**26.4 Реализация итераторов**

#### 

#### Практика

##### **Задача 1. Бесконечный итератор**

Реализуйте итератор-счётчик, который не принимает никаких атрибутов и имеет только один статический атрибут — счётчик. Итератор увеличивает счётчик и возвращает предыдущее значение. У вас должен получиться бесконечный итератор.

То есть при вызове такого кода в основной программе

my\_iter = СountIterator()

for i\_elem in my\_iter:

    print(i\_elem)

значения будут выдаваться бесконечно.

##### **Задача 2. Случайная сумма**

Алексею по работе необходимо обрабатывать огромные массивы данных из миллионов элементов. Каждый новый элемент — это сумма случайного вещественного числа от 0 до 1 и предыдущего элемента (первый элемент — просто случайное вещественное число от 0 до 1). Алексею нельзя хранить в памяти весь этот список, а со значениями работать как-то надо.

Помогите Алексею, реализуйте такой класс-итератор и проверьте его работу. Также сделайте, чтобы при каждом новом вызове итератора в цикле значения считались заново.

Пример работы программы:

Кол-во элементов: 5

Элементы итератора:

0.74

1.13

1.95

2.2

2.55

Новое кол-во элементов: 6

Элементы итератора:

0.79

1.58

2.55

2.81

3.06

3.34

##### **Задача 3. Простые числа**

Реализуйте класс-итератор Primes, который принимает максимальное число N и выдаёт все простые числа от 1 до N.

Основной код:

prime\_nums = Primes(50)

for i\_elem in prime\_nums:

    print(i\_elem, end=' ')

Ожидаемый результат кода:

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47