PHASE 1 — Learn to Control the Screen (Foundation)

Step 1: Master a simple redraw loop
Goal: Understand the “game loop” idea.
- Your program clears the terminal
- Draws something simple (like a number or “Hello”)
- Waits for a key
- Updates something
- Redraws from scratch
When you’re comfortable constantly repainting the screen, you're ready for the editor.

Step 2: Learn to place the visible cursor
Goal: Separate logical cursor from terminal cursor.
- Keep track of a (row, col) in your program
- After drawing, move the terminal cursor to that coordinate
You now control where the cursor goes.

PHASE 2 — Build Your Editor’s Skeleton (Core Concepts)

Step 3: Create a buffer
Goal: Store text in a simple structure.
- Represent the text as a list of lines (strings)
- Start with one empty line
This is the editor's model.

Step 4: Add a logical cursor
Goal: Make editing possible.
Track:
- Current line index
- Current column index

Step 5: Draw the full buffer on screen
Goal: Connect data to display.
- Clear screen each frame
- Draw each line of the buffer
- Place the visible cursor
This creates a basic viewer.

PHASE 3 — Add Movement and Text Editing (Turning Into an Editor)

Step 6: Handle arrow keys
Goal: Move the logical cursor safely.
- Up/Down: change line number
- Left/Right: change column number
- Clamp values to stay in bounds

Step 7: Handle character insertion
Goal: Make typing modify text.
- Insert character at cursor position
- Move cursor right

Step 8: Handle backspace

Goal: Realistic deletion.
Cases:
1. Cursor not at start → delete character
2. Cursor at column 0 → merge lines

Step 9: Handle Enter key
Goal: Create new lines.
- Split line at cursor
- Insert new line below
- Move cursor to new line start

PHASE 4 — Make It Look Like a Real Editor (Quality-of-Life)

Step 10: Add a status bar
Goal: Provide user info.
Show:
- Current mode
- Cursor position
- File name
- Shortcuts

Step 11: Allow scrolling
Goal: Support long files.
- Shift visible window when cursor moves out of view

Step 12: Open and save files
Goal: Practical usage.
- Read file into buffer
- Write buffer to disk

PHASE 5 — Optional Advanced Features (Deep Learning)

Step 13: Undo/redo system
- Store past states

Step 14: Syntax highlighting
- Highlight based on patterns

Step 15: Better text structures
- Gap buffer
- Rope
- Piece table

Step 16: Multiple views and modes
- Vim-like modes
- Split screens
- Search/replace dialog