

# Лекция 6

Введение в Python. Часть 2

# Битовые операции

Операнд	Описание
	Побитовый OR
^	Побитовый XOR
&	Побитовый AND
<<, >>	Смещения
~x	Побитовый NOT

# Импорт собственных модулей

- Код модуля выполняется при импорте.
- У любого модуля python есть атрибут `__name__`.
- Если модуль запускается как главный модуль программы, атрибут `__name__` хранит значение `"__main__"`.
- Если модуль импортируется, атрибуту `__name__` присваивается имя модуля.

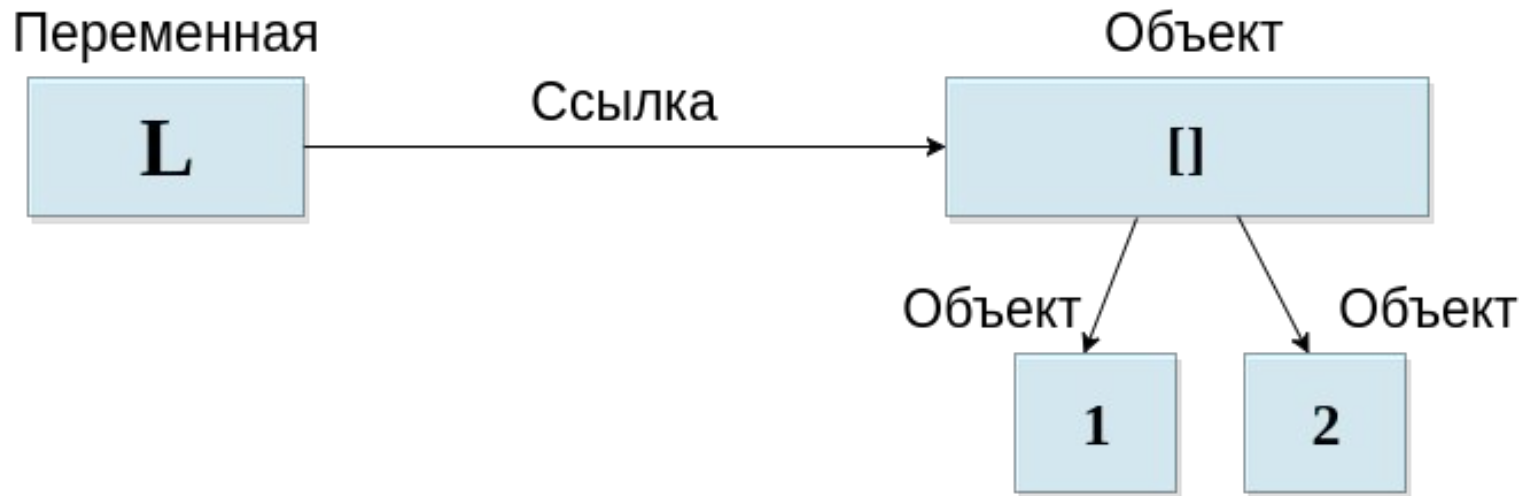
# Операция присваивания

x = 100



# Операция присваивания

$L = [1, 2]$



# Операция присваивания

`x = 100`

- Создается объект 100
- Переменная x создается, если до этой строки она не была создана
- В переменную x записывается ссылка на объект 100

# Переменные, объекты, ссылки

x = 100

- **Переменная** хранит ссылку на объект
- **Объект** - область памяти, которая хранит значение объекта. Имеет два стандартных поля: тип и счетчик ссылок на этот объект
- **Ссылка** - это автоматически разыменовываемый указатель на объект

## Разделяемые ссылки

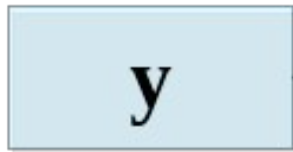
`x = 100`

`y = x`

Переменная



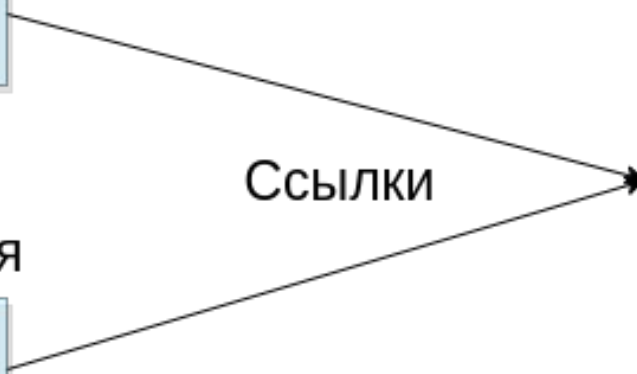
Переменная



Объект



Ссылки



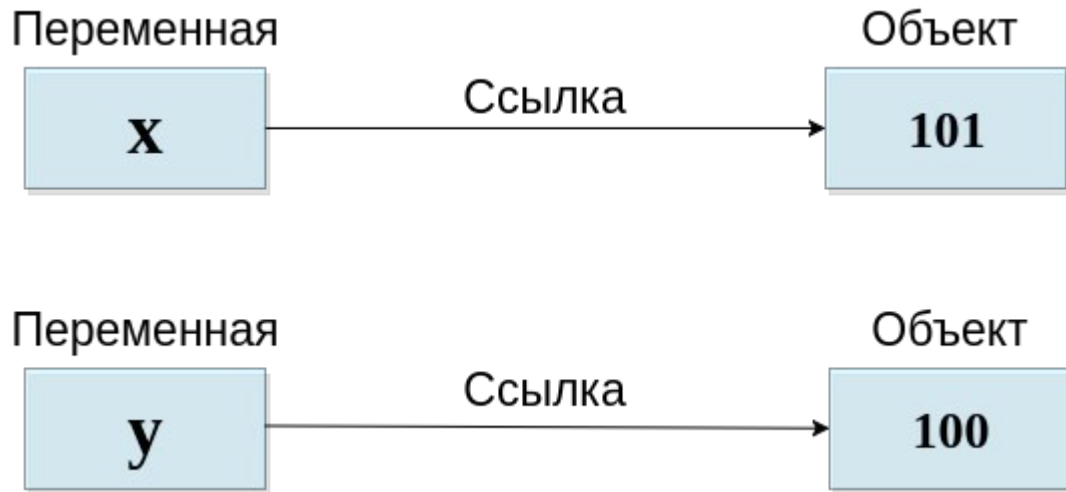


# Разделяемые ссылки неизменяемых объектов

`x = 100`

`y = x`

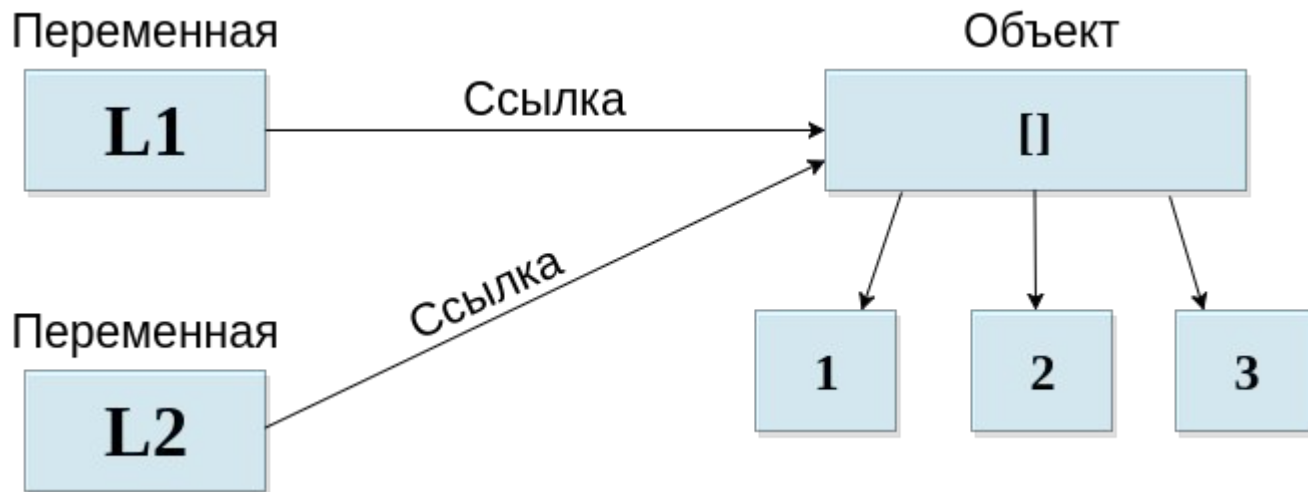
`x += 1`



# Разделяемые ссылки СПИСКОВ

$L1 = [1, 2, 3]$

$L2 = L1$

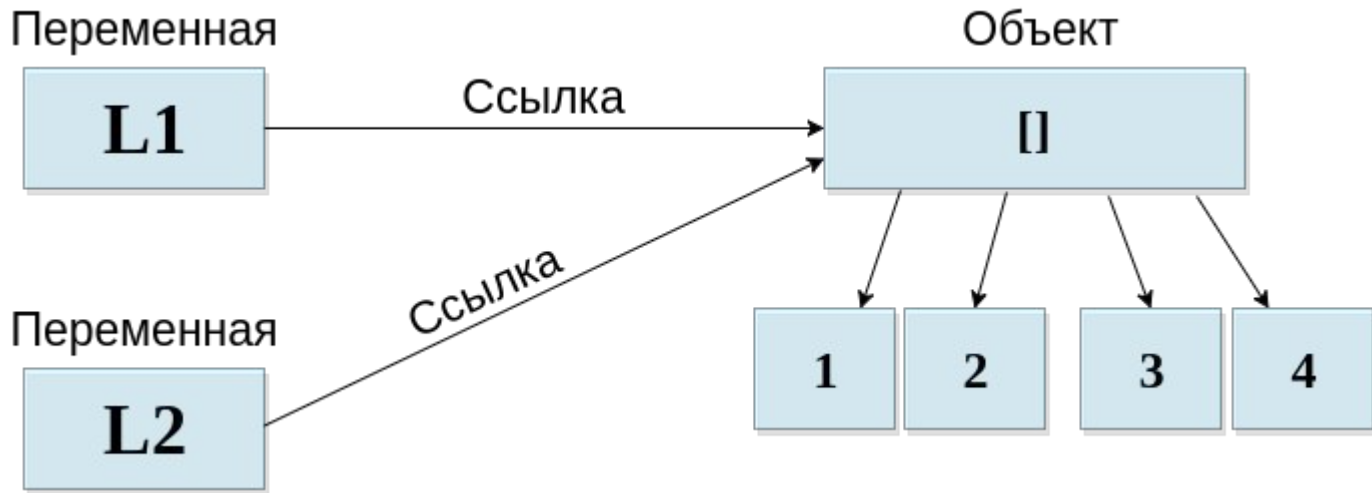


# Разделяемые ссылки СПИСКОВ

```
L1 = [1, 2, 3]
```

```
L2 = L1
```

```
L1.append(4)
```



## Проверка идентичности объектов

`a = 100900`

`b = 100900`

`a == b` # проверка на равенство значений объектов

`a is b` # проверка на идентичность объектов

`a is not b` # проверка на то, что объекты разные

# Изменяемые и неизменяемые объекты

## Неизменяемые:

- числа
- строки
- логический тип
- кортежи

## Изменяемые:

- списки
- словари
- множества

## Копирование списка

- `a = list(b)` # создание поверхностной копии списка b
- `a = b[:]` # создание поверхностной копии списка b
- `a = b.copy()` # создание поверхностной копии списка b

## Копирование списка

➤ Модуль `copy`:

- метод `copy()`:

`a = copy.copy(b)` # создание поверхностной копии списка `b`

- метод `deepcopy()`:

`a = copy.deepcopy(b)` # создание полной копии списка `b`

# Передача аргументов в функцию

- Аргументы передаются через автоматическое **присваивание** объектов локальным переменным.
  - Изменение **изменяемого** объекта внутри функции приведет к изменению объектов в вызывающей программе.
  - Изменение **неизменяемого** объекта внутри функции не приведет к изменению объектов в вызывающей программе.



# Генераторы списков

```
L = [x for x in range(10)]
```

```
L = []
```

```
for i in range(10):
```

```
    L.append(i)
```

## Матрицы на основе списка

➤ `matrix = [[0, 0, 0],  
 [0, 0, 0],  
 [0, 0, 0]]`

➤ Генератор списка:

```
matrix = [[0 for x in range(3)] for y in range(3)]
```

# Области видимости переменных

- Область видимости - это место в коде, где определяются имена и где они могут быть найдены.
- Пространство имён - место, где находятся имена.
- Любая функция содержит в себе пространство имён: имена, определяемые внутри функции, видны только в пределах этой функции.
- Функции образуют локальную область видимости, а модули — глобальную.

# Области видимости переменных

- Если присваивание происходит внутри функции, переменная является **локальной** для этой функции.

```
x = 100
def f():
    x = 777
    print(x)
f()
print(x)
```

# Области видимости переменных

- Если присваивание происходит в пределах некоторой внешней функции, переменная является **нелокальной** для внутренней функции.

```
def f2(): # внешняя функция
```

```
    def f1(): # внутренняя функция
```

```
        nonlocal x # без nonlocal мы не сможем изменить x
```

```
        x = 100
```

```
    x = 777
```

```
    f1()
```

```
f2()
```

# Области видимости переменных

- Если присваивание производится за пределами всех инструкций def, переменная является **глобальной** для всего файла.

```
def f1():  
    global x  
    x = 100
```

```
x = 777  
f1()
```

# Полезные ссылки

1. Документация Python, побитовые операции  
<https://wiki.python.org/moin/BitwiseOperators>
2. Курс на Stepik “Python: основы и применение” уроки 1.2, 1.4, 2.2  
<https://stepik.org/course/512/syllabus>
3. Learning Python, Fourth Edition, by Mark Lutz. Copyright 2009 O'Reilly Media, Inc., 978-0-596-15806-4
4. Вики от института биоинформатики, области видимости:  
[http://wiki.bioinformaticsinstitute.ru/wiki/%D0%9E%D0%B1%D0%BB%D0%B0%D1%81%D1%82%D0%B8\\_%D0%B2%D0%B8%D0%B4%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8](http://wiki.bioinformaticsinstitute.ru/wiki/%D0%9E%D0%B1%D0%BB%D0%B0%D1%81%D1%82%D0%B8_%D0%B2%D0%B8%D0%B4%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8)
5. Документация Python, генераторы списков  
<https://docs.python.org/3/howto/functional.html#generator-expressions-and-list-comprehensions>