Вам дается последовательность целых чисел и вам нужно ее обработать и вывести на экран сумму первой пятерки чисел из этой последовательности, затем сумму второй пятерки, и т. д.

Но последовательность не дается вам сразу целиком. С течением времени к вам поступают её последовательные части. Например, сначала первые три элемента, потом следующие шесть, потом следующие два и т. д.

Реализуйте класс **Buffer**, который будет накапливать в себе элементы последовательности и выводить сумму пятерок последовательных элементов по мере их накопления.

Одним из требований к классу является то, что он не должен хранить в себе больше элементов, чем ему действительно необходимо, т. е. он не должен хранить элементы, которые уже вошли в пятерку, для которой была выведена сумма.

Класс должен иметь следующий вид

```
class Buffer:
    def __init__(self):
        # конструктор без аргументов

    def add(self, *a):
        # добавить следующую часть последовательности

    def get_current_part(self):
        # вернуть сохраненные в текущий момент элементы последовательности в
порядке, в котором они были
        # добавлены
```

Пример работы с классом

```
buf = Buffer()
buf.add(1, 2, 3)
buf.get_current_part() # вернуть [1, 2, 3]
buf.add(4, 5, 6) # print(15) – вывод суммы первой пятерки элементов
buf.get_current_part() # вернуть [6]
buf.add(7, 8, 9, 10) # print(40) – вывод суммы второй пятерки элементов
buf.get_current_part() # вернуть []
buf.add(1, 1, 1, 1, 1, 1, 1, 1, 1, 1) # print(5), print(5) – вывод сумм третьей
и четвертой пятерки
buf.get_current_part() # вернуть [1]
```

Обратите внимание, что во время выполнения метода **add** выводить сумму пятерок может потребоваться несколько раз до тех пор, пока в буфере не останется менее пяти элементов.