Python (BSU FAMCS Fall'18)

Домашнее задание 2

Преподаватели: Дмитрий Косицин, Светлана Боярович

Реализуйте класс-перечисление `Flag` (**enum**), значениями которого могут быть только целые числа, представляющие собой «флаги». Значение «флага» – бит с некоторым смещением (см. пример 1).

При создании класса все значения должны проверяться на целочисленность и уникальность. Если это не так, должно быть брошено исключение.

Должна быть возможность явно не указывать значения флагов, а сгенерировать их автоматически (см. пример 2).

У каждого значения должны быть поля `name` и `value`. Также при выводе на экран значение должно быть понятно представлено (не просто число, а еще и имя) – см. пример 3.

По `Flag` должна быть предоставлена возможность проитерироваться («магический» метод `__iter__`, и, если нужно, другой с другим интерфейсом).

Обеспечьте возможность получить доступ к членам класса по имени и значению (см. пример 4). Можно перегрузить для этого некоторые операторы.

Все значения должны проходить проверку на то, что они являются целыми числами (`int`), а также значениями `Flag`'а – проходить проверку посредством оператора `in`, а также являться экземплярами класса, в котором объявлены (упрощенный вариант – экземплярами некоторого другого класса, например, `FlagValue`) – см. пример 5.

Над всеми значениями должна быть реализована возможность производить операции логического «и» и «или». Логическое «и» (`&`) будет проверять то, что некоторый бит установлен в значении, а логическое «или» (`|`) будет создавать группу флагов. Такая группа флагов также должна проходить проверку на то, что значение является значением `Flag`'а (см. пример 6).

Предусмотрите также некоторый набор юнит-тестов к вашему классу.

Пример 1

```
class MyEnum(FlaggedEnum):
    read_only = (1 << 0)
    lowercased = (1 << 2)
    immediate = (1 << 3)
```

Пример 2

```
class MyEnumAuto(FlaggedEnum):
    auto_field_1 = auto
    auto_field_2 = auto
```

Пример 3

```
assert MyEnum.read_only.name == 'read_only'
assert MyEnum.read_only.value == (1 << 0)

print(MyEnum.read_only)  # e.g. 'MyEnum.read_only: 1'
```

Пример 4

```
assert MyEnum.get_by_name('read_only') == MyEnum.read_only  # just an example
```

Пример 5

```
assert MyEnum.read_only in MyEnum

assert isinstance(MyEnum.read_only, int)

assert isinstance(MyEnum.read_only, MyEnum)
# assert isinstance(MyEnum.read_only, FlagValue)  # simplified version
```

Пример 6

```
value = MyEnum.read_only | MyEnum.immediate  # both flags are set

# check that flags are set
assert value & MyEnum.read_only and value & MyEnum.immediate

assert value in MyEnum
assert isinstance(value, int)

assert isinstance(value, MyEnum)
# assert isinstance(value, FlagValue)  # simplified version
```