Live coding exercise

Context

Your test script outputs simple text logs of sensor readings. You need a quick utility to parse these logs, skip any malformed lines, and summarize each sensor's statistics.

Starter Log Format

The log file is plain text. Each valid line is:

```
<timestamp> | <sensor_name> | <value>
```

- `timestamp`: float seconds since start
- `sensor_name`: string without pipes
- `value`: float reading

Example (`example.log`):

```
0.10 | Temp     | 23.5
0.15 | Press    | 1.02
BAD LINE – skip me
0.20 | Temp     | 23.7
0.25 | Humidity| 45.1
```

Your Tasks

1. **Implement** `parse_log`

   ```
   from typing import List, Tuple

   def parse_log(path: str) -> List[Tuple[float, str, float]]:
   ```

```
    """
    - Read the file at `path`.
    - For each line, attempt to split on '|' into (ts, name, val).
    - Strip whitespace, convert ts and val to float.
    - If any error occurs (wrong format or conversion), print a warning and
skip that line.
    - Return a list of tuples (timestamp, sensor_name, value).
    """
    ...
```

2. **Implement** `summary_report`

```
from typing import Dict, Any

def summary_report(
    records: List[Tuple[float, str, float]]
) -> Dict[str, Dict[str, float]]:
    """
    - Input: list of (timestamp, sensor_name, value).
    - Output: dict mapping each sensor_name to a dict with:
        {
          'count': int,
          'min': float,
          'max': float,
          'avg': float
        }
    - Compute these stats per sensor.
    """
    ...
```

3. **Write** a `main()` block that:
    - Reads the filename from `sys.argv[1]`.
    - Calls `parse_log`, then `summary_report`.
    - Prints each sensor's stats in a table sorted by sensor name, e.g.:

```
Sensor    Count    Min     Max     Avg
-------------------------------------
Humidity  1        45.10   45.10   45.10
Press     1        1.02    1.02    1.02
Temp      2        23.50   23.70   23.60
```

Stretch Goal (if time permits)

Add an optional command-line argument `-threshold N`.

After printing the table, also print:

```
ALERT: Temp average 23.60 exceeds threshold 23.0
```

for any sensor whose average > threshold.