DISCUSSION POINTS — Lesson 2

Authorial Thinking • NumPy (model) + Pygame (view)

1) Data representations: Scalar · Vector · Matrix

- Scalar — a single number (e.g., TILE_SIZE=32, FPS=60).
- Vector — an ordered RGB triple (a color).
- Matrix — screen_matrix[y, x] ∈ {0..5}: palette indices.
- Pipeline: (y, x) → index → RGB → tile on screen.

2) Model ↔ View: NumPy (data) and Pygame (rendering)

- Model: screen_matrix and palette (NumPy) are the single source of truth.
- View: Pygame reads the model and draws; order matters: tiles → grid.
- Rendering never mutates the model; changes are done via NumPy slices/masks.

3) Matrix traversal and a uniform rule

- Row-major traversal: y (row) first, then x (column).
- Uniform rule: (y, x) ↦ (x*TILE_SIZE, y*TILE_SIZE, TILE_SIZE, TILE_SIZE).
- One formula removes axis confusion and eases testing.

4) Intentional inefficiency (and estimating complexity)

- Full redraw of 16×32 cells per frame: ~O(TILE_Y*TILE_X).
- Pedagogically transparent: Model → View is visible head-on.
- Show upgrades later: dirty rects; prebuilt tile blits; surfarray.

5) Authorial Thinking (position + responsibility + next step)

- The student formulates their own rules (slices/masks) → feels like an author.
- Responsibility: understand rule consequences and stand by the result.
- Forward motion: record a version and make the next deliberate step.

Mini-rubric (in class):

- Originality of the rule: what here is mine?
- Reproducibility: can someone rebuild it from my description?
- Next step: what will I change next and why?

Prepared for class • August 24, 2025