**Log Monitoring Code — Mistakes Fixed and Improvements Made**

**1. Regex Pattern Fixes:**

* **First version:**  
  Pattern was overly strict and incorrect in places:

python

CopyEdit

pattern = r'(\d+\.\d+\.\d+\.\d+) - (\w+) \[(\d{2}/\w{3}/\d{4}:\d{2}:\d{2}:\d{2} \+\d{4})\] "(\w+\.\w+\.\w+)" "(\w+ /.+ HTTP/\d\.\d)" (\d+) (\d+) (\d+)'

* + The domain and request groups used very restrictive regex tokens like \w+\.\w+\.\w+ and \w+ /.+ HTTP/\d\.\d, which would fail on valid inputs that contain dashes, underscores, or other characters.
  + Quotes inside regex weren't consistently handled; groups like "(...) " could fail if fields contained spaces or special chars.
* **Second version:**  
  Used a more flexible and correct pattern:

python

CopyEdit

pattern = (

r'(\d+\.\d+\.\d+\.\d+)' # IP

r' - (\w+)' # Action or user

r' \[(\d{2}/\w{3}/\d{4}:\d{2}:\d{2}:\d{2} \+\d{4})\]' # Timestamp

r' "([^"]+)"' # Domain (any string except quotes)

r' "([^"]+)"' # Request (any string except quotes)

r' (\d+)' # Status code

r' (\d+)' # Bytes sent

r' (\d+)' # Unknown field

)

* + [^"]+ inside quotes matches any string except quotes, handling more realistic log formats.
  + This pattern is more robust and matches typical nginx combined log format.

**2. Parsing Status Code as Integer:**

* **First version:**  
  Status code is returned as a string:

python

CopyEdit

'status': status

But then in is\_error\_status(status), you compare status >= 400 — comparing string with int, which causes logic errors or exceptions.

* **Second version:**  
  Converted status code to int properly:

python

CopyEdit

'status': int(status\_str)

This avoids comparison errors and makes the error status check work correctly.

**3. Improved Error Handling for Timestamp Parsing:**

* **First version:**  
  No exception handling when parsing timestamp with datetime.strptime.  
  If the timestamp format is slightly off, the program crashes or returns None without explicit handling.
* **Second version:**  
  Wrapped timestamp parsing in a try-except block:

python

CopyEdit

try:

timestamp = datetime.strptime(timestamp\_str, '%d/%b/%Y:%H:%M:%S %z')

except ValueError:

return None

This makes the parser more robust to malformed or unexpected log lines.

**4. More Pythonic and Efficient File Reading:**

* **First version:**  
  Reads entire file into a list first, then loops over lines:

python

CopyEdit

with open(log\_file, 'r') as f:

lines = f.readlines()

for line in lines:

...

This loads the entire file into memory unnecessarily.

* **Second version:**  
  Reads file line by line directly in the loop:

python

CopyEdit

with open(log\_file, 'r') as f:

for line in f:

...

This is memory efficient and better for large log files.

**5. Correct Window Size and Error Rate Calculation:**

* **First version:**
  + Variable naming unclear (window\_size without units).
  + The final alert after the loop doesn’t check if window\_requests is zero before dividing, causing a possible division by zero error.
* **Second version:**
  + Renamed to window\_size\_minutes for clarity.
  + Checked window\_requests > 0 before calculating error rate at the end:

python

CopyEdit

if window\_requests > 0:

error\_rate = window\_errors / window\_requests

if error\_rate > error\_threshold:

...

Prevents runtime errors and clarifies logic.

**6. Improved Alert Printing Format:**

* **First version:**  
  Prints error rate as a raw float (e.g., 0.1 instead of 10%).
* **Second version:**  
  Prints error rate as percentage with 2 decimal places:

python

CopyEdit

print(f"Alert! Error rate {error\_rate \* 100:.2f}% exceeds threshold at {current\_window\_start}")

This is clearer and more user-friendly.

**Summary Table of Main Fixes:**

| **Issue** | **First Version** | **Second Version (Corrected)** |
| --- | --- | --- |
| Regex pattern | Overly restrictive, brittle | Flexible, realistic parsing |
| Status code data type | String, causing comparison issues | Converted to integer |
| Timestamp parsing | No error handling | try-except for robustness |
| File reading | Read all lines at once | Iterate lines one-by-one |
| Division by zero risk | Not checked before dividing | Check for zero before dividing |
| Alert message formatting | Raw float output | Nicely formatted percentage output |