MOCK TESTS BY WORKEARLY

TOPIC: PYTHON

PART 1

1) Given a list of timestamps in sequential order, return a list of lists grouped by week (7 days) using the first timestamp as the starting point

EXAMPLE:

```
ts = [
    '2019-01-01',
    '2019-01-02',
    '2019-01-08',
    '2019-02-01',
    '2019-02-02',
    '2019-02-05',
]

output = [
    ['2019-01-01', '2019-01-02'],
    ['2019-01-08'],
    ['2019-02-01', '2019-02-02'],
    ['2019-02-05'],
]
```

SOLUTION

This question sounds like it should be a SQL question doesn't it? Weekly aggregation implies a form of GROUP BY in a regular SQL or pandas question. In either case, aggregation on a dataset of this form by week would be pretty trivial.

But since it's a scripting question, it's trying to pry out if the candidate deal with unstructured data. Data scientists deal with a lot of unstructured data.
In this function we have to do a few things.

- Loop through all of the datetimes
- Set a beginning timestamp as our reference point.
- Check if the next time in the array is more than 7 days ahead.

  a. If so, set the new timestamp as the reference point.

  b. If not, continue to loop through and append the last value.

A date in Python is not a data type of its own, but we can import a module named datetime to work with dates as date objects.

To create a date, we can use the datetime() class (constructor) of the datetime module. The datetime() class requires three parameters to create a date: year, month, day.

The datetime() class also takes parameters for time and timezone (hour, minute, second, microsecond, tzone), but they are optional, and has a default value of 0, (None for timezone).

SOLUTION

```python
from datetime import datetime nts = [] #convert to datetime for testing
for t in ts:
    nts.append(datetime.strptime(t, '%Y-%m-%d'))

def group_week(ts, delim='-'):
    '''
    Groups an ordered list of timestamps as strings by week.
    The first day of the first week is defined by the earliest timestamp.
    Dates should be ordered by year, month and day

    Parameters:
    ts: str, list of timestamps
    delim: str, delimeter that separates the date
    '''
    out, week, week_ind = [], [], 0
    for i,t in enumerate(ts):
        if i == 0:
            week.append(t)
            start_date = datetime.strptime(t, f'%Y{delim}%m{delim}%d')
            continue
        t_date = datetime.strptime(t, f'%Y{delim}%m{delim}%d')
        n = (t_date - start_date).days // 7
        if n == week_ind:
            week.append(t)
        elif n > week_ind:
            week_ind = n
            out.append(week)
            week = []
            week.append(t)

        # error raising
        else:
            print('Dates do not appear to be in order.')
            return 0

    # Make sure we don't miss out last week
    if len(week) > 0:
        out.append(week)

    return out
```