Практические задания по темам (Классы в языке Python. Определение данных, методов, операций. Наследование. Множественное наследование. Композиция при разработке классов.)

1. Исправьте в коде все ошибки так, чтобы скрипт заработал.

```
#Создаём суперкласс.
class SuperClass():

    # Конструктор суперкласса.
    def __init__(self):
        self.num = num

    # Метод суперкласса.
    def get_num():
        print(self.num)

# Создаем подкласс.
class SubClass(self, SuperClass):

    # Конструктор подкласса.
    def __init__(self, num):
        # Вызываем конструктор суперкласса.
        super()._init_(self, num)
        print('Экземпляр создан!')

# Создаем 1-й экземпляр подкласса.
obj_1 = SubClass()
# Выводим значение атрибута.
print(obj_1.num)

# Создаем 2-й экземпляр подкласса.
obj_2 = SubClass(5)
# Выводим значение атрибута.
obj_1.get_num(self)
```

2. Создайте простейший в мире класс SimplePass. Затем создайте экземпляр класса и выведите на экран его тип.

3. Определите класс A, включающий:

- строку документирования класса '"Класс A"';
- метод set_a() для установки значения атрибута a;
- метод get_a() для получения значения этого атрибута.

Выведите на экран документацию класса. Затем создайте первый экземпляр класса и при помощи определенных методов установите и выведите на экран значение его атрибута a. Далее создайте второй экземпляр класса, после чего также установите и выведите на экран значение атрибута a, но уже при помощи прямого доступа к атрибуту по точке.

4. Определите класс B, включающий:

- строку документирования класса '"Класс B"';
- конструктор, инициализирующий атрибут данных b создаваемых экземпляров;
- метод get_b() для получения значения этого атрибута.

Выведите на экран документацию класса. Затем создайте экземпляр класса obj и при помощи метода экземпляра выведите на экран значение его атрибута b.

5. Определите класс C, наследующий классы A (задача №3) и B (задача №4) и включающий:

- строку документирования класса '"Класс C = A + B"';
- конструктор, инициализирующий дополнительно атрибуты данных a и c создаваемых экземпляров;
- собственные методы set_b() и set_c() для установки значений соответствующих атрибутов;
- собственный метод get_c() для получения значения атрибута c.

Выведите на экран документацию класса. Затем создайте экземпляр класса obj, после чего при помощи соответствующих методов экземпляра выведите на экран значения его атрибутов a, b и c.

6. Определите класс D, включающий:

- статический метод stat_print_dict, выводящий на экран словарь атрибутов переданного ему объекта класса;
- метод класса cls_print_dict, выводящий на экран словарь атрибутов своего класса.

Создайте экземпляр класса obj и, вызвав оба метода из этого экземпляра, выведите на экран словарь атрибутов класса D. Объясните различие в использовании методов.

7. Определите класс E, наследующий класс D (задача №6) и включающий единственный атрибут данных класса e = 'Класс E'. Создайте экземпляр obj_1 класса D и, вызвав оба метода из этого экземпляра, выведите на экран словарь атрибутов класса. Затем создайте экземпляр obj_2 класса E и также, вызвав оба метода из этого экземпляра, выведите на экран словарь атрибутов этого класса. Объясните результаты.

8. Определите класс F, наследующий класс A (задача №3), включающий:

- конструктор, обновляющий строку документации создаваемых экземпляров на 'Объект класса F';
- расширенный метод set_a() для установки значения атрибута a, который должен дополнительно выводить сообщение 'Атрибут a установлен!'.

Выведите на экран документацию класса. Затем создайте экземпляр класса obj, после чего выведите его документацию и далее, при помощи соответствующих

методов экземпляра, установите и выведите на экран значение его атрибута a. При расширении метода set_a() используйте по-очереди три варианта синтаксиса для доступа к методу суперкласса (незадействованные инструкции закомментируйте).

9. Определите класс Counter, реализующий десятичный счетчик, который может увеличивать или уменьшать свое значение на единицу в заданном диапазоне, включая границы диапазона. В классе должны быть предусмотрены следующие возможности:

- конструктор для инициализации счетчика значениями по умолчанию (стартовое значение, нижняя и верхняя границы диапазона),
- метод для его инициализации произвольными значениями,
- а также методы для увеличения и уменьшения текущего значения счетчика.

Все методы класса должны принимать только именованные параметры и проверять выход текущего значения счетчика за допустимый диапазон. Создайте экземпляр счетчика со значениями по умолчанию и выведите на экран его начальные параметры. Далее проверьте его работу циклом в пределах диапазона, увеличивая и выводя на экран его текущее значение от минимально возможного до максимального. Затем переустановите счетчик, задав отрицательную нижнюю и положительную верхнюю границы, а также установив положительное стартовое значение для отсчета. Опять же, проверьте его работу циклом, уменьшая и выводя на экран его текущее значение от стартового до минимально возможного. Задайте заведомо большее количество итераций циклов в обоих случаях, обеспечив прерывание их работы при попытке выхода счетчика за пределы диапазона.

10. Определите класс Circle, представляющий окружность и включающий:

- статический метод, переводящий метры в сантиметры или наоборот;
- конструктор, инициализирующий радиус экземпляра;
- методы получения длины и площади окружности.

Используя созданный класс, расчитайте и выведите на экран длину и площадь окружности в сантиметрах зная, что ее радиус равен 2.55 метра.

11. Определите класс A, включающий метод plus() для сложения двух чисел, и класс B, включающий метод minus() для нахождения разности двух чисел. Оба метода должны инициализировать атрибуты last_res своих экземпляров списками формата [a, b, a op b] и возвращать результат арифметической операции. Затем определите класс C, содержащий конструктор для инициализации его атрибутов plus и minus соответствующими объектами методов первых двух классов. Далее создайте экземпляр класса obj, после чего найдите сумму и разность двух чисел, используя возможности созданного экземпляра.

12. Определите суперкласс Сотрудник, включающий:

- конструктор, инициализирующий имя работника, его должность (по умолчанию None) и оклад (по умолчанию 0);

- метод экземпляра для повышения оклада на какую-то часть (например, на 0.3, т.е. на 30%) с округлением результата до копеек;
- магический метод __str__ для перегрузки строкового представления объекта, который должен выводить данные о работнике в формате 'Атрибут: объект.атрибут' по одной записи на каждой строке.

Также определите подкласс Менеджер, наследующий суперкласс Сотрудник и переопределяющий метод повышения оклада таким образом, чтобы он еще больше повышал оклад за счет дополнительного бонуса в виде какой-то части оклада. Далее:

- создайте экземпляр иван_менеджер созданного подкласса с начальным окладом в 1700 рублей;
- повысьте сотруднику оклад за счет стандартной надбавки в 0.335 и бонуса за должность в 0.25;
- выведите строковое представление объекта экземпляра с информацией о сотруднике на экран.