Python. Домашнее задание 2

Преподаватели: Дмитрий Косицин, Светлана Боярович и Анастасия Мицкевич

Менеджер событий

В данной задаче вам предлагается продумать и реализовать **EventManager** – менеджер событий. Идея состоит в том, чтобы организовать универсальный способ выполнения некоторых действий (по сути, функций) при возникновении события.

События

События будут возникать в произвольные моменты времени и индексироваться. Событие возникает при вызове метода *notify* у объекта класса **EventManager**. Метод, в свою очередь, присваивает событию следующий индекс (начиная с 0) и выполняет все зарегистрированные действия, которые удовлетворяют условиям.

Действия

Все действия должны характеризоваться периодичностью вызова – вызываться либо на каждом k-том событии, либо на конкретных событиях (определенных индексах), либо на всех.

Важным аспектом реализации здесь является ***оптимизация***: необходимо обходить только те действия, которые *должны* выполниться при очередном возникновении события. Не следует каждый раз проверять, должно или нет выполняться определенное дейстие.

Сбор и регистрация действий

За сбор действий, которые могут быть выполнены при возникновении события, должен отвечать класс **Collector**. Он должен уметь пройти по всем модулям (в том числе вложенным) в некотором/некоторых папках (пакетах), определить, что может быть подписано, и сообщить о всех собранных действиях **EventManager**'у.

Способ определения действий, которые могут быть подписаны, определите сами, но помните, что реализация должна быть *расширяемой* и максимально *независимой*.

Также **EventManager** должен поддерживать *регистрацию* и *удаление* определенных действий между событиями.

Вызов действий

При возникновении события **EventManager** должен вызывать все зарегистрированные действия, удовлетворяющие условиям на текущий индекс события.

Действия должны вызываться согласно некоторому *приоритету*. Систему приоритетов для действий, а также то, как они будут храниться и выглядеть, продумайте сами.

Реализуйте *обработку исключений*: по умолчанию, исключения при вызове действий должны быть обработаны – залогировано сообщение об ошибке, а исключение – подавлено. Предусмотрите возможность действиям быть *критическими*: исключения, произошедшие при выполнении таких действий, должны быть залогированы и проброшены далее из **EventManager**'а.

Особенности реализации

Реализация системы должна быть ***расширяемой***: проверьте, насколько легко добавить действия, которые выполняются каждые k событий, но не более m раз. Добавьте поддержку такого типа действий.

Также реализация должна быть максимально ***эффективной***: используйте генераторы и итераторы для обхода, не создавайте коллекций-результатов явно. Реализация должна быть направлена на обработку большого количества действий при большом количестве событий, а вот регистрация и удаление действий будут выполняться нечастно.

В отдельном файле (или нескольких) реализуйте *юнит-тесты* для вашего кода. Тестируйте только самые тонкие и сложные моменты – не увлекайтесь и не тратьте время на мелочи.

Реализация, разумеется, должна быть написана в хорошем ***Python-стиле*** с соблюдением всего пройденного ранее и сделанных вам замечаний.