Объекты и переменные

Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

Объекты и переменные

Исходный код

```
a = set()
b = a
```

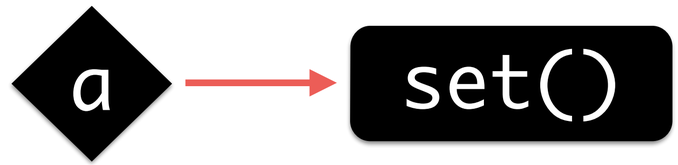
Память

Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

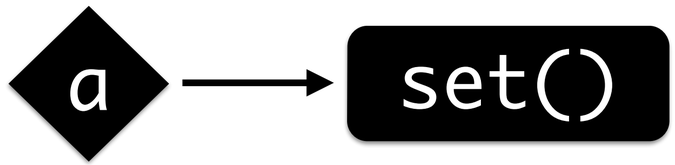


Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память



Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

Объекты и переменные

Исходный код

```
a = set()
b = a
```

Память

Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

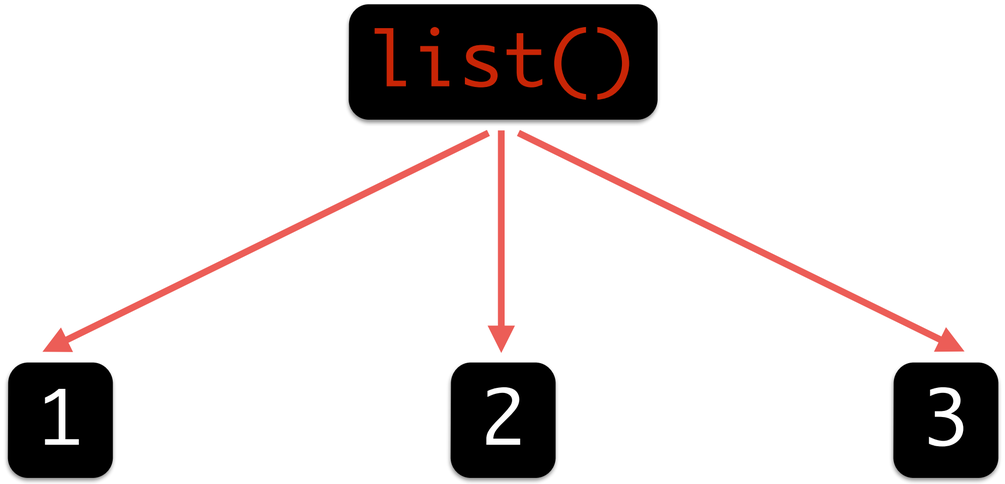
Память

Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

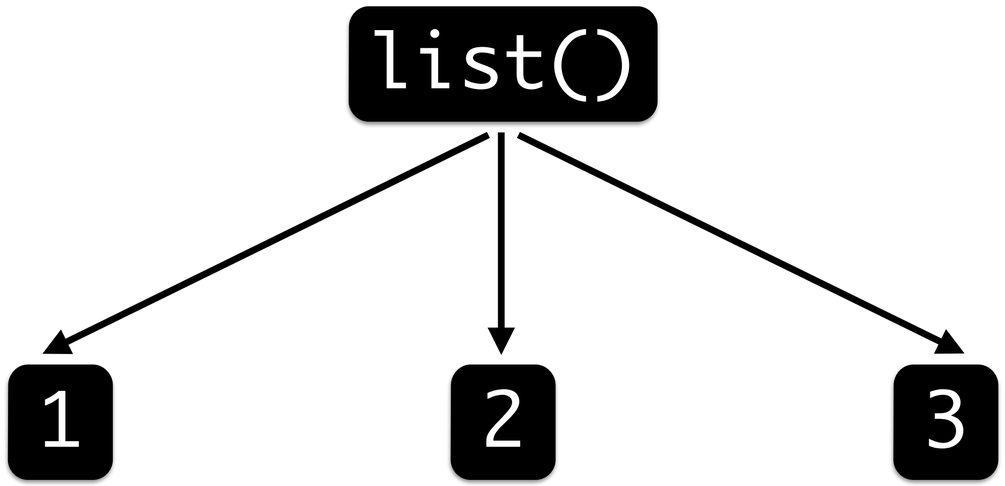


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

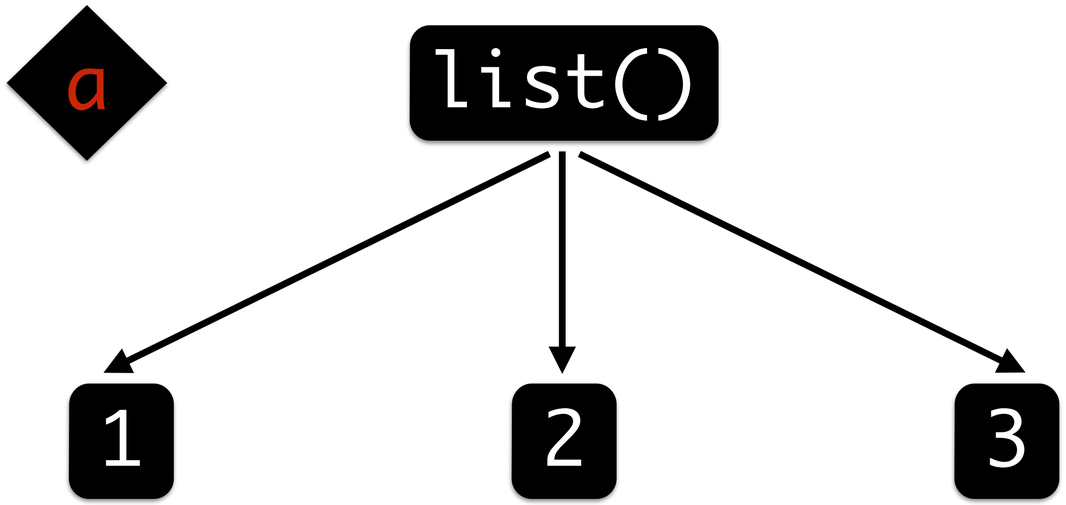


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

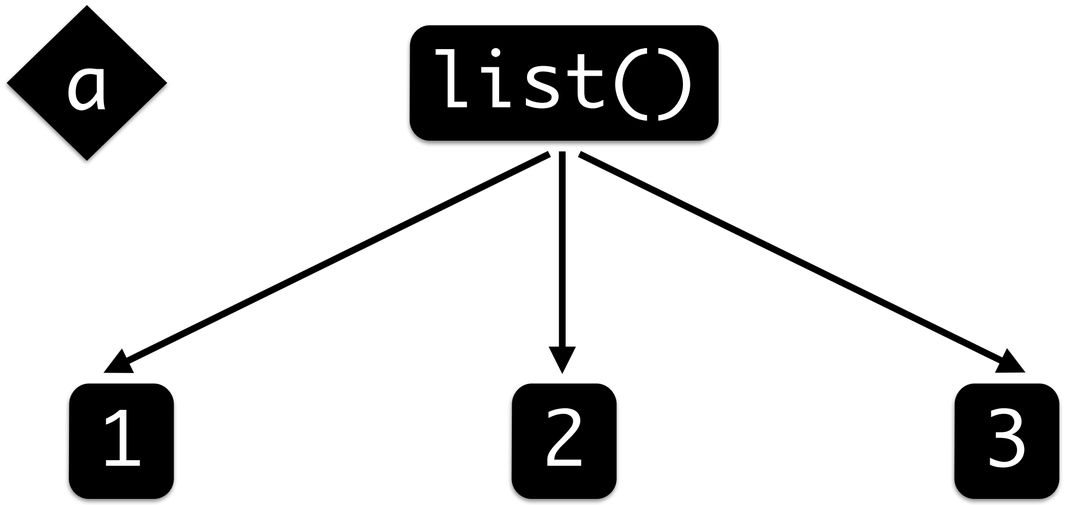


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

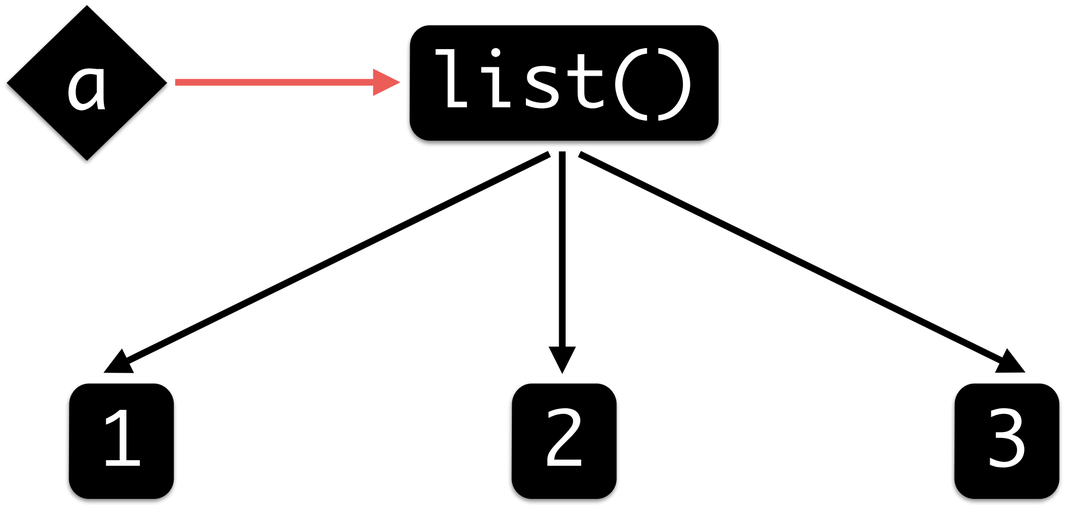


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

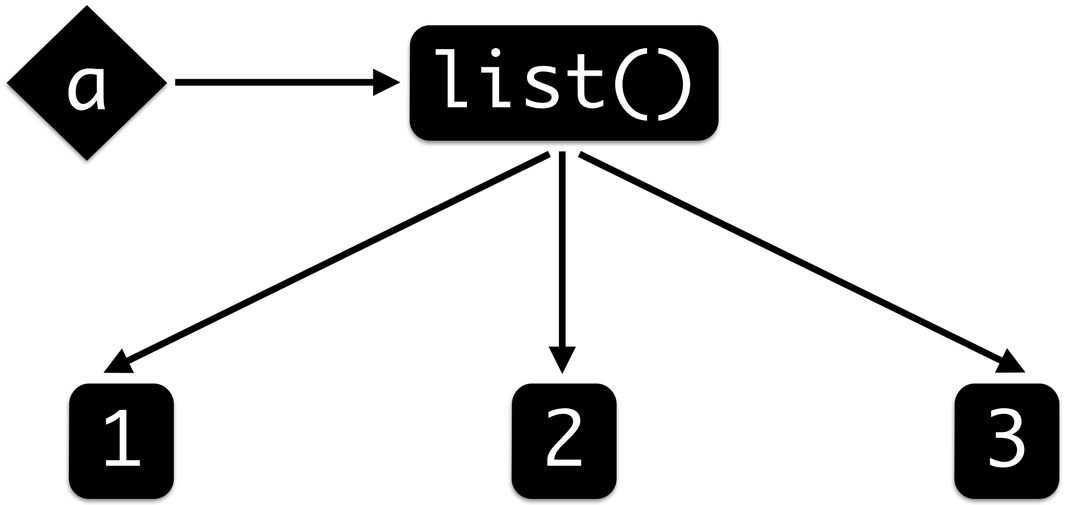


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

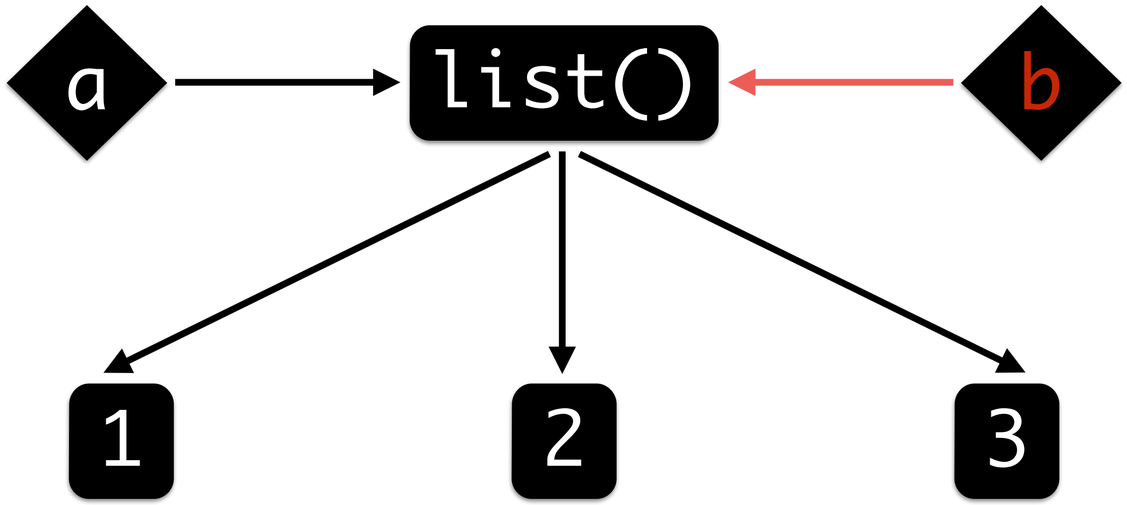


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

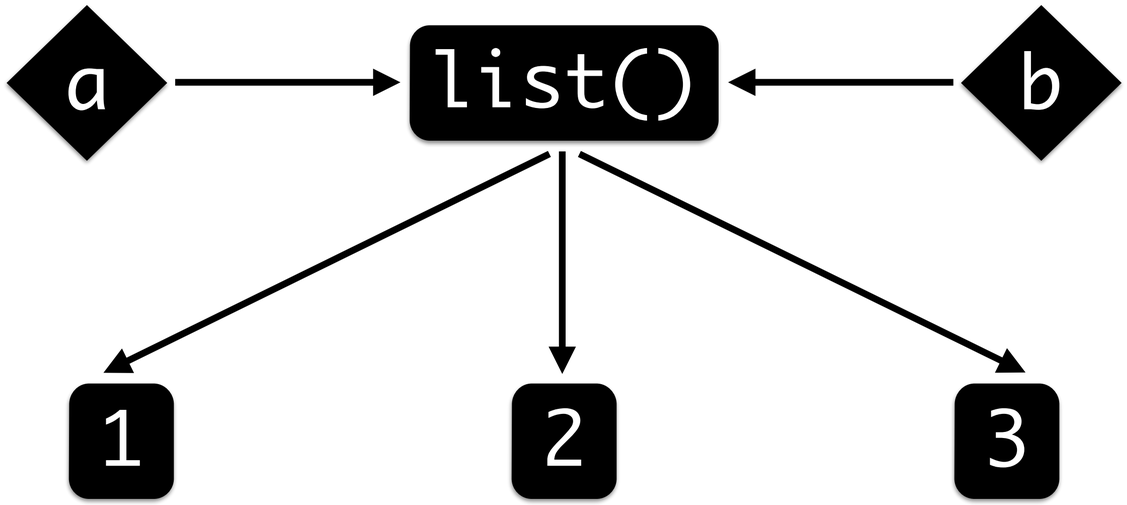


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

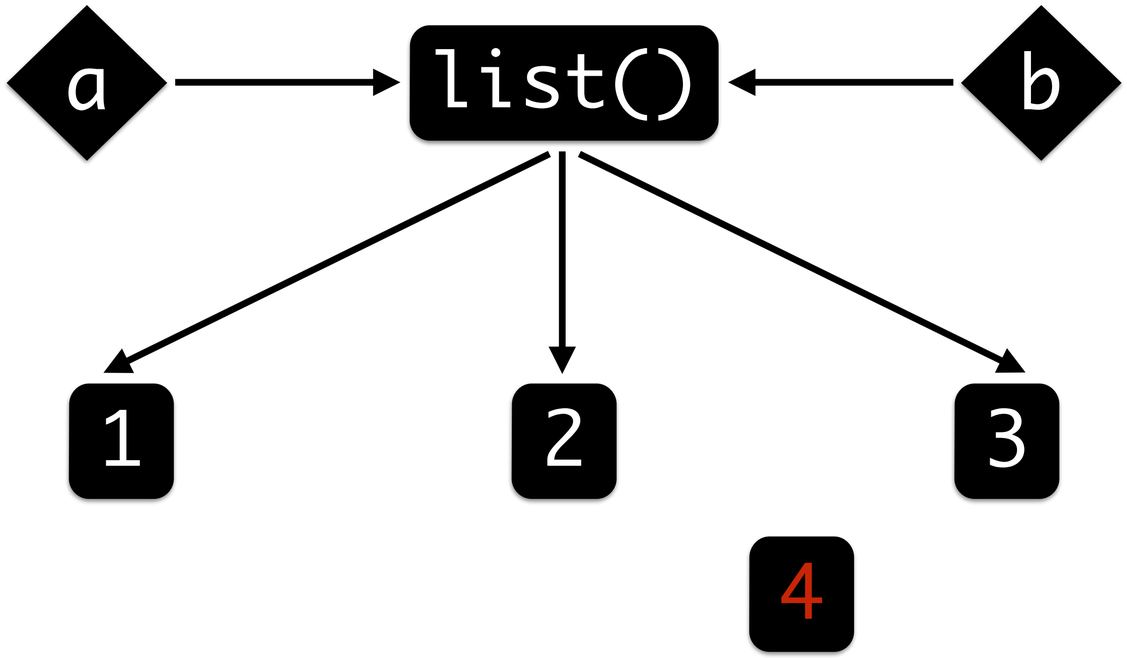


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

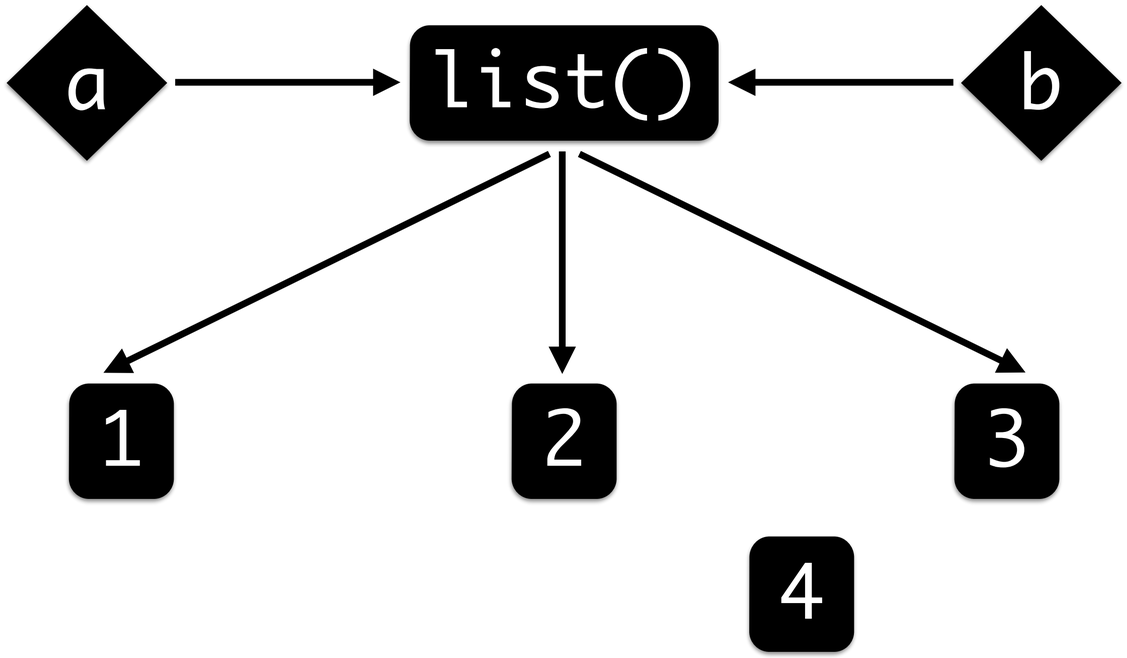


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

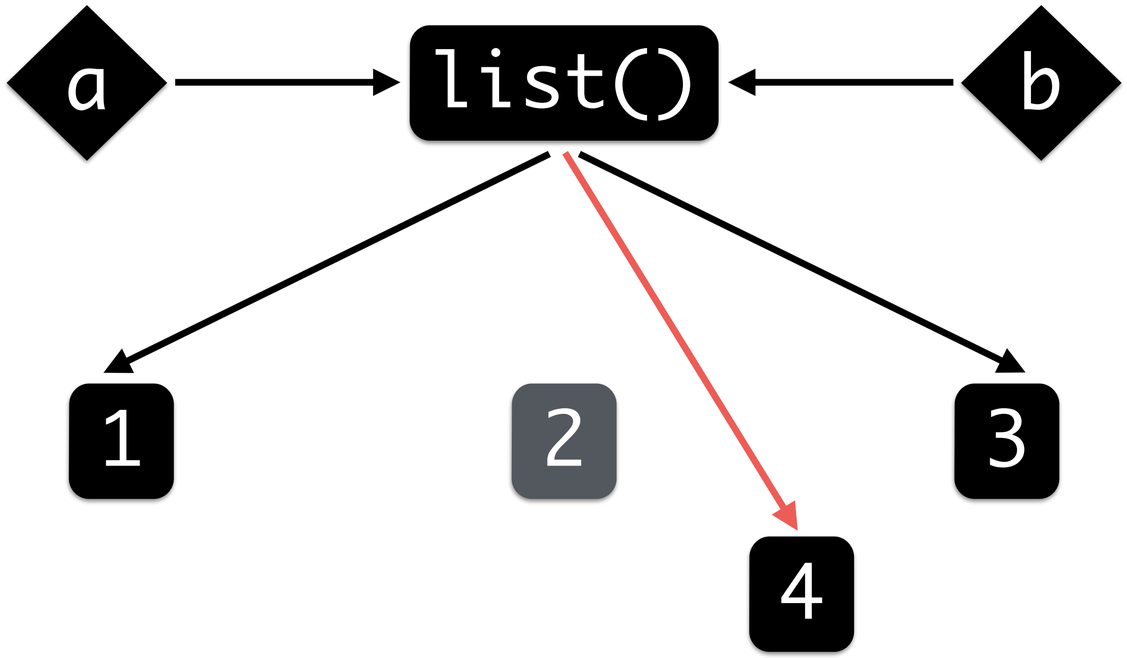


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

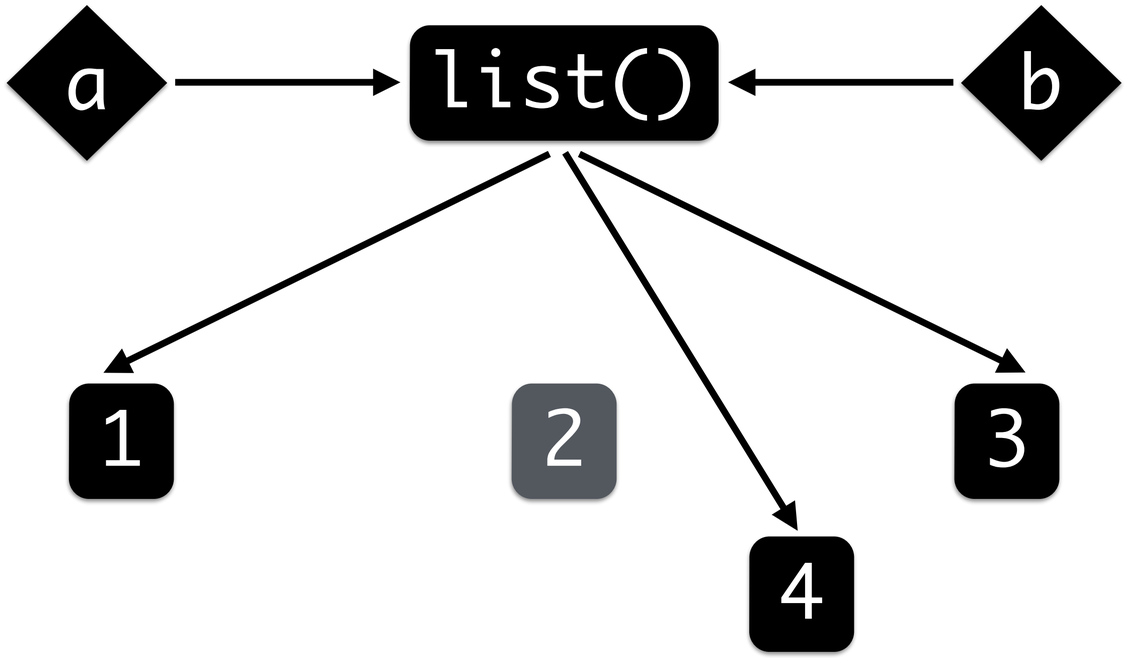


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память

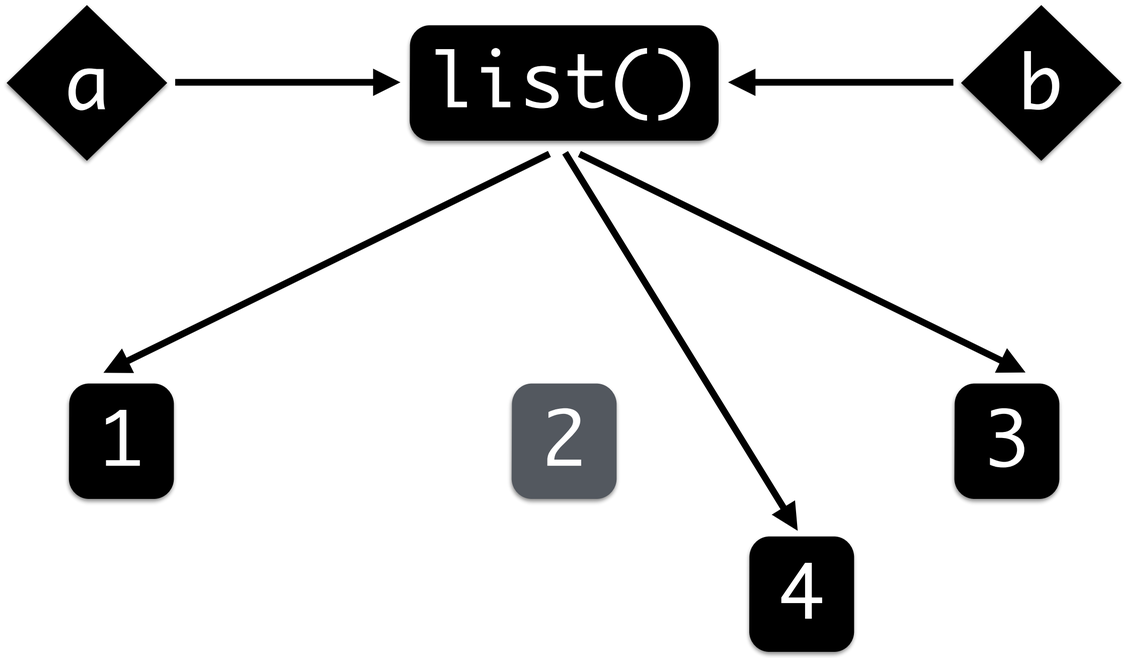


Объекты и переменные

Исходный код

```
a = [1, 2, 3]
b = a
b[1] = 4
print(a)
print(b)
```

Память



Переменные хранят ссылки на объекты

locals()

Как получить копию?

Копия при помощи конструктора

Исходный код

```
a = [1, 2, 3]
b = list(a)
b[1] = 4
print(a)
print(b)
```

Память

Копия при помощи конструктора

Исходный код

```
a = [1, 2, 3]
b = list(a)
b[1] = 4
print(a)
print(b)
```

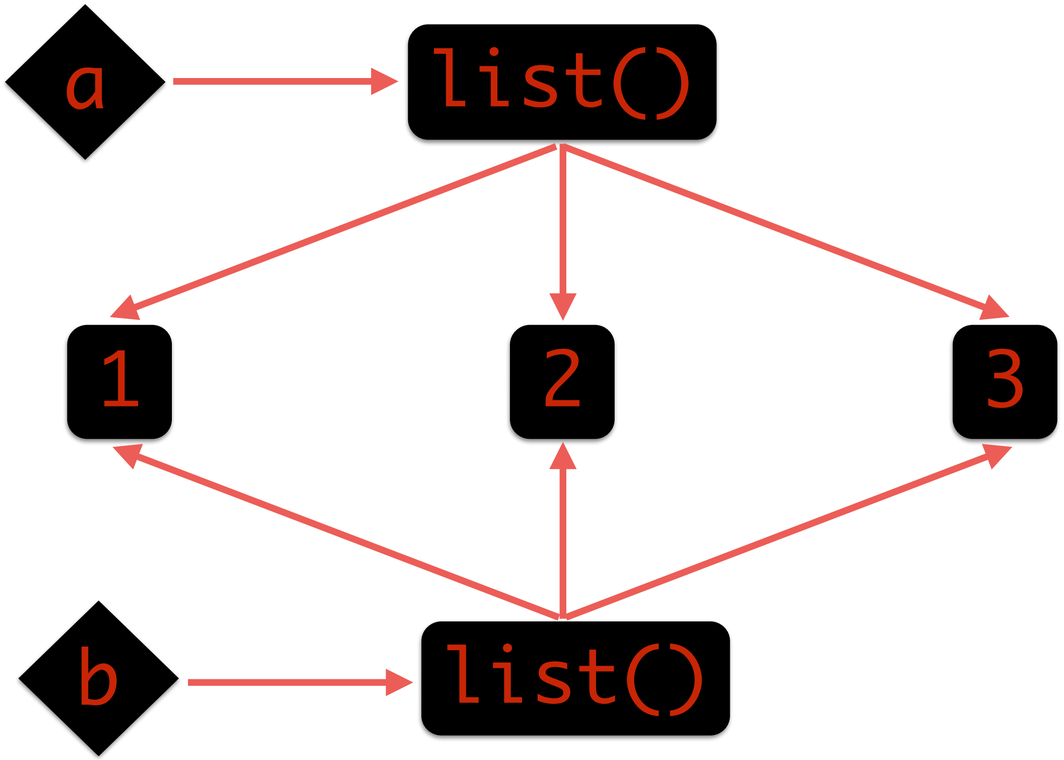
Память

Копия при помощи конструктора

Исходный код

```
a = [1, 2, 3]
b = list(a)
b[1] = 4
print(a)
print(b)
```

Память

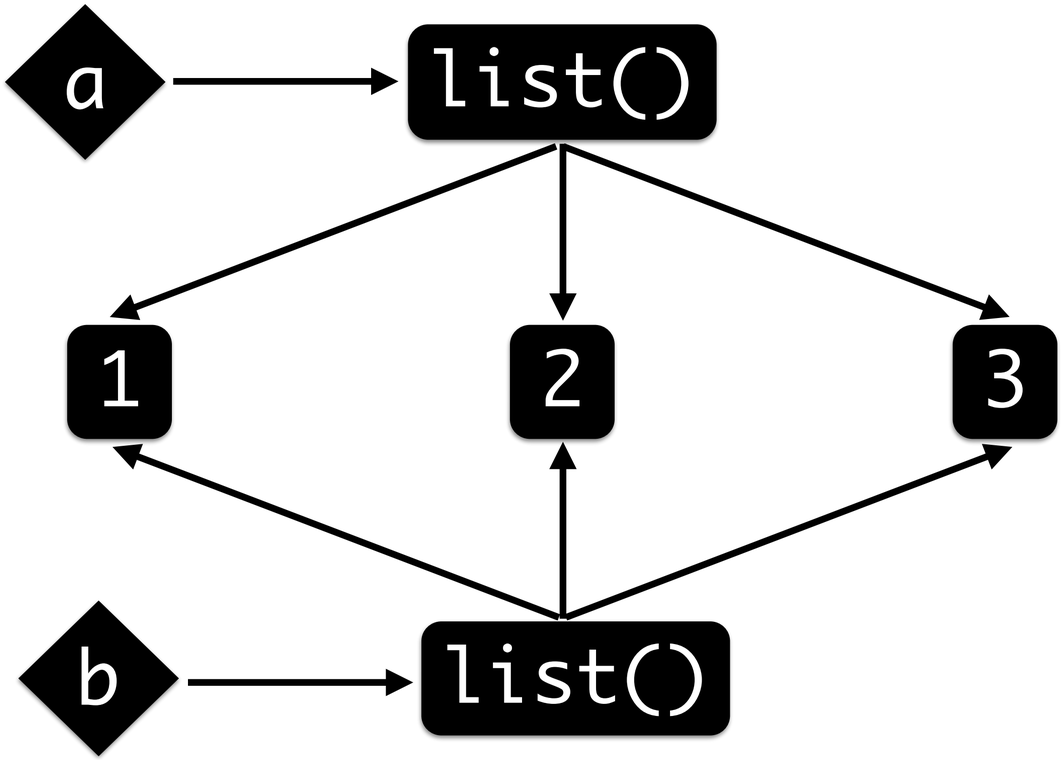


Копия при помощи конструктора

Исходный код

```
a = [1, 2, 3]
b = list(a)
b[1] = 4
print(a)
print(b)
```

Память

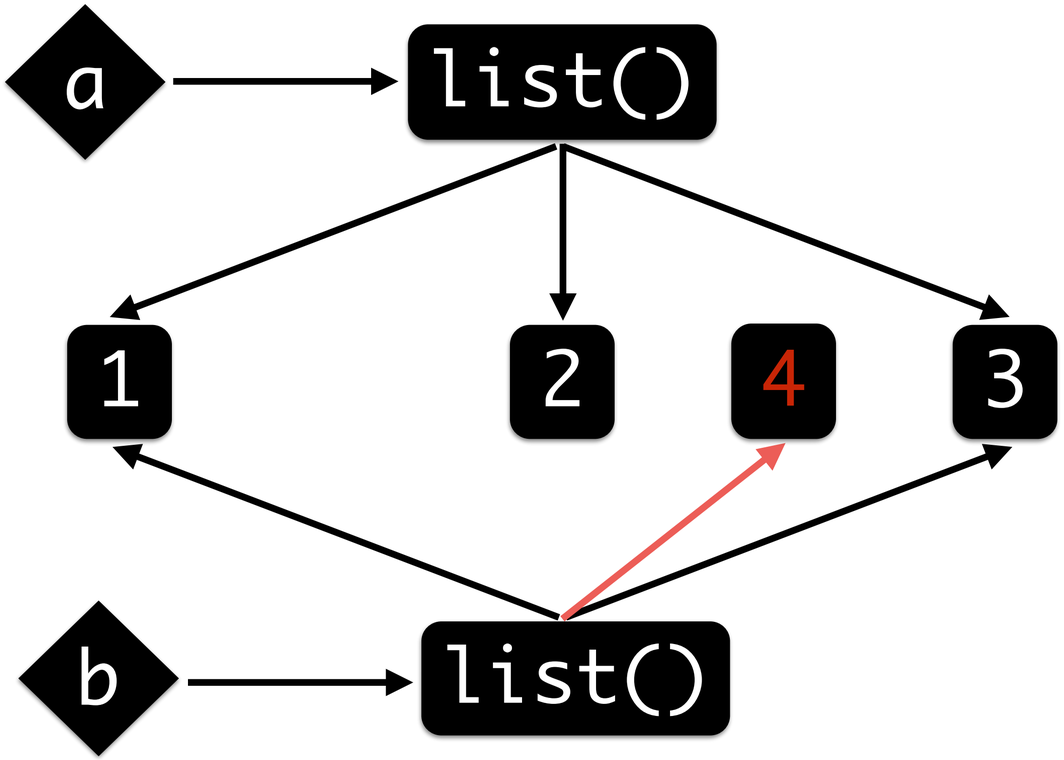


Копия при помощи конструктора

Исходный код

```
a = [1, 2, 3]
b = list(a)
b[1] = 4
print(a)
print(b)
```

Память

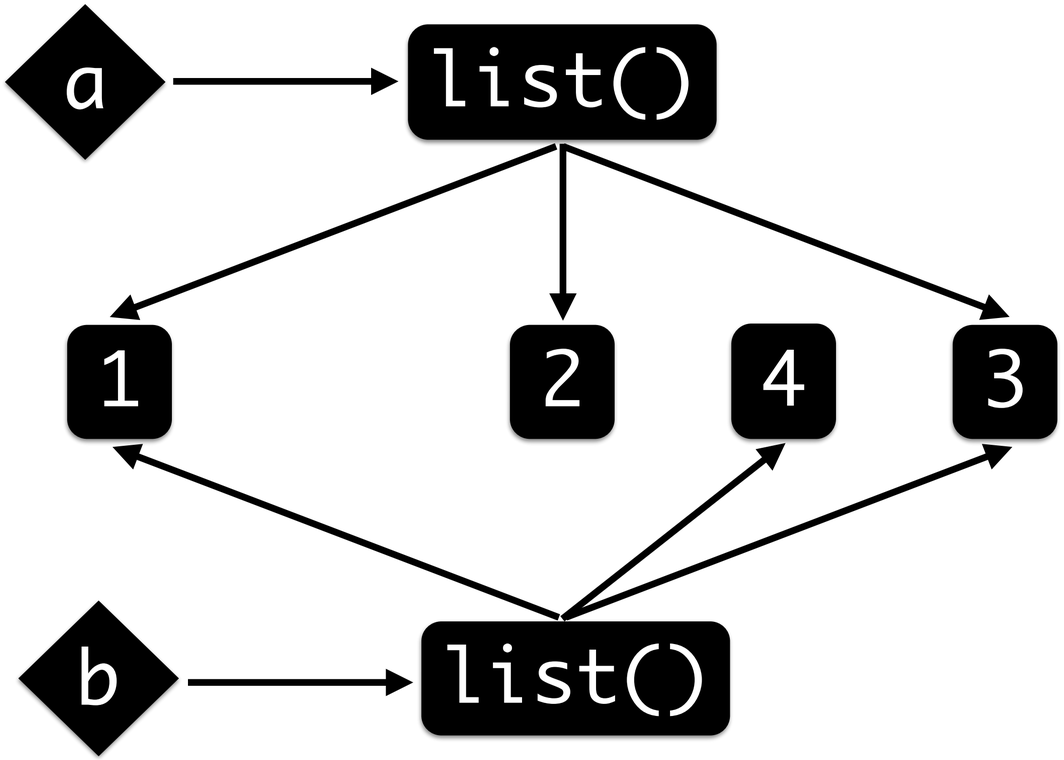


Копия при помощи конструктора

Исходный код

```
a = [1, 2, 3]
b = list(a)
b[1] = 4
print(a)
print(b)
```

Память

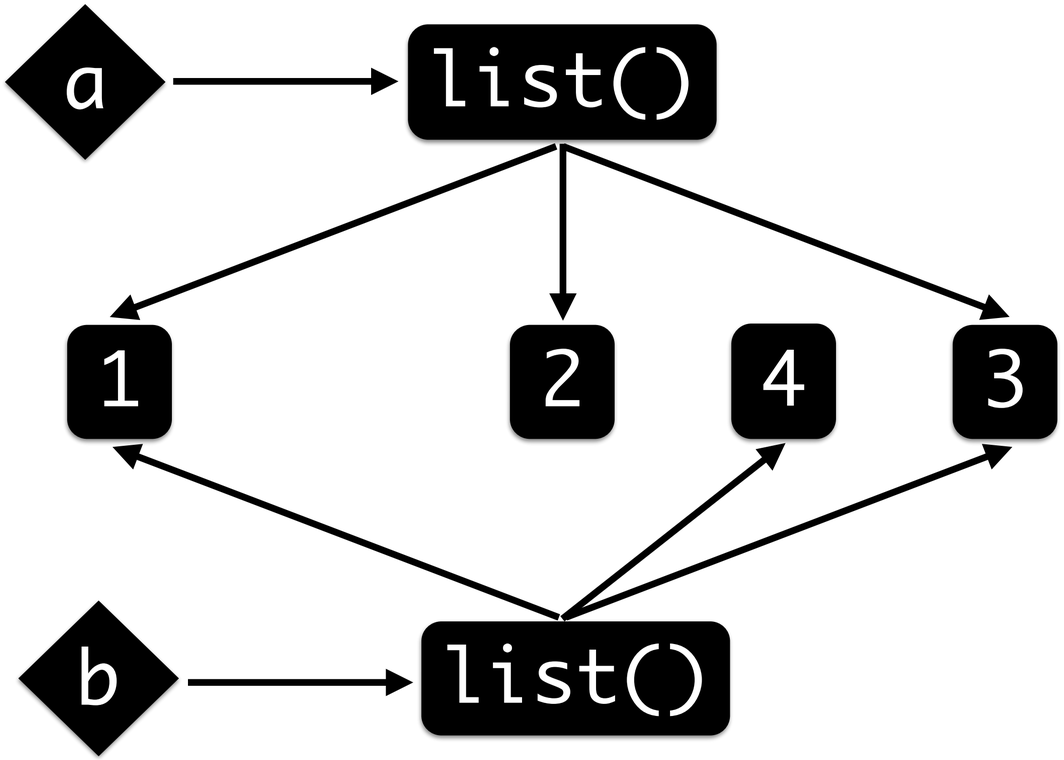


Копия при помощи конструктора

Исходный код

```
a = [1, 2, 3]
b = list(a)
b[1] = 4
print(a)
print(b)
```

Память



Копия при помощи конструктора

Исходный код

```
a = [1, [2, 3]]
b = list(a)
b[1][0] = 4
print(a)
print(b)
```

Память

Копия при помощи конструктора

Исходный код

```
a = [1, [2, 3]]
b = list(a)
b[1][0] = 4
print(a)
print(b)
```

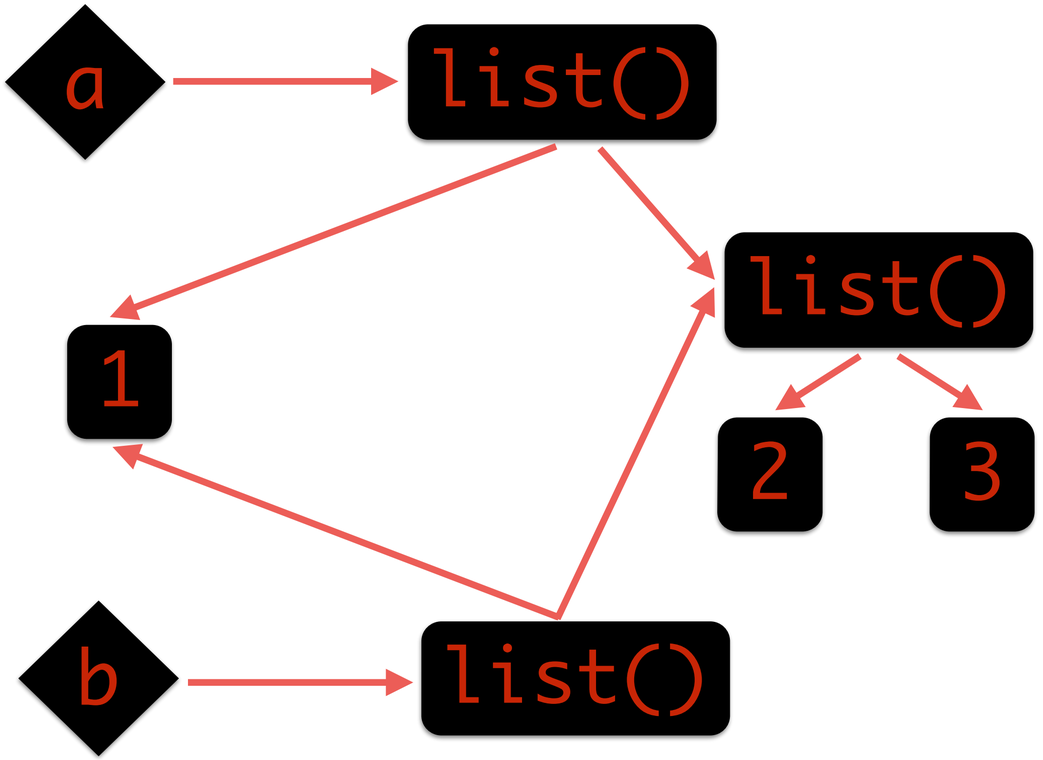
Память

Копия при помощи конструктора

Исходный код

```
a = [1, [2, 3]]
b = list(a)
b[1][0] = 4
print(a)
print(b)
```

Память

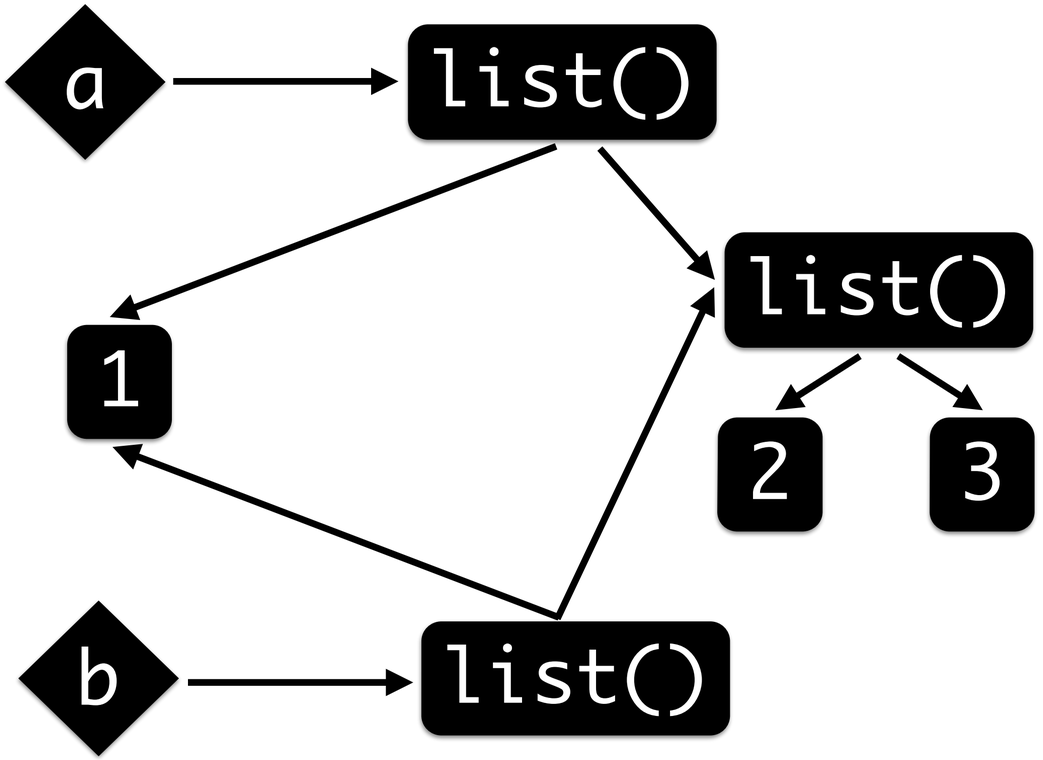


Копия при помощи конструктора

Исходный код

```
a = [1, [2, 3]]
b = list(a)
b[1][0] = 4
print(a)
print(b)
```

Память

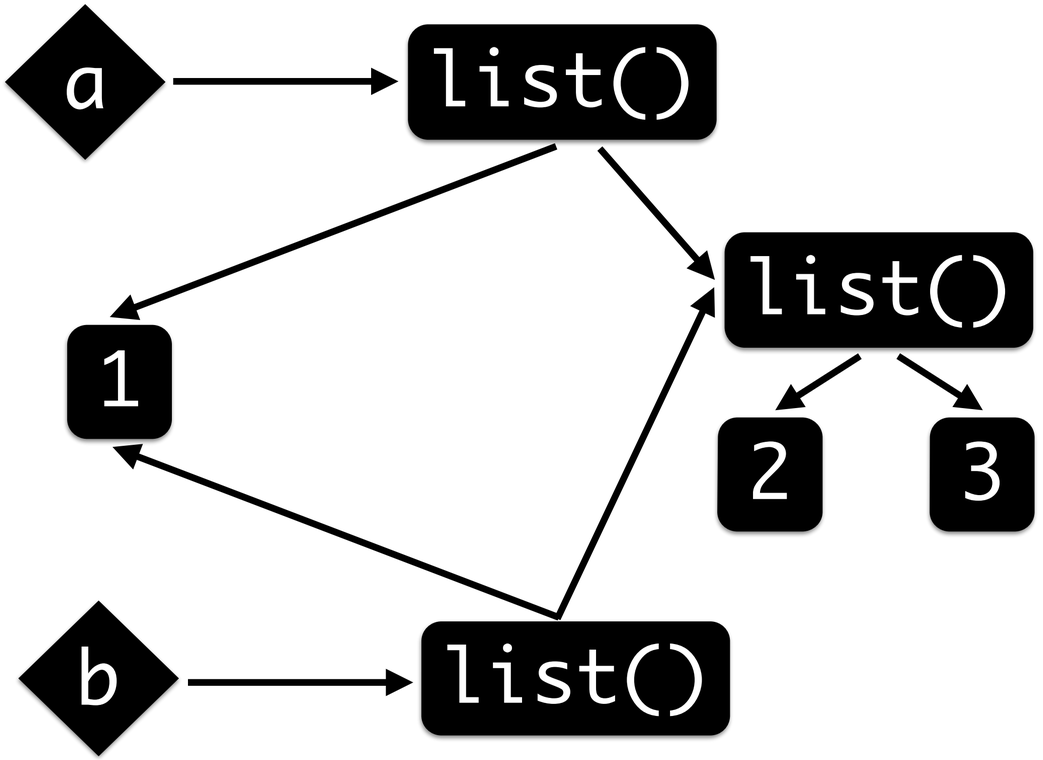


Копия при помощи конструктора

Исходный код

```
a = [1, [2, 3]]
b = list(a)
b[1][0] = 4
print(a)
print(b)
```

Память

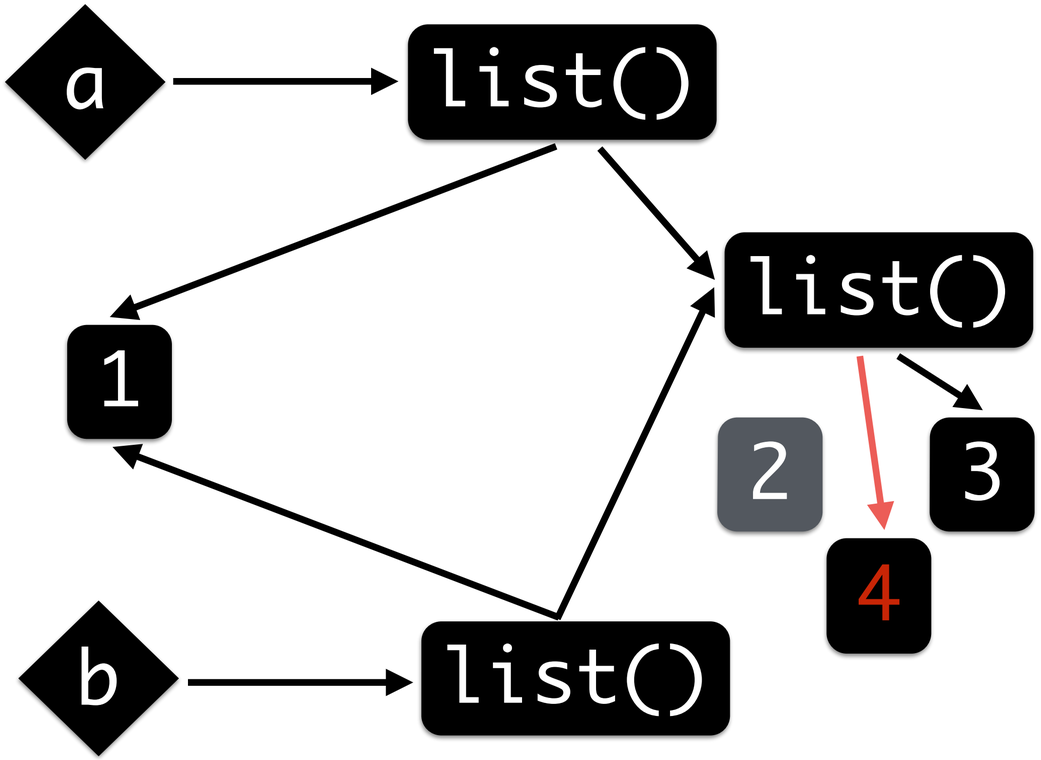


Копия при помощи конструктора

Исходный код

```
a = [1, [2, 3]]
b = list(a)
b[1][0] = 4
print(a)
print(b)
```

Память

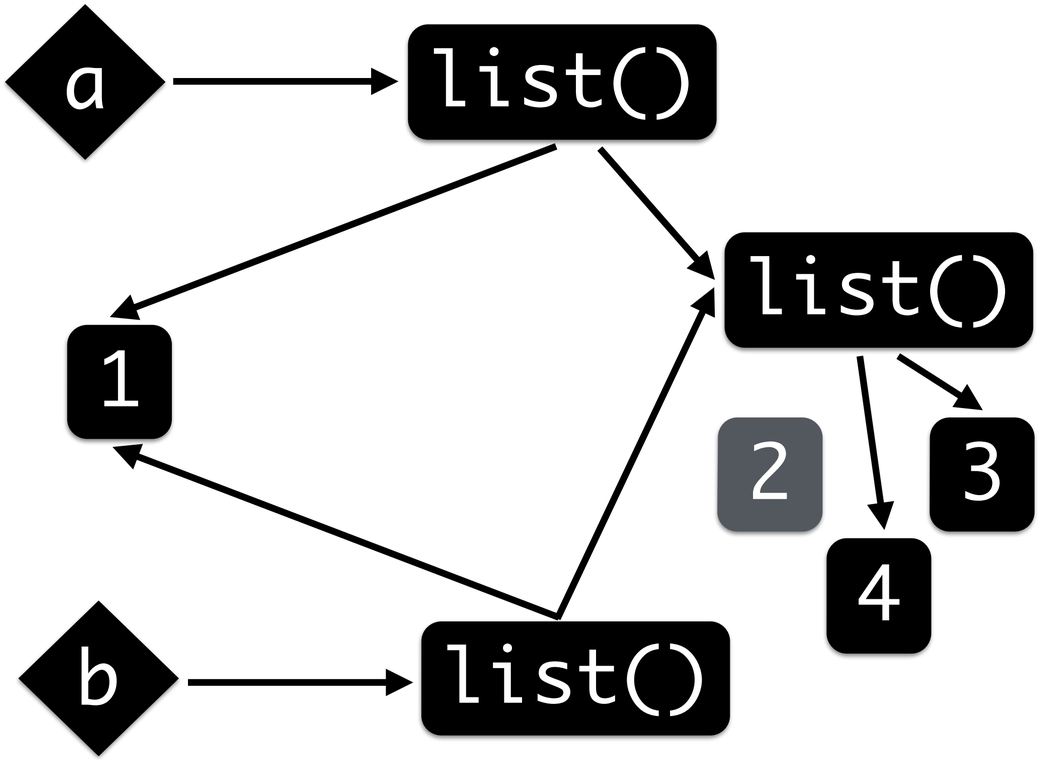


Копия при помощи конструктора

Исходный код

```
a = [1, [2, 3]]
b = list(a)
b[1][0] = 4
print(a)
print(b)
```

Память



Такое копирование называется поверхностным (shallow copy)

Копия при помощи **copy.copy()**

Исходный код

```
import copy
a = [1, [2, 3]]
b = copy.copy(a)
b[1][0] = 4
print(a)
print(b)
```

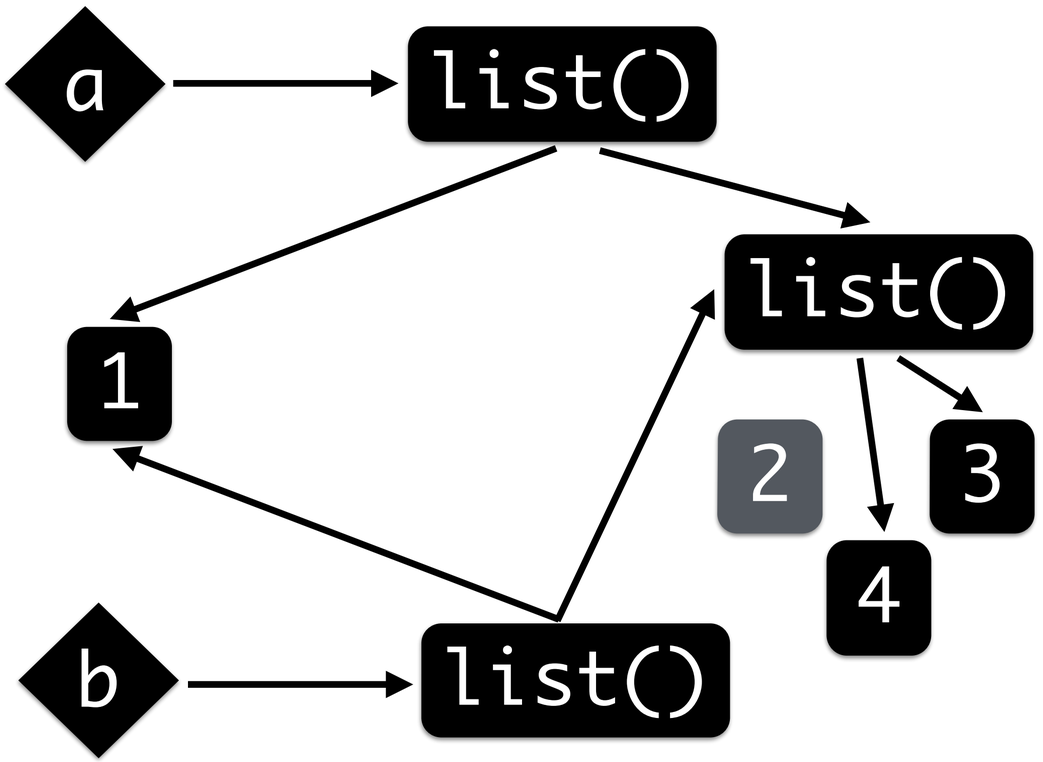
Память

Копия при помощи **copy.copy()**

Исходный код

```
import copy
a = [1, [2, 3]]
b = copy.copy(a)
b[1][0] = 4
print(a)
print(b)
```

Память



Копия при помощи **copy.deepcopy()**

Исходный код

```
import copy
a = [1, [2, 3]]
b = copy.deepcopy(a)
b[1][0] = 4
print(a)
print(b)
```

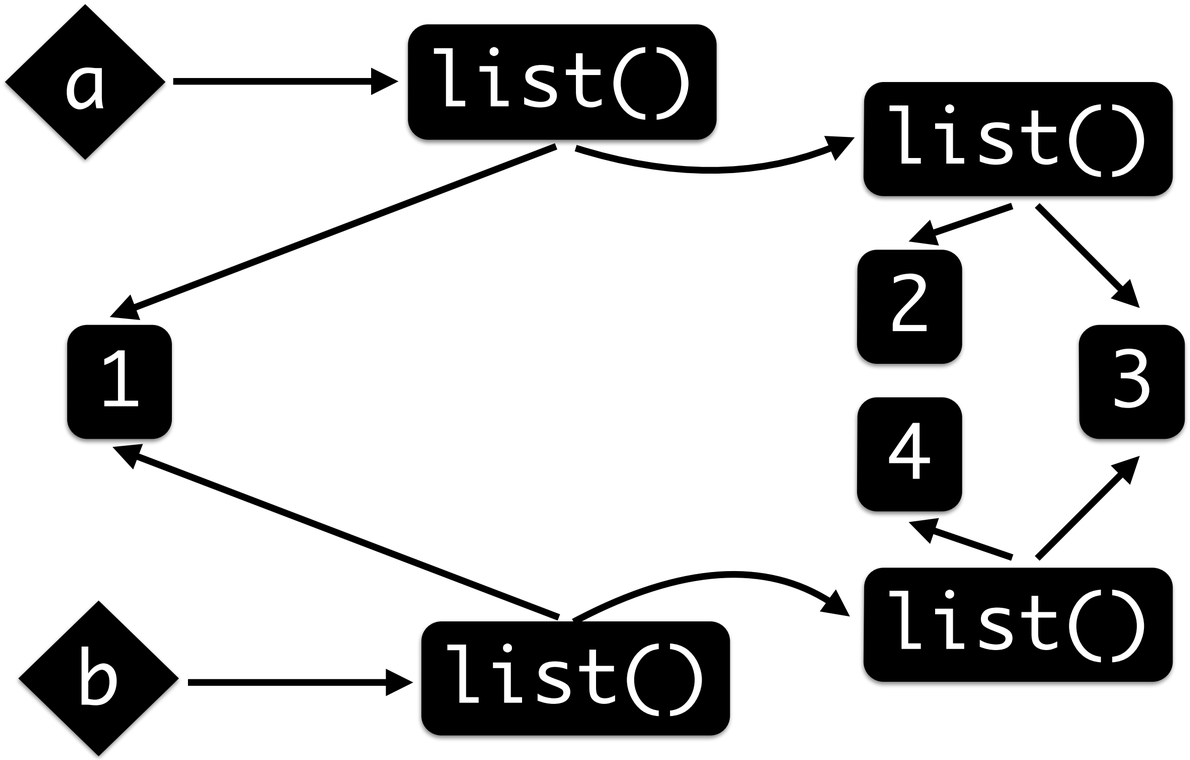
Память

Копия при помощи **copy.deepcopy()**

Исходный код

```
import copy
a = [1, [2, 3]]
b = copy.deepcopy(a)
b[1][0] = 4
print(a)
print(b)
```

Память



Но иногда даже **copy.deepcopy()** недостаточно глубок

Ветвления и циклы

Условные выражения

```
if_stmt ::=  "if" expression ":" suite
             ( "elif" expression ":" suite )*
             ["else" ":" suite]

suite           ::=  stmt_list NEWLINE
                     | NEWLINE INDENT statement+ DEDENT
statement       ::=  stmt_list NEWLINE | compound_stmt
stmt_list       ::=  simple_stmt (";" simple_stmt)* [";"]
```

Цикл **while**

```
while_stmt ::=  "while" expression ":" suite
                ["else" ":" suite]
```

```
"break"
"continue"
```

Цикл **for**

```
for_stmt ::=  "for" target_list "in" expression_list ":" suite
              ["else" ":" suite]
```

Устройство цикла **for**

```
product = 1
for i in [1, 2, 4, 8]:
    product *= i
print(product)
```

```
product = 1
i = iter([1, 2, 4, 8])
while True:
    try:
        product *= next(i)
    except StopIteration:
        break
print(product)
```

Функции

str.format