Test Data Engineer

Exercises:

1. Implement the "annograms" function that uses the WORD.LST file to return anagrams of the word given in the "word" parameter.

```
def annograms(word):
    #Write your code here.
    words = [w.rstrip() for w in open('WORD.LST')]
    raise NotImplementedError

if __name__ == "__main__":
    print(annograms("train"))
    print('--')
    print(annograms('drive'))
    print('--')
    print(annograms('python'))
```

   WORD.LST: in this link.

2. Bleatrix Trotter the sheep has devised a strategy that helps her fall asleep faster. First, she picks a number N. Then she starts naming N, 2 × N, 3 × N, and so on. Whenever she names a number, she thinks about all of the digits in that number. She keeps track of which digits (0, 1, 2, 3, 4, 5, 6, 7, 8, and 9) she has seen at least once so far as part of any number she has named. Once she has seen each of the ten digits at least once, she will fall asleep.
   Bleatrix must start with N and must always name (i + 1) × N directly after i × N.

   For example, suppose that Bleatrix picks N = 1692. She would count as follows:
   - N = 1692. Now she has seen the digits 1, 2, 6, and 9.
   - 2N = 3384. Now she has seen the digits 1, 2, 3, 4, 6, 8, and 9.

- 3N = 5076. Now she has seen all ten digits, and falls asleep.

What is the last number that she will name before falling asleep? If she will count forever, print INSOMNIA instead.

INPUT

The first line of the input gives the number of test cases, T. T test cases follow. Each consists of one line with a single integer N, the number Bleatrix has chosen.

OUTPUT

For each test case, output one line containing Case #x: y, where x is the test case number (starting from 1) and y is the last number that Bleatrix will name before falling asleep, according to the rules described in the statement.

LIMITS

1 ≤ T ≤ 100.

DATASET

0 ≤ N ≤ 200.

SAMPLE

Input

```
5
0
1
2
11
1692
```

Output

```
Case #1: INSOMNIA
Case #2: 10
Case #3: 90
```

Case #4: 110

Case #5: 5076

- In Case #1, since 2 × 0 = 0, 3 × 0 = 0, and so on, Bleatrix will never see any digit other than 0, and so she will count forever and never fall asleep. Poor sheep!
- In Case #2, Bleatrix will name 1, 2, 3, 4, 5, 6, 7, 8, 9, 10. The 0 will be the last digit needed, and so she will fall asleep after 10.
- In Case #3, Bleatrix will name 2, 4, 6... and so on. She will not see the digit 9 in any number until 90, at which point she will fall asleep. By that point, she will have already seen the digits 0, 1, 2, 3, 4, 5, 6, 7, and 8, which will have appeared for the first time in the numbers 10, 10, 2, 30, 4, 50, 6, 70, and 8, respectively.
- In Case #4, Bleatrix will name 11, 22, 33, 44, 55, 66, 77, 88, 99, 110 and then fall asleep.
- Case #5 is the one described in the problem statement.

Sample input: in this link.

3. Given the following dataset: applicationdata.
   Generate a report with a tool of your choice (eg. pandas, jupyter notebooks, etc), the report must comply the following requirements :
   The summary_date must be in ISO Date format.
   The metrics must be grouped by campaign.
   The metrics that must be:
   Number of impressions.
   Number of clicks.
   Number of installs.
   Amount of spend.
   CTR (number of clicks/number of impressions)
   CPI (amount of spend/number of installs)

   Plus: if you wish, include an exploratory analysis of the dataset.

4. **Data pipeline Design**:

Design a data pipeline to create a week-over-week report using Google Analytics data. The report should provide insights into website traffic trends and key performance indicators for each week.

Data Extraction:

Describe your approach to extract data from the Google Analytics API and store it in BigQuery. Explain the steps you would take to achieve this, including any necessary authentication, data retrieval, and data loading into BigQuery. You can use any method or tool of your choice, and there is no restriction on using open-source or proprietary solutions.

Data Modeling:

Design a table to store the week-over-week report data. The table should have columns for week_start_date, sessions, pageviews, users, bounce_rate, conversion_rate, etc. The week_start_date column will be used to represent each reporting week's starting date.

Generating the Week-over-Week Report:

Use a SQL query to aggregate the data from the extracted table to calculate the week-over-week metrics. For example, calculate the percentage change in sessions, pageviews, users, etc., from the previous week.

Bonus: Using dbt for Data Modeling (Partial Implementation):
For bonus points, you can use dbt to automate the data modeling process. Create dbt models to define the data transformations required to calculate week-over-week metrics. Store the results in a separate table.

Note: This is a simplified technical test, and you are not required to implement the entire solution. Focus on the high-level design and key steps of the data pipeline, data extraction from the Google Analytics API, and data modeling process. Feel free to use any method or tool of your choice for the data extraction part.

Requirements:
The resolution of the exercises should be uploaded to GitHub and send the repository link to developers@winclap.com.