

Генераторы

In [1]:

```
def even_range(start, end):
    current = start
    while current < end:
        yield current
        current += 2
```

In [2]:

```
for number in even_range(0, 10):
    print(number)
```

0
2
4
6
8

In [3]:

```
ranger = even_range(0, 4)
```

In [4]:

```
next(ranger)
```

Out[4]: 0

In [5]:

```
next(ranger)
```

Out[5]: 2

In [6]:

```
next(ranger)
```

StopIteration Traceback (most recent call last)
<ipython-input-6-9065b0f81b55> in <module>()
----> 1 next(ranger)

StopIteration:

In [7]:

```
def list_generator(list_obj):
    for item in list_obj:
        yield item
        print('After yielding {}'.format(item))

generator = list_generator([1, 2])
```

In [8]:

```
next(generator)
```

Out[8]: 1

In [9]:

```
next(generator)
```

After yielding 1

Out[9]: 2

In [10]:

```
next(generator)
```

After yielding 2

StopIteration Traceback (most recent call last)
<ipython-input-10-1d0a8ea12077> in <module>()
----> 1 next(generator)

StopIteration:

In [11]:

```
def fibonacci(number):
    a = b = 1
    for _ in range(number):
        yield a
        a, b = b, a + b
```

In [12]:

```
for num in fibonacci(10):
    print(num)
```

1
1
2
3
5
8
13
21
34
55

In [13]:

```
def accumulator():
    total = 0
    while True:
        value = yield total
        print('Got: {}'.format(value))

        if not value: break
        total += value

generator = accumulator()
```

In [14]:

```
next(generator)
```

Out[14]: 0

In [15]:

```
print('Accumulated: {}'.format(generator.send(1)))
```

Got: 1
Accumulated: 1

In [16]:

```
print('Accumulated: {}'.format(generator.send(1)))
```

Got: 1
Accumulated: 2

In [17]:

```
print('Accumulated: {}'.format(generator.send(1)))
```

Got: 1
Accumulated: 3

In [18]:

```
next(generator)
```

Got: None

StopIteration Traceback (most recent call last)
<ipython-input-18-1d0a8ea12077> in <module>()
----> 1 next(generator)

StopIteration: