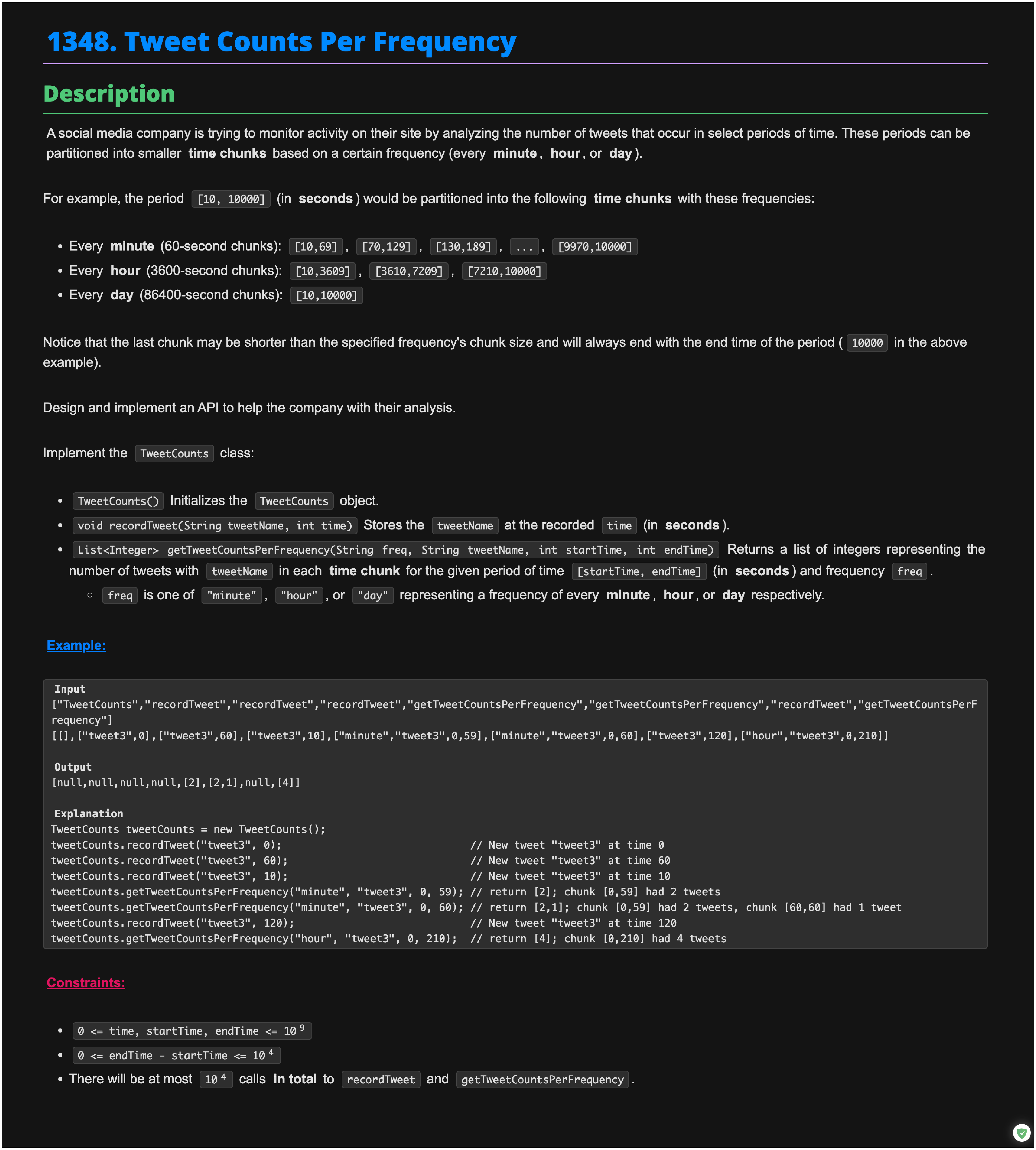

1348. Tweet Counts Per Frequency

Description

A social media company is trying to monitor activity on their site by analyzing the number of tweets that occur in select periods of time. These periods can be partitioned into smaller **time chunks** based on a certain frequency (every **minute**, **hour**, or **day**).

For example, the period `[10, 10000]` (in **seconds**) would be partitioned into the following **time chunks** with these frequencies:

- Every **minute** (60-second chunks): `[10,69]`, `[70,129]`, `[130,189]`, `...`, `[9970,10000]`
- Every **hour** (3600-second chunks): `[10,3609]`, `[3610,7209]`, `[7210,10000]`
- Every **day** (86400-second chunks): `[10,10000]`

Notice that the last chunk may be shorter than the specified frequency's chunk size and will always end with the end time of the period (`10000` in the above example).

Design and implement an API to help the company with their analysis.

Implement the `TweetCounts` class:

- `TweetCounts()` Initializes the `TweetCounts` object.
- `void recordTweet(String tweetName, int time)` Stores the `tweetName` at the recorded `time` (in **seconds**).
- `List<Integer> getTweetCountsPerFrequency(String freq, String tweetName, int startTime, int endTime)` Returns a list of integers representing the number of tweets with `tweetName` in each **time chunk** for the given period of time `[startTime, endTime]` (in **seconds**) and frequency `freq`.
  - `freq` is one of `"minute"`, `"hour"`, or `"day"` representing a frequency of every **minute**, **hour**, or **day** respectively.

Example:

```
Input
["TweetCounts","recordTweet","recordTweet","recordTweet","getTweetCountsPerFrequency","getTweetCountsPerFrequency","recordTweet","getTweetCountsPerFrequency"]
[[],["tweet3",0],["tweet3",60],["tweet3",10],["minute","tweet3",0,59],["minute","tweet3",0,60],["tweet3",120],["hour","tweet3",0,210]]

Output
[null,null,null,null,[2],[2,1],null,[4]]

Explanation
TweetCounts tweetCounts = new TweetCounts();
tweetCounts.recordTweet("tweet3", 0);                              // New tweet "tweet3" at time 0
tweetCounts.recordTweet("tweet3", 60);                             // New tweet "tweet3" at time 60
tweetCounts.recordTweet("tweet3", 10);                             // New tweet "tweet3" at time 10
tweetCounts.getTweetCountsPerFrequency("minute", "tweet3", 0, 59); // return [2]; chunk [0,59] had 2 tweets
tweetCounts.getTweetCountsPerFrequency("minute", "tweet3", 0, 60); // return [2,1]; chunk [0,59] had 2 tweets, chunk [60,60] had 1 tweet
tweetCounts.recordTweet("tweet3", 120);                            // New tweet "tweet3" at time 120
tweetCounts.getTweetCountsPerFrequency("hour", "tweet3", 0, 210);  // return [4]; chunk [0,210] had 4 tweets
```

Constraints:

- `0 <= time, startTime, endTime <=` 10^9
- `0 <= endTime - startTime <=` 10^4
- There will be at most 10^4 calls **in total** to `recordTweet` and `getTweetCountsPerFrequency`.