

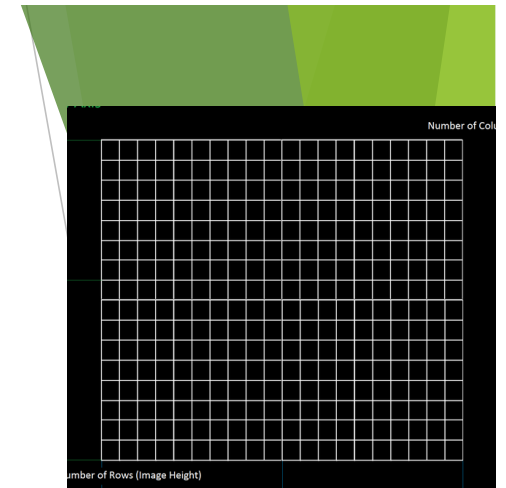
The background features abstract green geometric shapes. On the left is a tall, narrow green triangle pointing downwards. On the right is a complex, multi-layered green shape composed of several overlapping triangles and polygons in various shades of green. A thin, light gray line extends from the bottom left towards the right, passing behind the green shapes.

# Lab 07

## Rasterization -Line

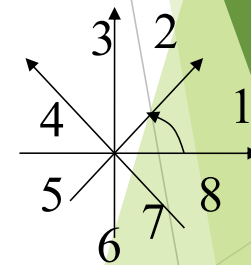
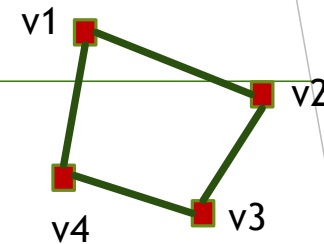
# Continue with the Previous Lab - 2D grid

- ▶ Clickable 2D Grid
  - ▶ Provide a popup menu to select the grid dimensions: (10 or 15 or 20 etc...)
  - ▶ Draw a 2D grid based on the selected dimension.
    - ▶ The default is 10  $\rightarrow$  x: (-10 ~ 10), y: (-10 ~ 10)
    - ▶ The origin (0,0) is at center
    - ▶ When the user select 15, the grid will be re-drawn to: x: (-15 ~ 15), y: (-15 ~ 15)
- ▶ When the user click on one of the cell
  - ▶ draw/fill the cell
    - ▶ You will need to implement a function to convert coordinates
    - ▶ Print out the coordinate (x, y) of this cell on the console window



# Midpoint algorithm

- ▶ Select endpoints (for example: v1, v2, v3, v4)
  - ▶ Connect line between each two endpoints: v1v2, v2v3, v3v4 and v4v1
- ▶ Use midpoint algorithm to draw the pixels along the line
  - ▶ Draw and print out all the pixels represent the line
  - ▶ Print out the coordinate (x, y) OF EACH PIXELS
  - ▶ Print out which region it belong to (e.g.: region1 or region2, etc)
    - ▶ E.g., Line V1v2: region 8
- ▶ Color:
  - ▶ Endpoints: **Red**
  - ▶ Pixel of E (east): **Green**
  - ▶ Pixel of NE (North east): **Blue**
- ▶ All regions (80%)
  - ▶ Considering all regions (First 2 region for 30% each, the rest regions total 20%)
- ▶ Pop-up Menu (10%)
  - ▶ Add one more option: normal/debug mode
  - ▶ Only print out the coordinate of the pixel where you click on



# Requirement

- ▶ Do not use other libraries. Only OpenGL API (gl, glu, glut) is allowed
- ▶ Write comments in your code
- ▶ Turn in your code and demo video

