```
"""Урок 7 Задание 3
Реализовать программу работы с органическими клетками, состоящими из ячеек. Необходимо
создать класс Клетка. В его конструкторе инициализировать параметр, соответствующий
количеству ячеек клетки (целое число). В классе должны быть реализованы методы
перегрузки арифметических операторов: сложение ( __add__() ), вычитание ( __sub__() ),
умножение ( __mul__() ), деление ( __truediv__() ). Данные методы должны применяться только
к клеткам и выполнять увеличение, уменьшение, умножение и целочисленное (с округлением
до целого) деление клеток, соответственно.
Сложение . Объединение двух клеток. При этом число ячеек общей клетки должно равняться
сумме ячеек исходных двух клеток.
Вычитание . Участвуют две клетки. Операцию необходимо выполнять только если разность
количества ячеек двух клеток больше нуля, иначе выводить соответствующее сообщение.
Умножение . Создается общая клетка из двух. Число ячеек общей клетки определяется как
произведение количества ячеек этих двух клеток.
Деление . Создается общая клетка из двух. Число ячеек общей клетки определяется как
целочисленное деление количества ячеек этих двух клеток.
В классе необходимо реализовать метод make_order() , принимающий экземпляр класса и
количество ячеек в ряду. Данный метод позволяет организовать ячейки по рядам.
Метод должен возвращать строку вида *****\n*****\n***** ..., где количество ячеек между \n
равно переданному аргументу. Если ячеек на формирование ряда не хватает, то в последний
ряд записываются все оставшиеся.
Например, количество ячеек клетки равняется 12, количество ячеек в ряду — 5. Тогда метод
make_order() вернет строку: *****\n*****\n** .
Или, количество ячеек клетки равняется 15, количество ячеек в ряду — 5. Тогда метод
make_order() вернет строку: *****\n*****\n***** .
"""
```

```
27
28 class Cell:
29     def __init__(self, nums):
30         self.nums = nums
31
32     def make_order(self, rows):
33         return '\n'.join(['*' * rows for _ in range(
   self.nums // rows)]) + '\n' + '*' * (self.nums % rows
   )
34
35     def __str__(self):
36         return self.nums
37
38     def __add__(self, other):
39         return f'Sum of cell is: {self.nums + other.
   nums}'
40
41     def __sub__(self, other):
42         return self.nums - other.nums if self.nums -
   other.nums > 0 \
43             else "Ячеек в первой клетке меньше равно
   второй, вычитание невозможно!"
44
45     def __mul__(self, other):
46         return f'Multiply of cells is: {self.nums *
   other.nums}'
47
48     def __truediv__(self, other):
49         return f'Truediv of cells is: {round(self.
   nums / other.nums)}'
50
51
52 cell_1 = Cell(15)
53 cell_2 = Cell(24)
54 print(cell_1 + cell_2)
55 print(cell_1 - cell_2)
56 print(cell_1 * cell_2)
57 print(cell_1 / cell_2)
58 print(cell_2.make_order(7))
```