**Лекция 2**

**Типы данных и переменные**

1. **Типы данных в Python**

В Python типы данных можно разделить на встроенные в интерпретатор *(built-in*) и не встроенные, которые можно использовать при импортировании соответствующих модулей.

К основным встроенным типам относятся:

1. *None* (неопределенное значение переменной)
2. Логические переменные (*Boolean Type*)
3. Числа (*Numeric Type*)
   1. *int* – целое число
   2. *float* – число с плавающей точкой
   3. *complex* – комплексное число
4. Списки (*Sequence Type*)
   1. *list* – список
   2. *tuple* – кортеж
   3. *range* – диапазон
5. Строки (*Text Sequence Type* )
   1. *str*
6. Бинарные списки (*Binary Sequence Types*)
   1. *bytes* – байты
   2. *bytearray* – массивы байт
   3. *memoryview* – специальные объекты для доступа к внутренним данным объекта через protocol buffer
7. Множества (*Set Types*)
   1. *set* – множество
   2. *frozenset* – неизменяемое множество
8. Словари (*Mapping Types*)
   1. *dict* – словарь
   2. **Изменяемые и неизменяемые типы данных**

В Python существуют изменяемые и неизменяемые типы.

К **неизменяемым** (*immutable*) типам относятся: целые числа (*int*),  числа с плавающей точкой (*float*), комплексные числа (*complex*), логические переменные (*bool*), кортежи (*tuple*), строки (*str*) и неизменяемые множества (*frozen set*).

К **изменяемым** (*mutable*) типам относятся: списки (*list*), множества (*set*), словари (*dict*).

Как уже было сказано ранее, при создании переменной, вначале создается объект, который имеет уникальный идентификатор, тип и значение, после этого переменная может ссылаться на созданный объект.

Неизменяемость типа данных означает, что созданный объект больше не изменяется. Например, если мы объявим переменную*k* = 15, то будет создан объект со значением 15, типа *int* и идентификатором, который можно узнать с помощью функции *id()*.

>>> k = **15**

>>> id(k)

**1672501744**

>>> type(k)

<*class* 'int'>

Объект с *id* = 1672501744 будет иметь значение 15 и изменить его уже нельзя.

Если тип данных изменяемый, то можно менять значение объекта. Например, создадим список [1, 2], а потом заменим второй элемент на 3.

>>> a = [**1**, **2**]

>>> id(a)

**47997336**

>>> a[**1**] = **3**

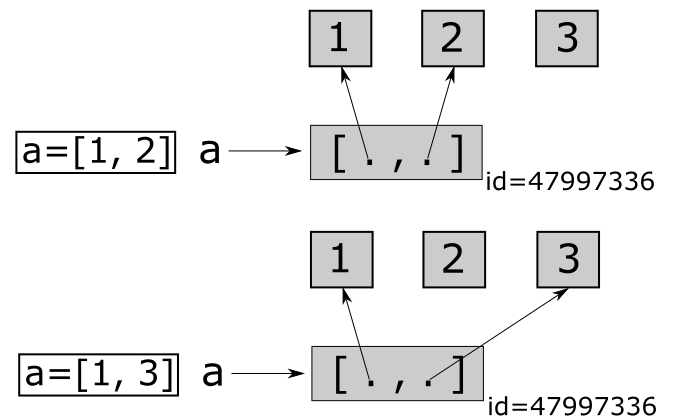
>>> a

[**1**, **3**]

>>> id(a)

**47997336**

Как видно, объект на который ссылается переменная *a*, был изменен. Это можно проиллюстрировать следующим рисунком.



В рассмотренном случае, в качестве данных списка, выступают не объекты, а отношения между объектами. Т.е. в переменной *a*хранятся ссылки на объекты содержащие числа 1 и 3, а не непосредственно сами эти числа.

* 1. **Переменные**

Переменная – это именованная область памяти, в которой во время выполнения программы хранятся данные определенного типа (о типах данных расскажем ниже). В [стандартах оформления кода PEP 8](https://peps.python.org/pep-0008/#type-variable-names) даны рекомендации по названиям переменных:

* Названия не должны начинаться с цифры, но могут заканчиваться цифрой. Например, назвать переменную 7up – неправильно, а так – seven11 – можно.
* Названия могут состоять из комбинации строчных, заглавных букв, цифр и символов подчеркивания: lower\_case, mixedCase, CapitalizedCase, UPPER\_CASE, lower123.
* Не следует давать переменным названия, совпадающие со служебными словами, названиями встроенных функций и методов, к примеру – **print**, **list**, **dict**, **set**, **pass**, **break**, **raise**.
* Следует избегать использования отдельных букв, которые могут быть ошибочно приняты друг за друга – l (L в нижнем регистре), I (i в верхнем регистре) или за нуль – O.
* В названиях не должно быть пробелов, дефисов и специальных символов, например, 'или $.
* Главный принцип именования переменных – осмысленность. По названию переменной должно быть понятно, какого рода данные в ней хранятся – например, **car\_model**, **petName,** **CARD\_NUMBER**более информативны, чем **a**, **а1**, **а2**.

Переменные выполняют две важные функции:

* делают код понятнее;
* дают возможность многократно использовать введенные данные.

Если программа небольшая, а введенное значение используется однократно, можно обойтись без использования переменной:

print(‘Привет,’, input(‘Как тебя зовут?’))