Lesson 4 Ruby task

Task: Healthcheck monitoring

Description:

Вам необходимо реализовать в модуле **Mtn** класс **Healthcheck**, который будет проверять сервера на доступность.

Сервера уже реализованы, вам достаточно знать, что это некий объект, у которого есть метод **#status**, работающий следующим образом:

- возвращает true, если сервер доступен и отвечает;

- возвращает ошибку **Mtn::Status400** или **Mtn::Status500**, если сервер доступен, но приболел;

- возвращает ошибку **Mtn::TimeoutError**, если сервер недоступен в принципе.

Все ошибки отнаследованы от **StandardError**.

Каждый сервис вашего **Healthcheck** должен принимать сервер при инициализации и иметь как минимум метод **#check!**, который будет проверять статус сервера. При этом:

•сервис ведет подсчет количества проверок каждого сервиса и их положительных\отрицательных статусов (смотри пример ниже)

•в случае, когда сервер доступен, возвращает строку вида: "OK (количество_положительных_ответов/общее_количество_проверок_этого_сервера")

•в случае, когда сервер недоступен, возвращает строку вида:

•"КЛАСС_ПОСЛЕДНЕЙ_ОШИБКИ_КАПСЛОКОМ (количество_положительных_ответов/общее_количество_проверок_этого_сервера)"

•если же сломался сам сервис, возвращает просто строку:

•"FAIL"

Пример проверки вашего сервиса на 3-х серверах:

```
# server1-3 приходят извне

mtn1 = Mtn::Healthcheck.new(server1)
mtn2 = Mtn::Healthcheck.new(server2)
mtn3 = Mtn::Healthcheck.new(server3)
mtn4 = Mtn::Healthcheck.new('xx не сервер или ошибка случилась внутри самого
сервиса xx')

mtn1.check! # => "OK (1/1)"
mtn2.check! # => "MTN::TIMEOUTERROR (0/1)"
mtn3.check! # => "MTN::STATUS500 (0/1)"
mtn4.check! # => "FAIL"

mtn1.check! # => "OK (2/2)"
mtn2.check! # => "MTN::TIMEOUTERROR (0/2)"
mtn3.check! # => "MTN::STATUS400 (0/2)"
mtn4.check! # => "FAIL"

# и так далее, как видно, сервер может поменять свой статус или ошибку
```

Ничего выводить на экран не нужно, это просто строка как значение!