В сообщение скидывайте ссылку на репозиторий на GithHub

https://github.com/AleksandrRadist/BSR-2020-2021-Materials.git - тесты

Создайте два калькулятора: для подсчёта денег и калорий. Пользовательскую часть калькуляторов, их «лицо», писать не нужно, напишите только логику — отдельный класс для каждого из калькуляторов.
Калькулятор денег должен уметь:

1. Сохранять новую запись о расходах методом `add_record()`
2. Считать, сколько денег потрачено сегодня методом `get_today_stats()`
3. Определять, сколько ещё денег можно потратить сегодня в рублях, долларах или евро — метод `get_today_cash_remained(currency)`
4. Считать, сколько денег потрачено за последние 7 дней — метод `get_week_stats()`

Калькулятор калорий должен уметь:

1. Сохранять новую запись о приёме пищи— метод `add_record()`
2. Считать, сколько калорий уже съедено сегодня — метод `get_today_stats()`
3. Определять, сколько ещё калорий можно/нужно получить сегодня — метод `get_calories_remained()`
4. Считать, сколько калорий получено за последние 7 дней — метод `get_week_stats()`

У калькуляторов много пересекающихся функций: они должны уметь хранить какие-то записи (о еде или деньгах, но по сути - всё числа и даты), знать дневной лимит (сколько в день можно истратить денег или сколько калорий можно получить) и суммировать записи за конкретные даты. Всю эту общую функциональность заложите в родительский класс **Calculator**, а от него унаследуйте классы **CaloriesCalculator и CashCalculator.**
Конструктор класса **Calculator** должен принимать один аргумент — число `limit`(дневной лимит трат/калорий, который задал пользователь). В конструкторе создайте пустой список, в котором потом будут храниться записи (назовите его `records`).
Чтобы было удобнее создавать записи, создайте для них отдельный класс **Record.**В нём сохраните:

- число `amount` (денежная сумма или количество килокалорий),
- дату создания записи `date` (передаётся в явном виде в конструктор, либо присваивается значение по умолчанию — текущая дата),
- комментарий `comment`, поясняющий, на что потрачены деньги или откуда взялись калории.

Примеры таких записей:

```
# для CashCalculator
r1 = Record(amount=145, comment="Безудержный шопинг", date="08.03.2019")
r2 = Record(amount=1568, comment="Наполнение потребительской корзины",
date="09.03.2019")
r3 = Record(amount=691, comment="Катание на такси", date="08.03.2019")


# для CaloriesCalculator
r4 = Record(amount=1186, comment="Кусок тортика. И ещё один.", date="24.02.2019")
r5 = Record(amount=84, comment="Йогурт.", date="23.02.2019")
r6 = Record(amount=1140, comment="Баночка чипсов.", date="24.02.2019")
```

Подробнее о формате вывода

1. Метод `get_calories_remained()` **калькулятора калорий** должен возвращать ответ

- *«Сегодня можно съесть что-нибудь ещё, но с общей калорийностью не более N кКал»*, если лимит `limit` не достигнут,
- или *«Хватит есть!»*, если лимит достигнут или превышен.

2. Метод `get_today_cash_remained(currency)` **денежного калькулятора** должен принимать на вход код валюты: одну из строк `"rub"`, `"usd"` или `"eur"`.
Возвращает он сообщение о состоянии дневного баланса в этой валюте, округляя сумму до двух знаков после запятой (до сотых):

- *«На сегодня осталось N руб/USD/Euro»* — в случае, если лимит `limit` не достигнут,
- или *«Денег нет, держись»*, если лимит достигнут,
- или *«Денег нет, держись: твой долг - N руб/USD/Euro»*, если лимит превышен.

Курс валют укажите константами `USD_RATE` и `EURO_RATE`, прямо в теле класса *CashCalculator*. Какой курс вы укажете — не так важно, выберите любой, похожий на правду.