

■ Assignment: 2024 Movieland Personal Income Tax Calculator

Overview

Build a command-line tool that calculates the 2024 federal personal income tax for the fictional country of Movieland. This isn't just a bracket-sum calculator — you'll need to:

- 1 Derive tax brackets from a formula, not hardcode them.
- 2 Handle monthly earnings and year-to-date rules.
- 3 Apply nuanced rounding and conditional tax credits.
- 4 Output a structured JSON report.

1. Problem Statement

Run this command:

```
python movieland_tax.py income.csv panels.csv > result.json
```

Input Files:

1. income.csv (no header; 12 rows, one per month)

```
month,gross_income
2024-01,5800.00
2024-02,6200.00
...
2024-12,5400.00
```

2. panels.csv (no header; one row per solar panel purchase)

```
panel_id,purchase_date
A-123,2024-03-17
B-041,2024-07-02
...
```

Expected JSON Output (illustrative values):

```
{ "taxable_income": 67540.00, "total_tax": 11832.14, "effective_rate": 0.1751, "marginal_rate":
0.22, "breakdown": [ { "bracket": 1, "taxable": 2375.00, "rate": 0.07, "tax": 166.25 } ],
"credits": { "basic": 600, "green_bonus": 100, "net_tax": 11132.14 } }
```

2. Functional Requirements

- 1 Bracket calculation: Derive 2024 thresholds using a formula from the 2023 table.
- 2 Monthly processing: Tax each month until YTD crosses the top bracket, then recompute total.
- 3 Credits: Fixed credit plus $\$50 \times \text{panel count}$ (capped at 20) if taxable income $< \$90,000$.
- 4 Rounding: Round each tax value up to the nearest \$0.07.
- 5 CLI behavior: Clear errors for malformed CSVs, missing files, or invalid values.
- 6 Performance: Complete processing in < 2 seconds, no internet access allowed.

3. Non-Functional Constraints

- 1 Single entry file: movieland_tax.py (helpers allowed).

- 2 Standard library only: csv, json, decimal, math, itertools, dataclasses, typing, pathlib, argparse, datetime.
- 3 Must run offline. No pip installs or APIs.

4. Deliverables

File	Purpose
movieland_tax.py	The main implementation
README.md	$\leq \frac{3}{4}$ page: usage, assumptions, and approach
requirements.txt	Specify 'standard library only' or your Python version
tests/ (optional)	Unit tests (strongly encouraged)

5. Submission Tips

- 1 Start with a small CSV and manually verify tax logic.
- 2 A helper like `ceil_to_nearest(value, 0.07)` will help you get rounding right.
- 3 Write tests for your tax logic. Rounding bugs are hard to eyeball.
- 4 Clean code + clear thinking > feature overload. Keep it focused.

Appendix: Tax Rules (Previously in Personal_Tax_Rules_2024.md)

Bracket Threshold Formula

Use this formula to compute 2024 thresholds from 2023:

$$2024 \text{ threshold} = \text{ceil}((2023 \text{ threshold} \times 1.021) - 37)$$

2023 Brackets (for reference):

Bracket	Rate	Upper Bound (2023)
1	7%	\$2,375
2	11%	\$8,950
3	16%	\$23,400
4	22%	\$55,800
5	29%	∞

Rounding Rule

Every tax value must be rounded UP to the nearest \$0.07.

Examples:

- \$81.03 → \$81.06
- \$99.99 → \$100.01

Credits

- Basic credit: \$600 (fixed)
- Green Bonus: \$50 × number of solar panels (up to 20), only if taxable income < \$90,000

Monthly Taxation Rule

Each month should be taxed independently until total income exceeds the top bracket. After that, recompute tax from scratch using YTD.