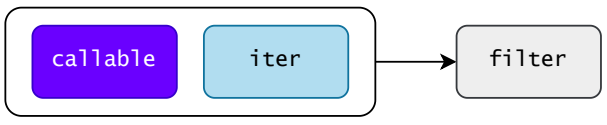
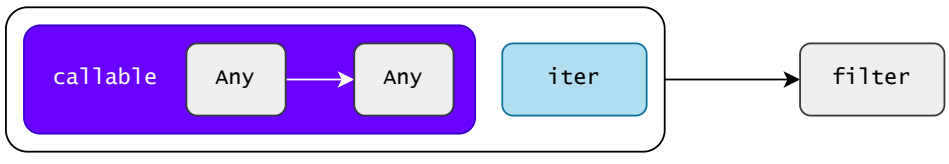
FILTER

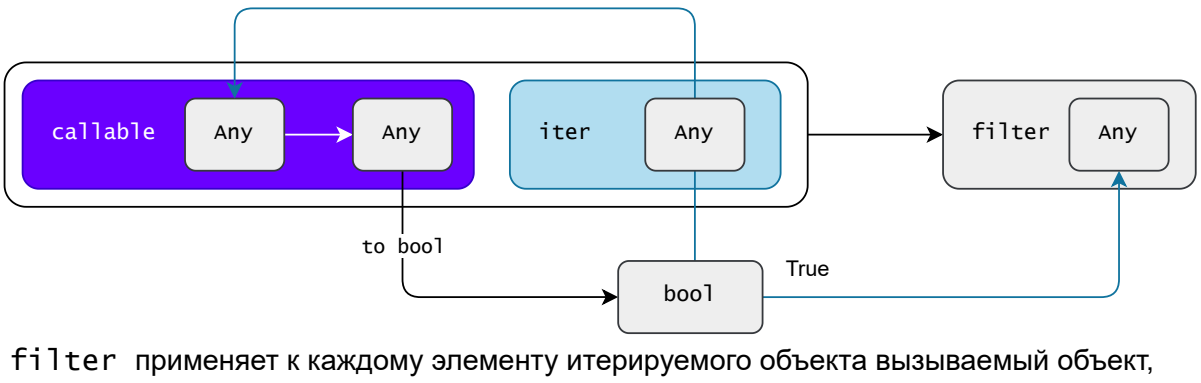
`filter(callable, iterable)`



Первый аргумент это вызываемый тип, второй - итерируемый



Вызываемый тип должен принимать на вход только 1 аргумент
Но стоит обратить внимание что на вход `callable` будут подаваться элементы из `iter`



`filter` применяет к каждому элементу итерируемого объекта вызываемый объект, возвращенный ответ преобразуется в `bool`, если `True`,
то этот **исходный элемент будет сохранён в результате фильтрации**

Пример 1: Отфильтровывает список и оставляет только нечетные элементы

Через обычную функцию

```python
list1 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

def is_odd(n):
    return n % 2 != 0

list2 = list(filter(is_odd, list1))

print(list2)  # [1, 3, 5, 7, 9]
```

Через lambda функцию

```python
list1 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

list2 = list(filter(lambda x: x % 2 != 0, list1))

print(list2)  # [1, 3, 5, 7, 9]
```

Список является итерируемым объектом

Пример 2: Получить список только тех отзывов, где оценка равна 5

Через обычную функцию

```python
reviews = [
    {
        "reviewId": 501,
        "userId": 101,
        "username": "techguy123",
        "rating": 5,
        "comment": "Amazing sound quality and battery life!"
    },
    {
        "reviewId": 502,
        "userId": 102,
        "username": "jane_doe",
        "rating": 4,
        "comment": "Great headphones but a bit pricey."
    }
]


def is_rating_5(review):
    return review["rating"] == 5

rating_5_reviews = list(filter(is_rating_5, reviews))
```

Через lambda функцию

```python
reviews = [
    {
        "reviewId": 501,
        "userId": 101,
        "username": "techguy123",
        "rating": 5,
        "comment": "Amazing sound quality and battery life!"
    },
    {
        "reviewId": 502,
        "userId": 102,
        "username": "jane_doe",
        "rating": 4,
        "comment": "Great headphones but a bit pricey."
    }
]


rating_5_reviews = list(filter(lambda x: x["rating"] == 5, reviews))
```