекта в один момент времени не обладают одинаковым идентификатором.

```
>>> x = [1, 2, 3]
>>> print(id(x))
51747776
>>> print(id([1, 2, 3]))
51756528
```

Идентификатор объекта - это какое-то число. Переменная в языке Python - это всего лишь ссылка на объект. Значение переменной это всегда значение объекта.

```
>>> x = [1, 2, 3]
>>> y = x
 >>> print(y is x)
True
>>> x.append(4)
>>> print(x)
[1, 2, 3, 4]
>>> print(y)
[1, 2, 3, 4]
```

**append** изменяет (изменяемый) объект, конкатенация создает новый. В следующем коде идет переопределение переменной **t**, а не объекта, на который ссылается **s**.

```
>>> x = [1, 2, 3]
>>> y = x
>>> y.append(4)
>>> s = "123"
>>> t = s
>>> t = t + "4"
 >>> print(str(x) + " " + s)
[1, 2, 3, 4] 123
```

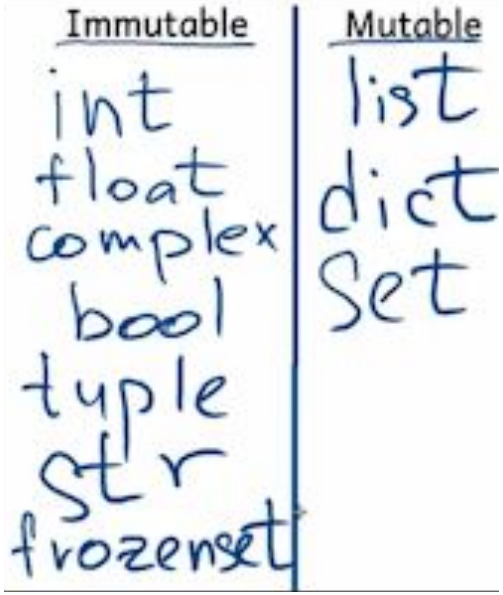
У любого объекта есть тип, который определяет, что можно сделать с объектом. Тип определяет поведение объекта и возможные принимаемые значения для объекта. Тип объекта не меняется в течение жизни объекта. Узнать тип объекта можно с помощью функции type():

```
>>> x = [1, 2, 3]
 >>> type(x)
<class 'list'>
>>> type(4)
 <class 'int'>
>>> type(type(x))
 <class 'type'>
```

list - это стандартный тип отвечающий за список в языке Python. Типы в языке Python также являются объектами. Тип типа **x** - это класс type.

**Изменяемые и неизменяемые типы**

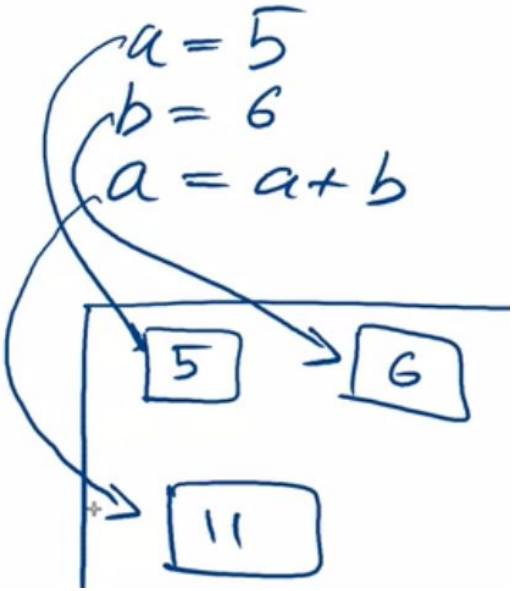
Всегда стоит помнить какие типы данных являются изменяемыми и всегда держать в голове, если у вас несколько переменных ссылаются на один и тот же объект изменяемого типа. Это значит, что изменение какой-либо переменной, которая ссылается на этот объект повлечет изменений объекта, а как следствие и всех переменных, которые ссылаются на этот объект.



**Неизменяемые объекты**

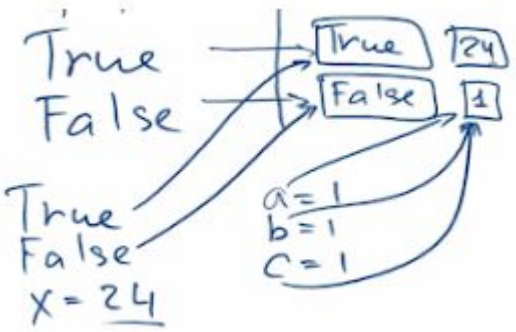
Immutable object - объект, который НЕ может менять свое значение.

Числа (int, float, complex) являются неизменяемыми объектами:

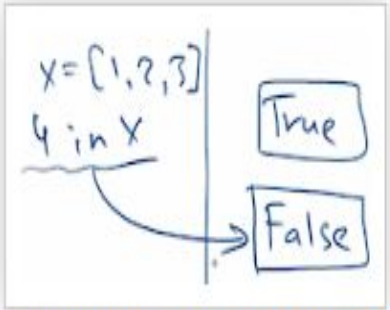



Для того чтобы получить сумму мы создали новый объект.

Интерпретатор не всегда создает новый объект каждый раз, когда встречает новое число. Если объекта в памяти еще не было, тогда он будет создан, а если он уже был, то Python может переиспользовать старый:



Логический тип bool также относится к неизменяемым типам. Для значений логического типа в памяти хранится всего два объекта:




В любой момент времени любой объект логического типа ссылается всегда на один из этих двух объектов. Такая оптимизация позволяет не хранить в памяти огромное число логических объектов.

Также к неизменяемым объектам относятся кортеж (tuple) и строки как последовательности символов (str). Стандартной кодировкой строк в языке Python является кодировка UTF-8.
У множества есть неизменяемая версия - frozenset.

## Что такое множество?

Множество в python - "контейнер", содержащий не повторяющиеся элементы в случайном порядке.

Единственное отличие set от frozenset заключается в том, что set - изменяемый тип данных, а frozenset - нет.

Примерно похожая ситуация с списками и кортежами.

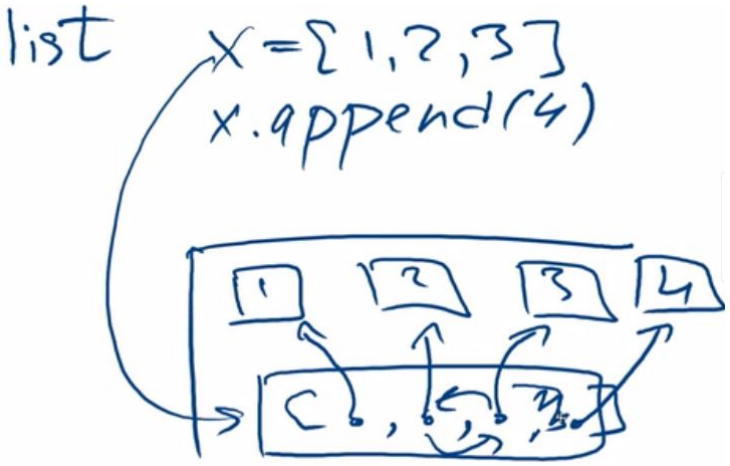
```
>>> a = set('qwerty')
>>> b = frozenset('qwerty')
>>> a == b
True
>>> True
True
>>> type(a - b)
<class 'set'>
>>> type(a | b)
<class 'set'>
>>> a.add(1)
>>> b.add(1)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'frozenset' object has no attribute 'add'
```

**Изменяемые объекты**

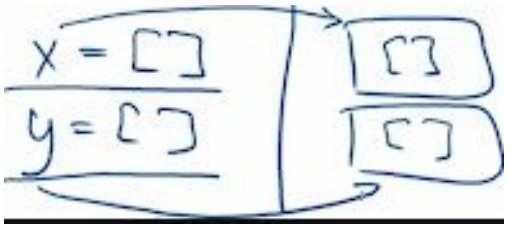
Mutable object - объект, который может изменять свое значение в течение своей жизни.

Изменяемых стандартных типов в языке Python всего 3: список, словарь и множество.

Список является изменяемым типом



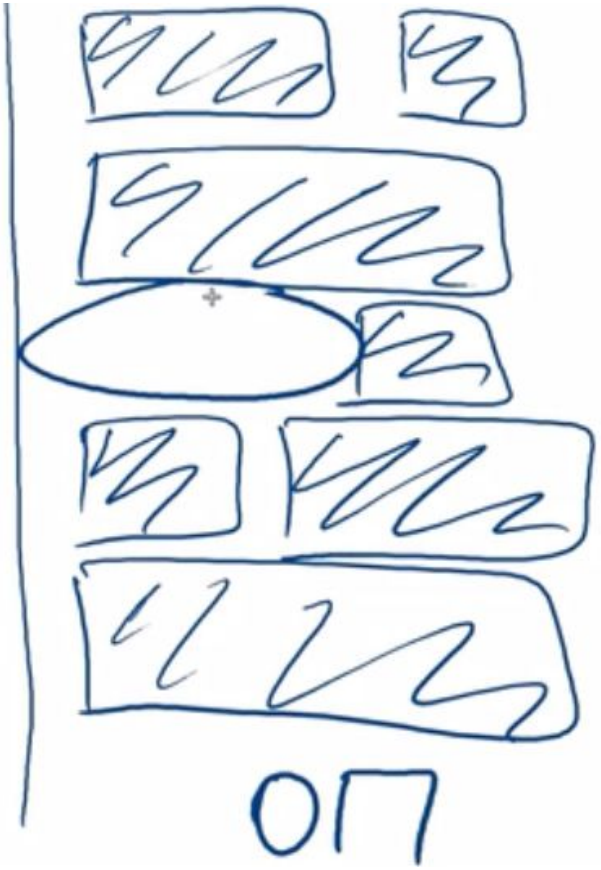

Для списков гарантируется, что когда интерпретатор встретит в коде список он создаст для него новый объект:



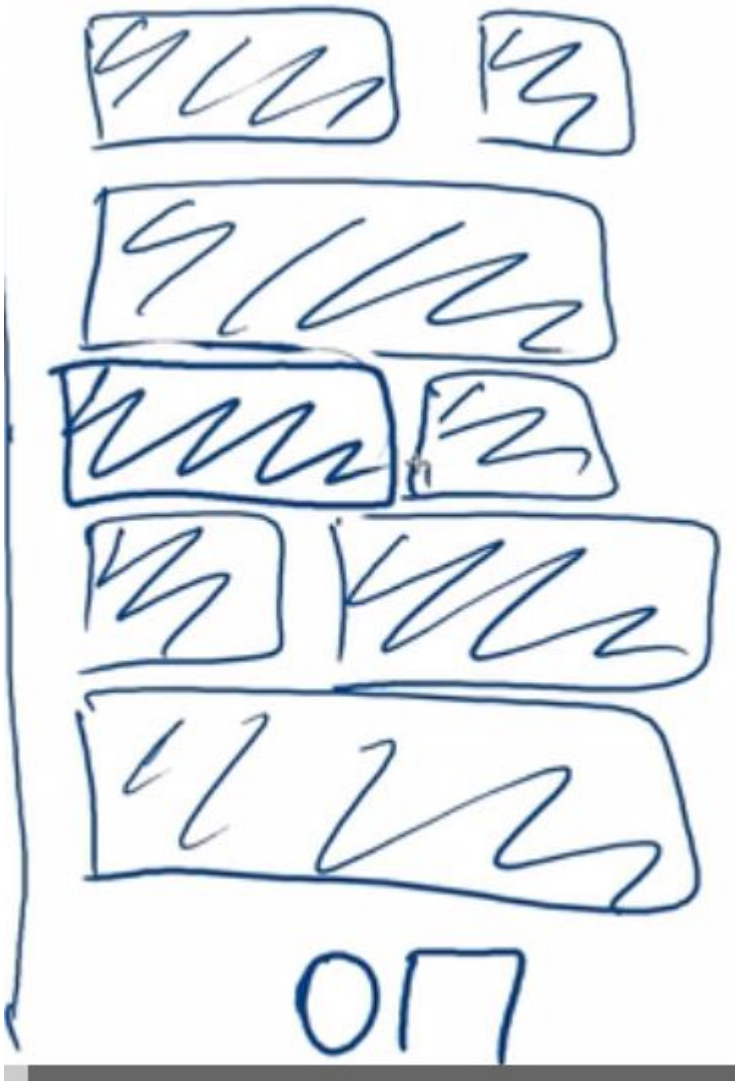

Это верно для всех изменяемых типов языка Python. Словари (dict) являются изменяемым типом данных. Множество (set) также является изменяемым типом данных

**Основные этапы в жизни любого объекта**

Первым этапом является выделение памяти. Первым дело ищется свободная зона в памяти, где можно создать объект:



Вторым этапом в жизни объекта является его создание:



Объекты содержат в себе счетчик ссылок. Объекты знают сколько раз на них ссылаются.

В языке Python ссылаются не только переменные. Счетчик ссылок нужен объекту для того чтобы понять, когда его можно удалить.

После выделения памяти под объект и создания объекта следуют несколько стадий:

1. Первая стадия, когда на объект что-то ссылается - объект живет полноценной жизнью.
2. Вторая стадия, когда на объект никто не ссылается, интерпретатор помечает объект на удаление. Далее объект просто существует и ждет, когда придет сборщик мусора и удалит его. В какой-то момент времени приходит сборщик мусора, осматривает всю оперативную память, находит все помеченные объекты, удаляет объекты и освобождает ту память, которую они занимали.

После выполнения этой части кода число ссылок на объект, содержащий строку "Count me!" может быть равно 1 или 3.

```
s = "Count me!"
t = s
a = [s, t]
s = "Other string"
a = [s]
```

Все зависит от того запускался ли сборщик мусора. Значение количества ссылок до и после запуска сборщика мусора могут отличаться. В данном случае он мог отработать перед выполнением последней строки кода и удалить часть ссылок, которые уже неактуальны. Поэтому число ссылок - либо/либо. Когда запустится сборщик мусора - определяется самим интерпретатором, поэтому он может отработать в любой момент времени.