RAM Cleaner Automation Script - Detailed Project Documentation

Overview:
This project is a Python-based RAM cleaner that monitors system memory usage in real time and clears the working set
of processes when usage exceeds a specified threshold. The logs are stored for analysis and can be viewed live in PowerShell.

Key Components:
1. Memory Monitoring:
- The script uses psutil (a Python library) to fetch system memory usage statistics.
- It checks the RAM usage percentage continuously in a loop.

2. Threshold Setting:
- The threshold (e.g., 60%) defines when the cleaning process starts.
- If usage >= threshold, the script proceeds to free up memory.

3. Clearing Memory:
- The script targets all running processes and attempts to clear their working sets using OS commands.
- This can free unused memory but might cause temporary slowdowns as processes reload their data into RAM.

4. Logging:
- Every action is recorded in ram_cleaner.log with a timestamp and memory usage percentage.
- Log format example:

```
2025-08-09 18:45:52,639 – INFO – RAM 72.7% >= 60% – clearing working set
```

5. Real-Time Log Viewing:
- In PowerShell, the following command is used to view logs in real time:

```
Get-Content ram_cleaner.log -Wait
```

- Press CTRL + C to stop viewing logs.

Positive Aspects:
+ Keeps memory usage under control automatically.
+ Logs provide transparency and help in debugging.
+ Threshold can be easily adjusted for different needs.

Negative Aspects:
- Aggressive cleaning may slow down the system temporarily.
- Some critical processes may not respond well to forced working set clearance.
- Continuous running consumes some CPU cycles.

Code Explanation (Line-by-Line):

1. Import Statements:
import psutil – For fetching system and process statistics.
import os – To run OS-level commands.
import logging – To log activities to a file.
import time – To manage delays between checks.

2. Logging Setup:

```
logging.basicConfig(filename="ram_cleaner.log", level=logging.INFO, format="%(asctime)s - %(levelname)s - %(message)s")
```

- Stores logs in ram_cleaner.log with timestamps and message formats.

3. Threshold Setting:

```
THRESHOLD = 60
```

- This defines the RAM usage percentage at which cleaning will be triggered.

4. Function: clear_memory():
- Iterates through all processes using psutil.process_iter().
- Tries to run an OS command to clear their working set (Windows-specific).
- If it fails, logs an error but continues.

5. Main Loop:

```
while True:
```

- Gets current memory usage via psutil.virtual_memory().percent.
- If usage >= THRESHOLD:
* Logs the action.
* Calls clear_memory().
- Waits 5 seconds before the next check.

6. Exit Mechanism:
- To stop the script: Press CTRL + C in the terminal where it’s running.
- To stop real-time log viewing in PowerShell: Press CTRL + C.

Usage Instructions:
1. Install dependencies:

```
pip install psutil
```

2. Run the script in Python.
3. Monitor logs using PowerShell command:

```
Get-Content ram_cleaner.log -Wait
```

4. Adjust the THRESHOLD variable as needed.

Conclusion:

This script is useful for automatically controlling RAM usage on systems where high memory usage causes performance drops. However, it should be used with caution as aggressive memory clearing can slow applications temporarily.