1. Visual format

Progress lines should always look like a **single status line** that gets updated in place, for example:

```
[2025-11-16 18:25:24] [PROGRESS] 219 kept 119  skip 338   1.9/s
```

Core pieces:

- Timestamp in square brackets
- Label `PROGRESS` in the same label column as the other log levels.
- Short status message only:
    - current kept count
    - current skip count
    - total processed so far
    - Total processed
    - Jobs per second

Nothing else (no JSON blobs, no long URLs) should appear in this line. If you print something long here, it will either wrap or overflow, ruining the “single-line” behavior.

---

2. One-line only, no wrapping

Progress lines must **never go through the word-wrapping helper** we use for DEBUG/INFO/WARN/DONE. Instead, they should:

- Be formatted into one string
- Be printed with `print()` (or your `log_print`) **without** calling `_bk_log_wrap` / `textwrap.fill`
- Stay comfortably within your 120-character width so that they do not wrap, even if the terminal automatically wraps long lines.

So when you change wrapping settings for DEBUG/INFO/etc, leave the progress helper alone. It should not look at `_LOG_WRAP_WIDTH` or break long words.

---

3. Overwrite-in-place behavior

The progress system assumes:

- There is at most **one active progress line** at a time
- Every progress update does something like:
    - `\r` to return to column 0
    - print the new progress line
- Before **any other** "normal" log (KEEP, SKIP, DEBUG, REASON, DONE) prints, a helper such as `progress_clear_if_needed()` runs to:
    - overwrite the progress line with spaces
    - move the cursor to a clean new line

That is why you see calls like:

```
progress_clear_if_needed()
debug("ld_tag content (first 300 chars): ...")
```

If you ever print a non-progress log without clearing first, the spinner/counters will get stuck in the middle of other output.

---

4. Frequency control

To avoid flooding the terminal:

- A progress update should only be printed if:
    - Either a minimum time interval has passed since the last progress line (for example, 0.2–0.5 seconds), **or**
    - You are at the very end of the run and need to print the final line

So the "requirements" here:

- Track `last_progress_ts`

- Only print when `now - last_progress_ts >= MIN_INTERVAL` or on final flush

---

5. Data it depends on

For each progress update, the function expects:

- `kept_count` so far
- `skip_count` so far
- some notion of `total_processed = kept_count + skip_count`
- `start_ts` (to compute jobs per second)

If any of those are missing or `None`, you will get incorrect numbers or formatting errors. Whenever you add new filters or change where SKIPs occur, ensure that you still increment the same global counters that feed the progress line.

---

6. Interaction with your other changes

Given all that, when you tweak wrapping or add fields:

- It is safe to change `_bk_log_wrap` for DEBUG/INFO/WARN/KEEP/SKIP/REASON lines
- Do **not** send the PROGRESS line through `_bk_log_wrap`
- Keep the PROGRESS message short and number-only, no big JSON or long URLs

~~If you want, next step I can sketch the small `update_progress(...)` helper we expect, so you can compare it to the one in your file and make sure it still matches these rules.~~