при поддержке
{а}вижн

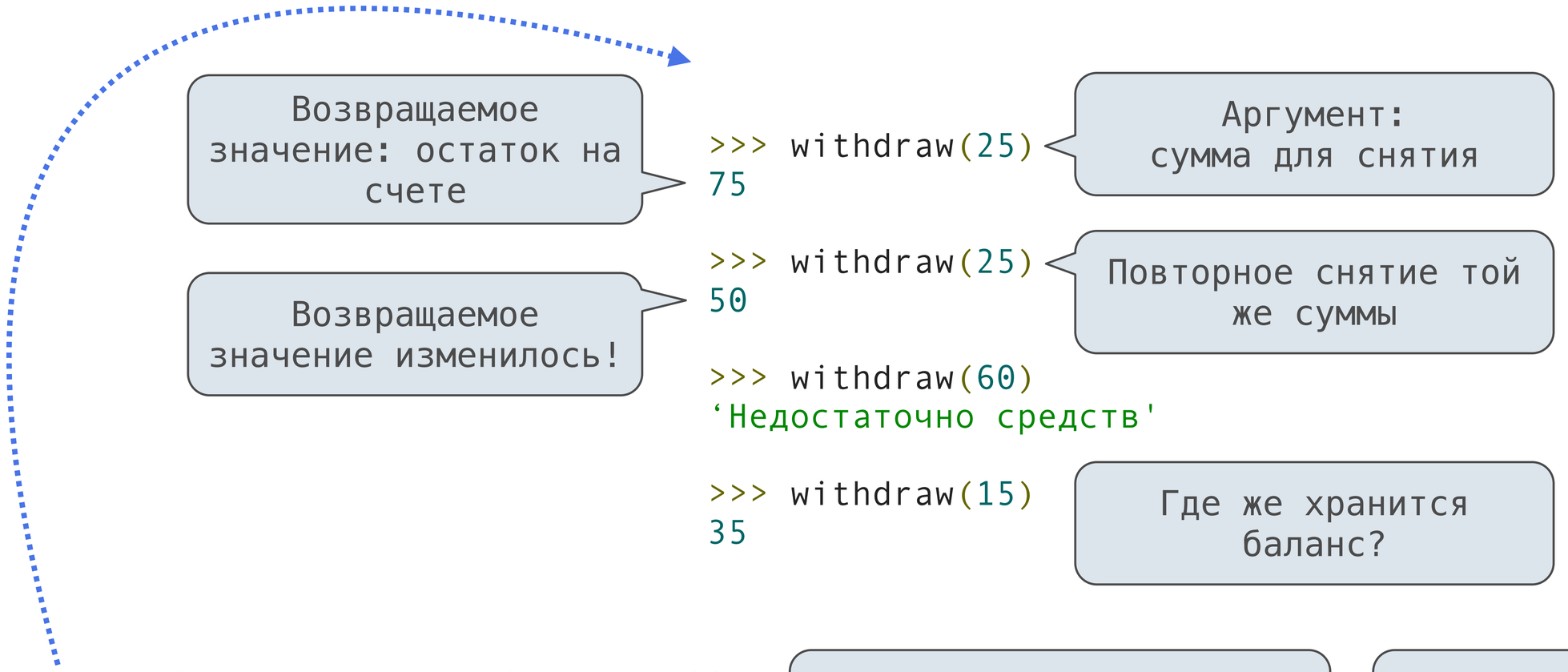
Лекция 13

Перевод и адаптация материалов Джона ДеНиро (John DeNero). Используется с разрешения автора.

Изменчивые функции

Функция, поведение которой меняется с течением времени

Смоделируем банковский счет с начальным балансом в 100 ₽

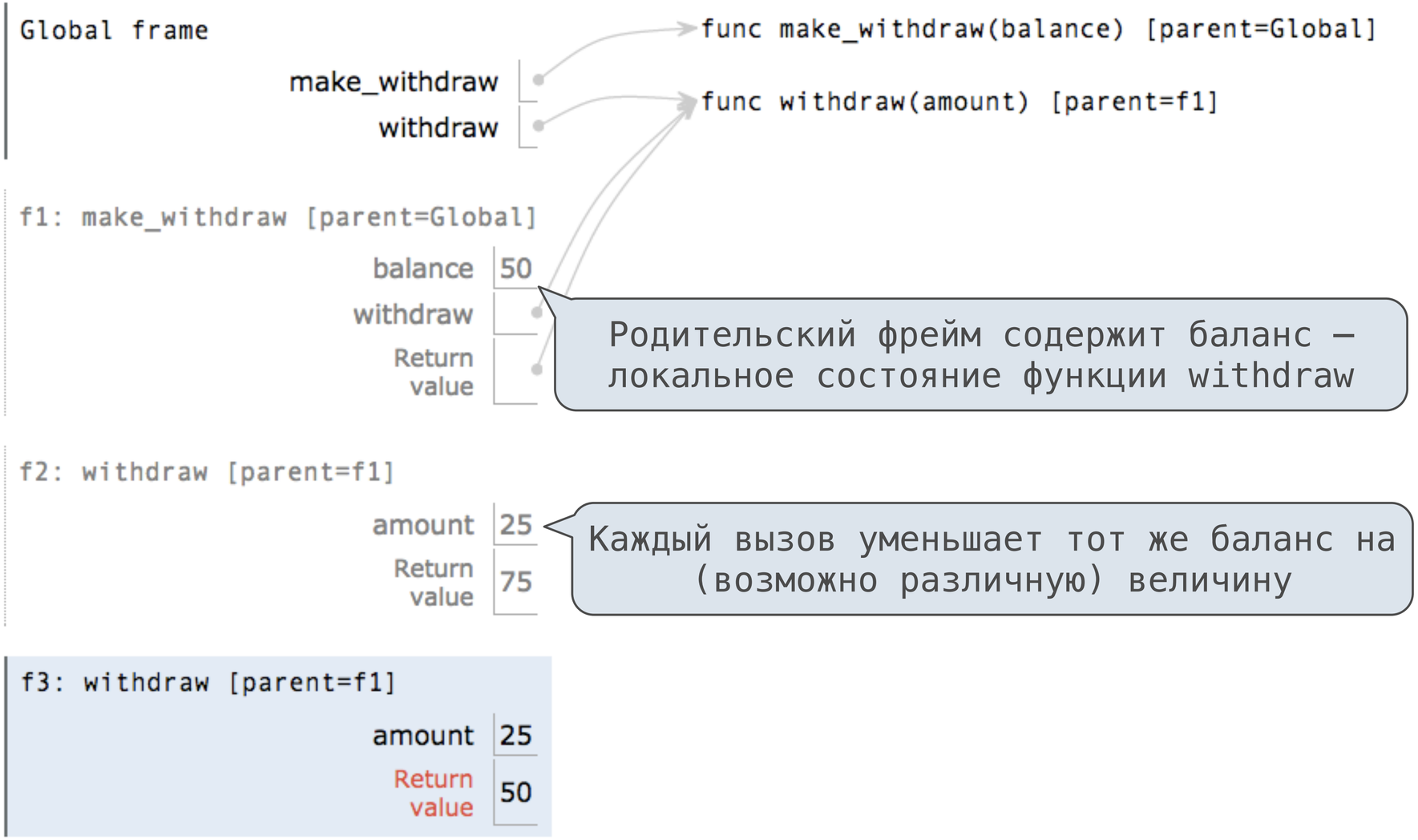



```
>>> withdraw = make_withdraw(100)
```

В родительском фрейме функции!

У функции есть тело и родительское окружение

Хранение локального состояния с помощью окружений

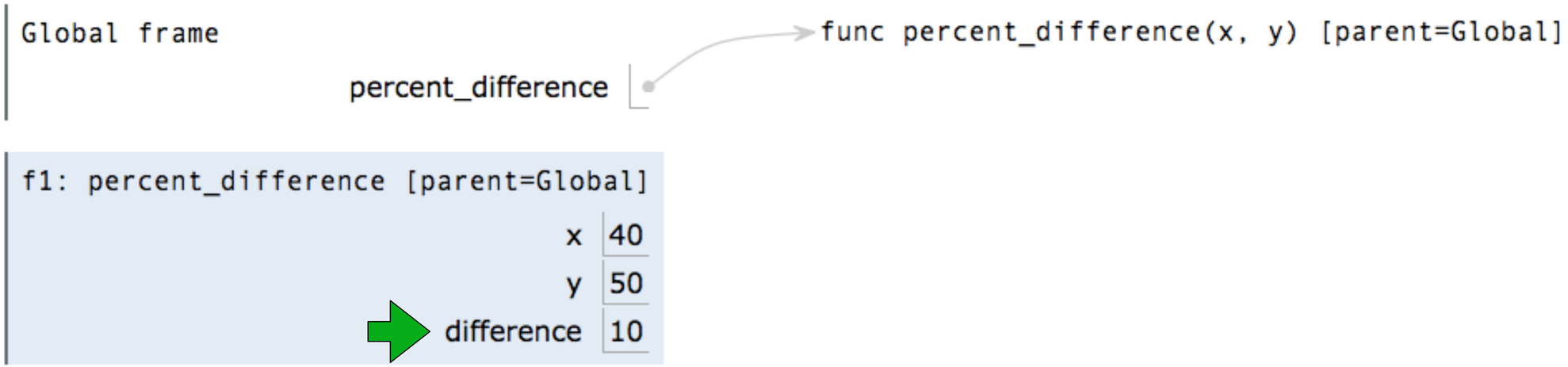


Интерактивная диаграмма

Повторение: локальное присвоение

```
def percent_difference(x, y):
    difference = abs(x-y)
    return 100 * difference / x
diff = percent_difference(40, 50)
```

Присвоение связывает имя со значением в первом фрейме текущего окружения



Правила выполнения инструкции присвоения:

1. Вычислить все выражения справа от =, слева направо.

2. Связать имена слева от = с полученными значениями в **текущем фрейме**

Интерактивная диаграмма

Нелокальное присвоение и хранение локального состояния

```
def make_withdraw(balance):
    """Возвращает функцию withdraw с начальным балансом."""
    def withdraw(amount):
        nonlocal balance
        if amount > balance:
            return 'Недостаточно средств'
        balance = balance - amount
        return balance
    return withdraw
```

Имя «balance» объявлено нелокальным в начале тела функции, в которой произойдет пересвязывание

Пересвязывание баланса в первом нелокальном фрейме, в котором это имя уже существует

(Пример)

Нелокальное присвоение

Действие нелокальных присвоений

```
nonlocal <имя>, <имя>, ...
```

Результат: Последующие присвоения этому имени влияют только на уже существующую связь в **первом нелокальном фрейме** текущего окружения, в котором это имя связано.

Документация Python: в «области видимости»

Из руководства по Python 3:

Указанные в инструкции nonlocal имена должны существовать в области видимости.

Указанные в инструкции nonlocal имена не должны совпадать с существующими в локальной области видимости.

Текущем фрейме

http://docs.python.org/release/3.1.3/reference/simple_stmts.html#the-nonlocal-statement

http://www.python.org/dev/peps/pep-3104/

Множественные толкования инструкции присвоения

```
x = 2
```

Состояние	Результат
• Нет инструкции nonlocal • «x» **не** связано локально	Создается связь имени «x» с объектом 2 в первом фрейме текущего окружения
• Нет инструкции nonlocal • «x» **связано** локально	Пересвязывание имени «x» с объектом 2 в первом фрейме текущего окружения
• nonlocal x • «x» **связано** в нелокальном фрейме	Пересвязывание «x» с 2 в первом нелокальном фрейме текущего окружения, в котором присутствует «x»
• nonlocal x • «x» **не** связано в нелокальном фрейме	`SyntaxError: no binding for nonlocal 'x' found`
• nonlocal x • «x» **связано** в нелокальном фрейме • «x» также связано локально	`SyntaxError: name 'x' is parameter and nonlocal`

Особенность Python

Python предвычисляет, какой фрейм содержит какое имя до выполнения тела функции.

Внутри тела функции все упоминания имени должны относиться к одному фрейму.

```
def make_withdraw(balance):
    def withdraw(amount):
        if amount > balance:
            return 'Недостаточно средств'
        balance = balance - amount
        return balance
    return withdraw

wd = make_withdraw(20)
wd(5)
```

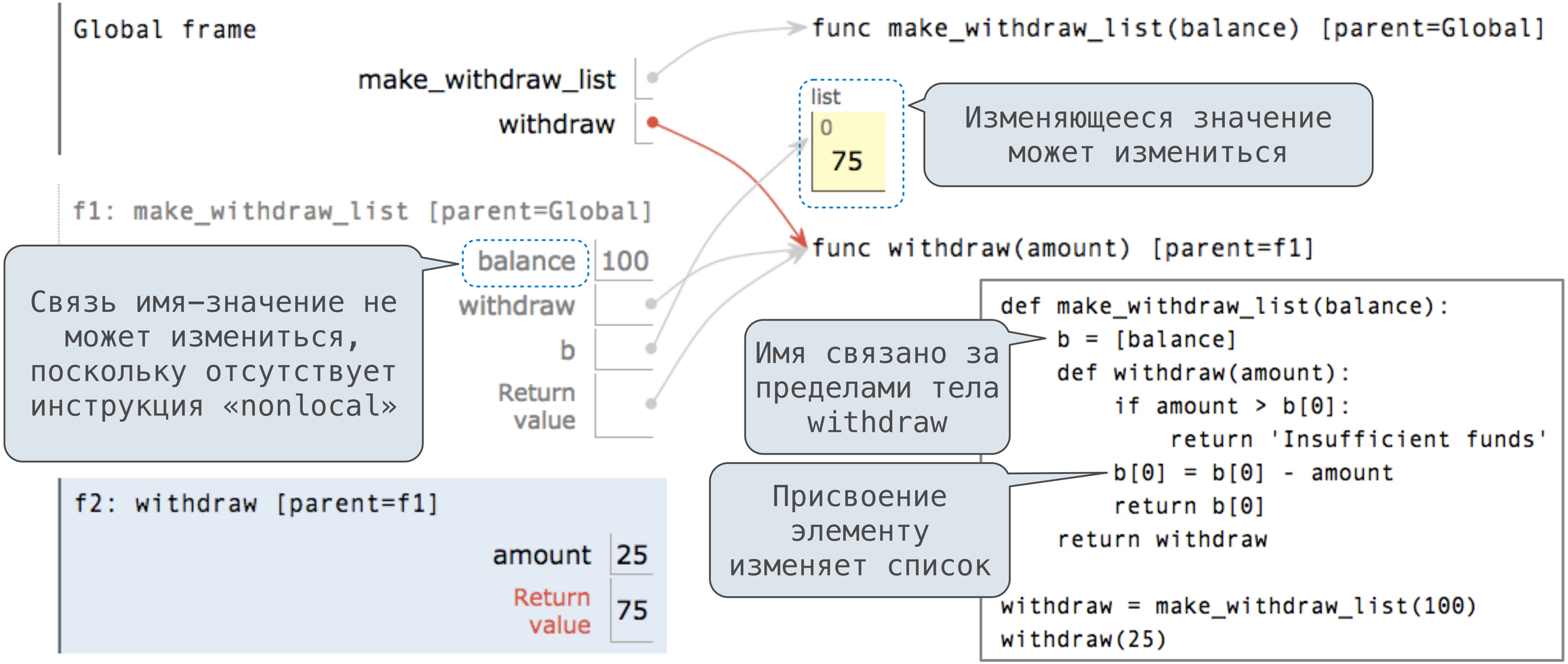
Локальное присвоение

UnboundLocalError: local variable 'balance' referenced before assignment

Интерактивная диаграмма

Изменяющиеся значения и хранение локального состояния

Изменяющиеся значения могут быть изменены *без* инструкции «nonlocal».



Интерактивная диаграмма

Множественные изменчивые функции

(Пример)

Ссылочная прозрачность

- Выражение называют **прозрачным для ссылок,** если при замене любого подвыражения его результатом не происходит изменение смысла программы.

```
mul(add(2, mul(4, 6)), add(3, 5))

mul(add(2,    24    ), add(3, 5))

mul(       26         , add(3, 5))
```

- Изменяющиеся операции нарушают условие *прозрачности для ссылок* поскольку они не только возвращают результат; **они изменяют окружение.**