1. Import Required Libraries

python

```
import time
import pandas as pd
from pytrends.request import TrendReq
```

- time → Used to introduce delays between requests.
- pandas → Used for data processing.
- TrendReq → Used to fetch Google Trends data.

2. Initialize the Pytrends Request Object

python

```
Trending_topics = TrendReq(hl='en-US', tz=360)
```

- hl='en-US' → Specifies the language as English (US).
- tz=360 → Sets the time zone offset (360 = UTC+6).

3. Define the Keyword List and Build the Payload

python

```
kw_list = ["Cloud Computing"]
Trending_topics.build_payload(kw_list, cat=0, timeframe='today 12-m')
```

- kw_list = ["Cloud Computing"] → The keyword to search in Google Trends.
- cat=0 → No specific category (default = 0).
- timeframe='today 12-m' → Retrieves data for the last 12 months.

4. Create a Function to Handle TooManyRequestsError (429)

python

```
def get_interest_over_time(trends_obj, retries=5, delay=10):
    for attempt in range(retries):
        try:
            data = trends_obj.interest_over_time()
            return data
        except Exception as e:
            if "429" in str(e):
```

```
            print(f"Too many requests. Retrying in {delay} seconds...")
            time.sleep(delay)  # Wait before retrying
        else:
            raise e
    print("Failed after multiple retries.")
    return None
```

What this function does:

1. **Tries up to retries=5 times** to fetch Google Trends data.
2. If a **TooManyRequestsError (429) occurs**, it:
    - Prints a message: "Too many requests. Retrying in 10 seconds..."
    - Waits **10 seconds** before retrying (time.sleep(delay)).
3. If another error occurs, it **raises the exception immediately**.
4. If **all retries fail**, it prints "Failed after multiple retries." and returns None.

5. Call the Function to Fetch Data

python

```
data = get_interest_over_time(Trending_topics)
```

- Calls the get_interest_over_time() function to fetch data **with retry logic**.

6. Process the Retrieved Data

python

```
if data is not None:
    data = data.sort_values(by="Cloud Computing", ascending=False)
    data = data.head(10)
    print(data)
else:
    print("No data retrieved due to request limitations.")
```

- If data is successfully retrieved:
    1. **Sorts it in descending order** (ascending=False).
    2. **Selects the top 10 rows** (head(10)).
    3. **Prints the result**.
- If no data is retrieved, it prints:
  "No data retrieved due to request limitations."