ΗΥ416
ΓΡΑΦΙΚΑ ΥΠΟΛΟΓΙΣΤΩΝ

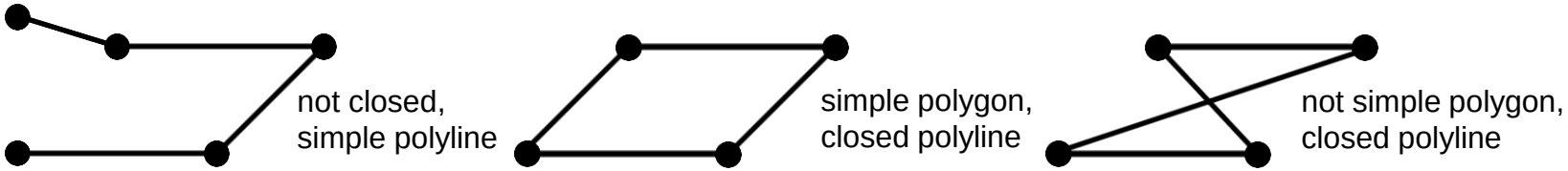
Πολύγωνα
Περιοχές Γεμίσματος Πολυγώνων

Π. ΤΣΟΜΠΑΝΟΠΟΥΛΟΥ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

2D Objects

2D Object Definition

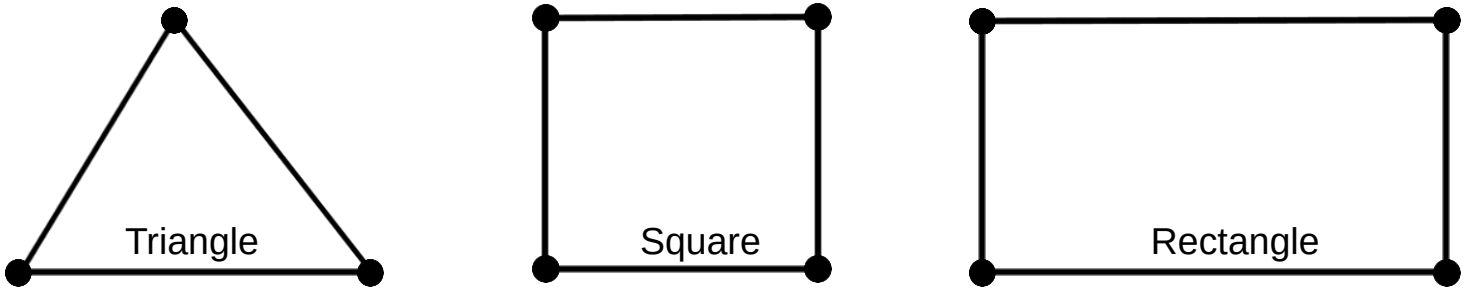
- Lines and polylines:
  - Polylines: lines drawn between ordered points
  - A closed polyline is a polygon, a simple polygon has no self-intersections



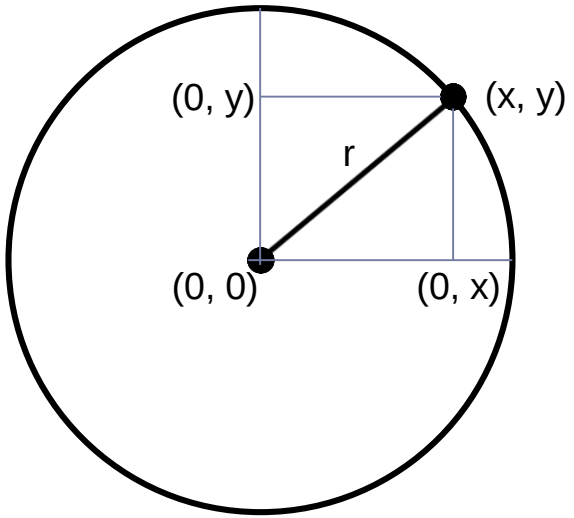
- Convex and concave polygons:
  - Convex: For every pair of points inside the polygon, the line between them is entirely inside the polygon.
  - Concave: For some pair of points inside the polygon, the line between them is not entirely inside the polygon. Not Convex.

2D Object Definition

- Special Polygons:

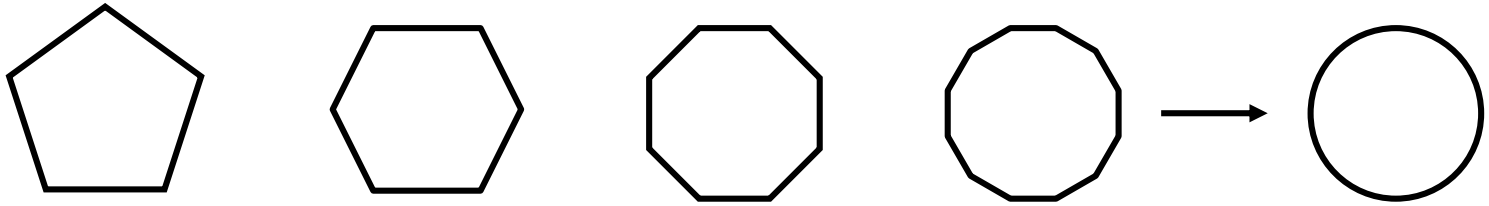


- Circles:
  - Set of all points equidistant from one point called the center
  - The distance from the center is the radius *r*
  - The equation for a circle centered at (0, 0) is $r^2 = x^2 + y^2$

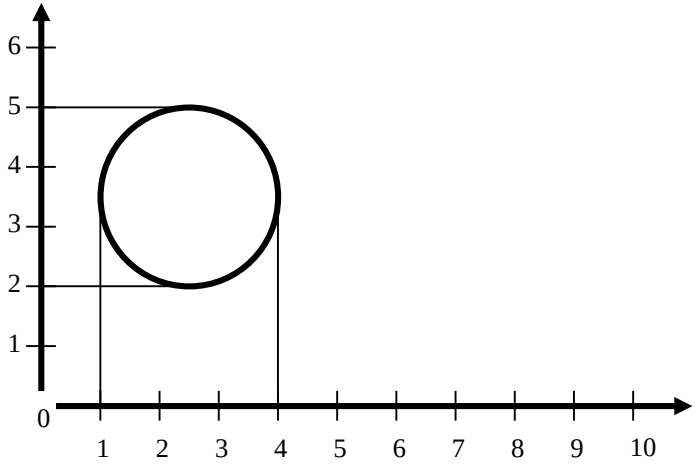


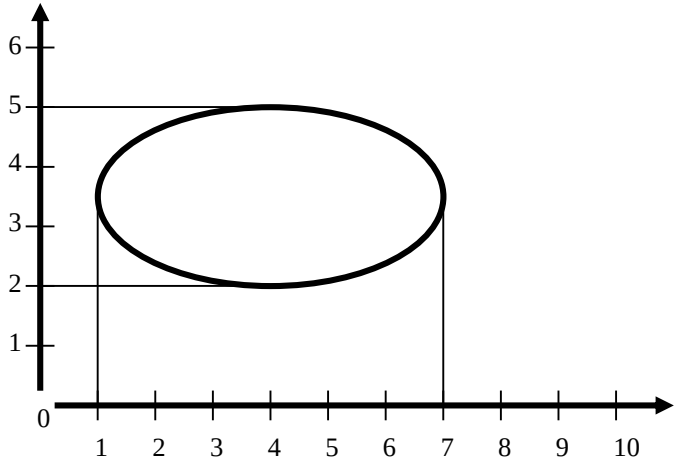
2D Object Definition

- A circle can be approximated by a polygon with many sides.



- Axis aligned ellipse: a circle scaled in the x and/or y direction





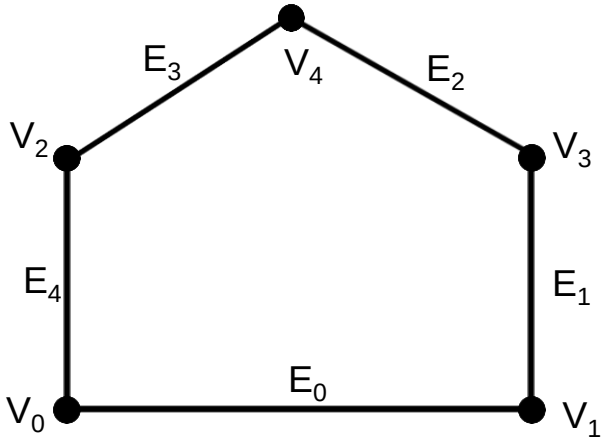
Scaled by a factor of 2 in the x direction and not scaled in the y direction. Width changes from 3 to 6.

Representing Shape

- Vertex and edge tables:
  - General purpose, minimal overhead, reasonably efficient
  - Each vertex listed once
  - Each edge is an ordered pair of indices to the vertex list (like triangles in WPF)

Vertices	
0	(0, 0)
1	(2, 0)
2	(0, 1)
3	(2, 1)
4	(1, 1.5)

Edges	
0	(0, 1)
1	(1, 3)
2	(3, 4)
3	(4, 2)
4	(2, 0)



  - Sufficient to draw shape and perform simple operations (transforms, point inside/outside)
  - Edges listed in counterclockwise order by convention

Polygons

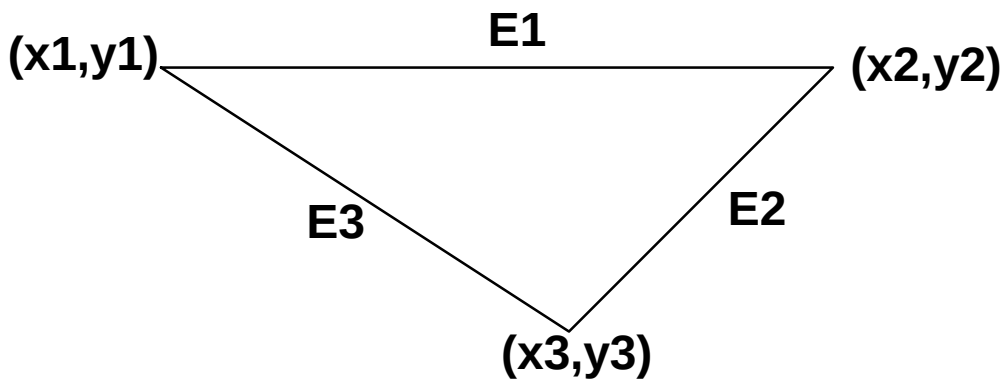
Polygons

A **polygon** is a many-sided **planar** figure composed of **vertices** and **edges**.

Vertices are represented by points (x,y).

Edges are represented as line segments which connect two points, (x1,y1) and (x2,y2).

$$P = \{ (x_i , y_i),\ i=1,n \}$$

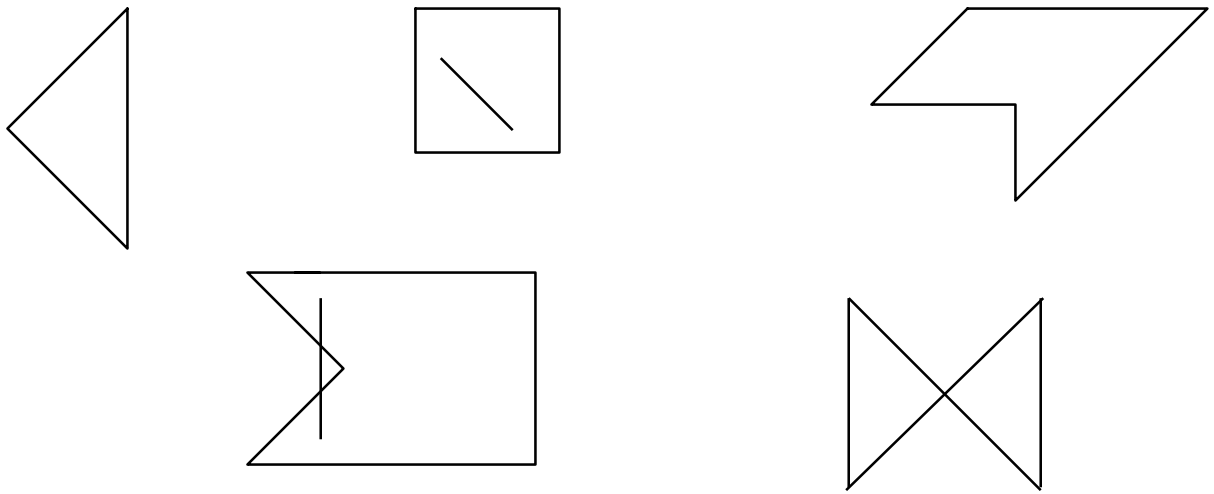


Convex and Concave Polygons

Convex Polygon - For any two points P_1, P_2 inside the polygon, all points on the line segment which connects P_1 and P_2 are inside the polygon.

- All points $P = uP_1 + (1-u)P_2$, u in [0,1] are inside the polygon provided that P_1 and P_2 are inside the polygon.

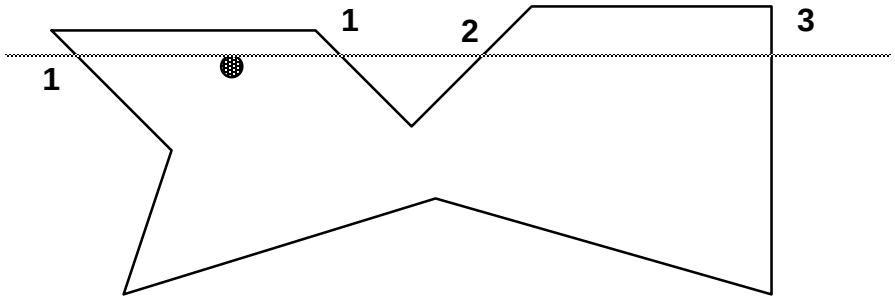
Concave Polygon - A polygon which is not convex.



Inside Polygon Test

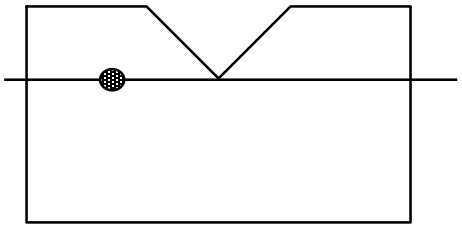
How do we know a point is "inside" a polygon?

Inside test: A point P is inside a polygon if and only if a scanline intersects the polygon edges an odd number of times moving from P in either direction.

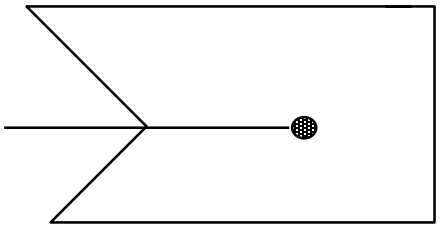


Problem when scan line crosses a vertex:

Does the vertex count as two points?

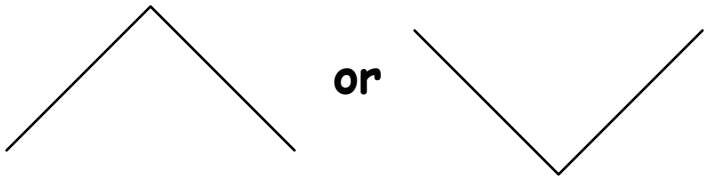


Or should it count as one point?

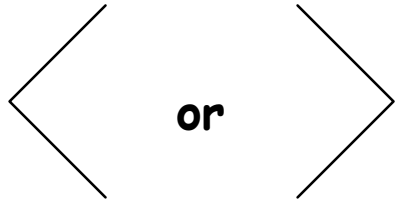


Max-Min Test

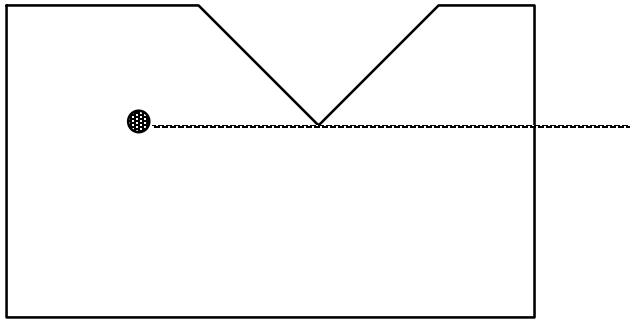
When crossing a vertex, if the vertex is a local maximum or minimum then count it twice, else count it once.

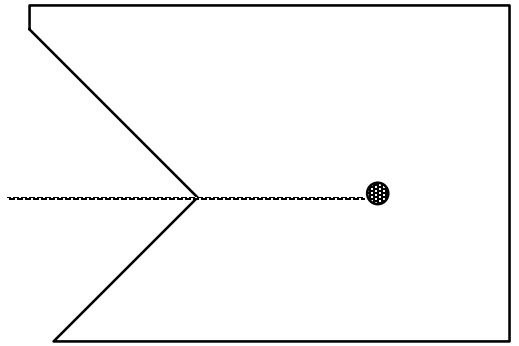


Count twice



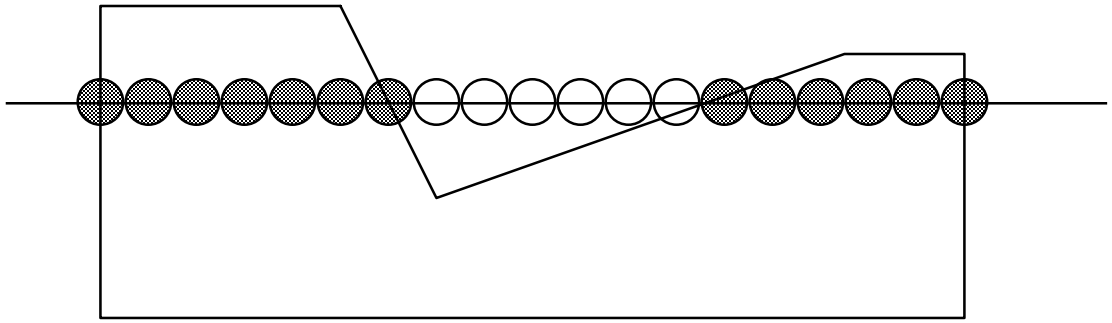
Count once





Filling Polygons

Fill the polygon 1 scanline at a time



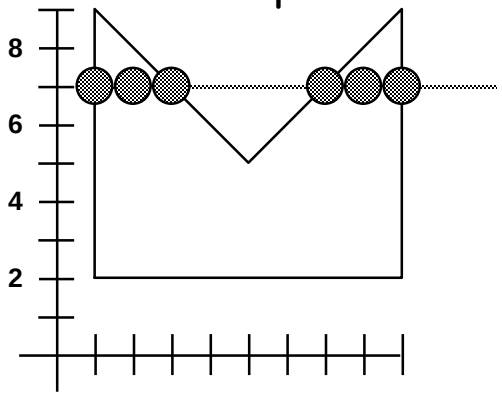
Determine which pixels on each scanline are inside the polygon and set those pixels to the appropriate value.

Look only for those pixels at which changes occur.

Scan-Line Algorithm

For each scan-line:

1. Find the intersections of the scan line with all edges of the polygon.
2. Sort the intersections by increasing x-coordinate.
3. Fill in all pixels between pairs of intersections.



For scan-line number 7 the sorted list of x-coordinates is (1,3,7,9)

Therefore fill pixels with x-coordinates 1-3 and 7-9.

Possible Problems

1. Horizontal edges → Ignore
2. Vertices → If local max or min, then count twice, else count once.
   (This is implemented by shortening on edge by one pixel.)
3. Calculating intersections is slow.

Edge Coherence

Not all edges intersect each scanline.

Many edges intersected by scanline i will also be intersected by scanline i+l

Formula for scanline s is y = s, for an edge is y = mx + b

Their intersection is

$s = mx_s + b \rightarrow x_s = (s-b)/m$

For scanline s + 1,

$x_{s+1} = (s+1 - b)/m = x_s + 1/m$

$$\mathbf{x_{s+1} = x_s + 1/m}$$

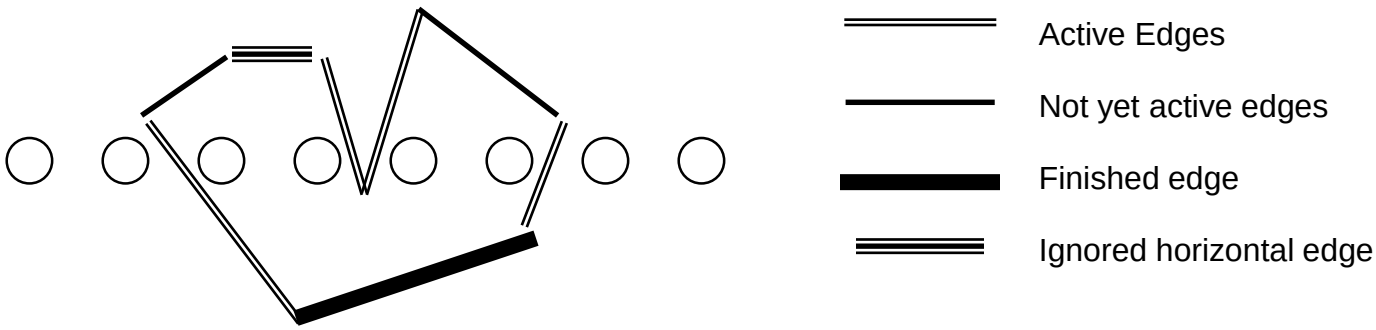
Processing Polygons

Polygon edges are sorted according to their minimum Y.

Scan lines are processed in increasing (decreasing) Y order.

When the current scan line reaches the lower endpoint of an edge it becomes active.

When the current scan line moves above the upper endpoint, the edge becomes inactive.



Active edges are sorted according to increasing X.

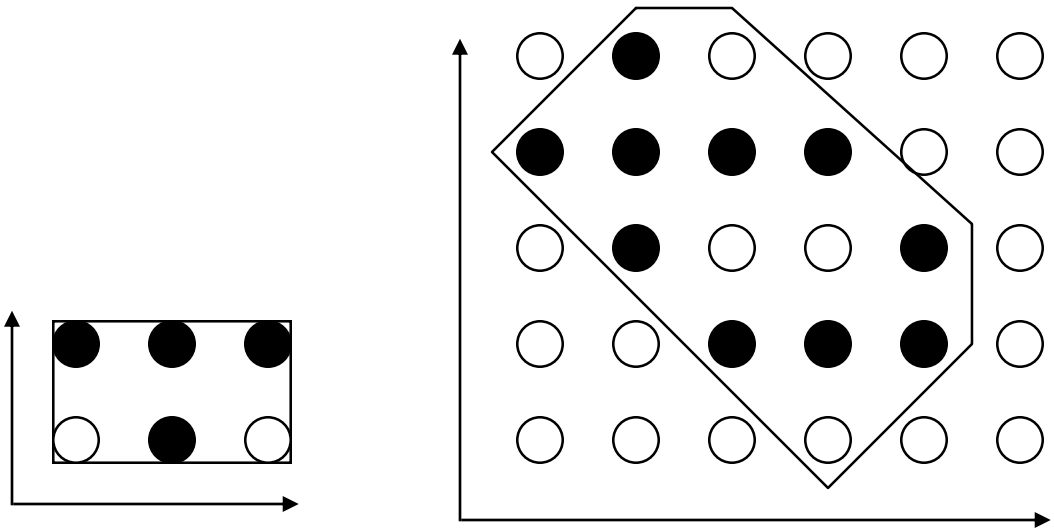
Filling the scan line starts at the leftmost edge intersection and stops at the second.

It restarts at the third intersection and stops at the fourth. . .

Fill Patterns

Fill patterns can be used to put a noticable texture inside a polygon. A fill pattern can be defined in a 0-based, *m* x *n* array. A pixel (*x*,*y*) is assigned the value found in:

pattern((x mod m), (y mod n))



Pattern **Pattern filled polygon**

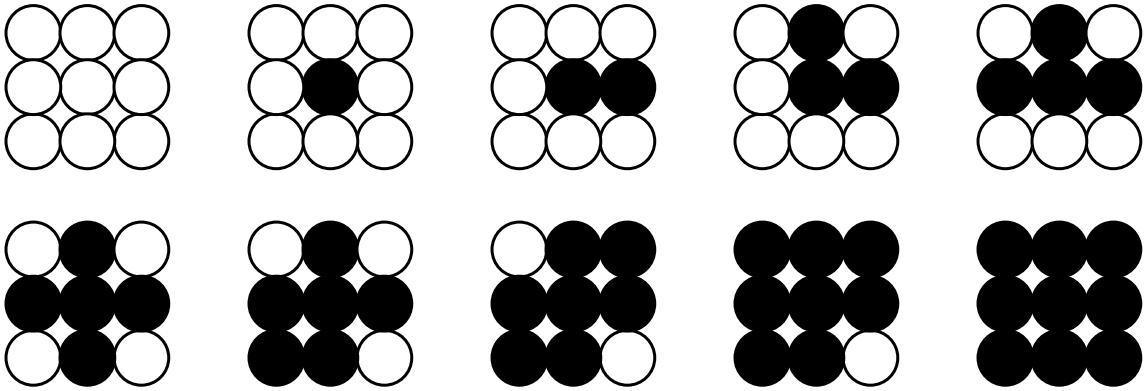
Halftoning (ενδιάμεσοι τόνοι χρώματος)

For bitmapped displays, fill patterns with different fill densities can be used to vary the range of intensities of a polygon.

The result is a tradeoff of resolution (addressability) for a greater range of intensities and is called *halftoning*.

The pattern in this case should be designed to *avoid* being noticed.

These fill patterns are chosen to minimize banding (ζώνη χρώματος).



Dithering (χρωματική αντιπαράθεση – διάχυση χρώματος)

Another method to increasing the number of apparent intensities on a bit-mapped display is *dithering*.

In an ordered dither the decision to turn a pixel on or off at point (x,y) depends on the desired intensity I(x,y) at that point and on an (n by n) dither matrix D_n.

The dither matrix is indexed from 0 to (n-1) along its rows and columns. Each of the integers 0 to n^2 -1 appears once in the matrix. For instance when n = 4, we have D_4 =

0	8	2	10
12	4	14	6
3	11	1	9
15	7	13	5

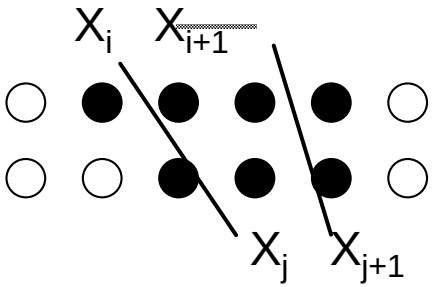
To process the point at (x,y), we first compute

i = x MOD 4, j = y MOD 4

Then if I(x,y) > D_4(i, j) the point (x,y) is turned on; otherwise it is not.

Antialiasing Polygons

Polygon edges suffer from aliasing just as lines do. If a line passes between two pixels, they share the intensity. The same method can be used on the scan line fill.

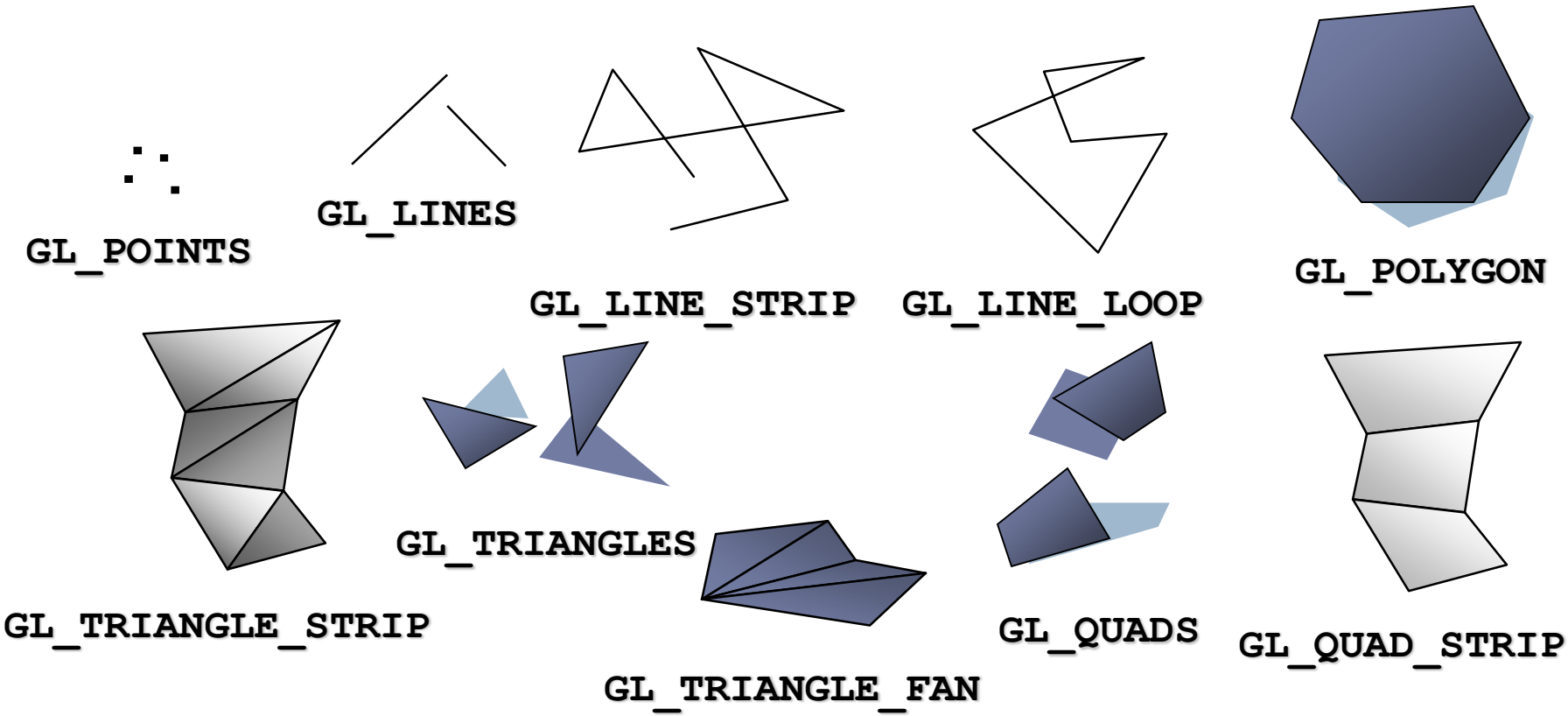


The fill begins at the leftmost edge intersection. If the intersection is between two pixels $X_i < X < X_{i+1}$ then pixel X_i is assigned the intensity $(X_{i+1} - X)$. Pixel X_i is assigned intensity 1.0 (unless the polygon is very narrow).

At the second intersection, where filling stops, the reverse is true. $X_j < X < X_{j+1}$ Pixel X_j is assigned intensity 1.0 and X_{j+1} is assigned $(X - X_j)$.

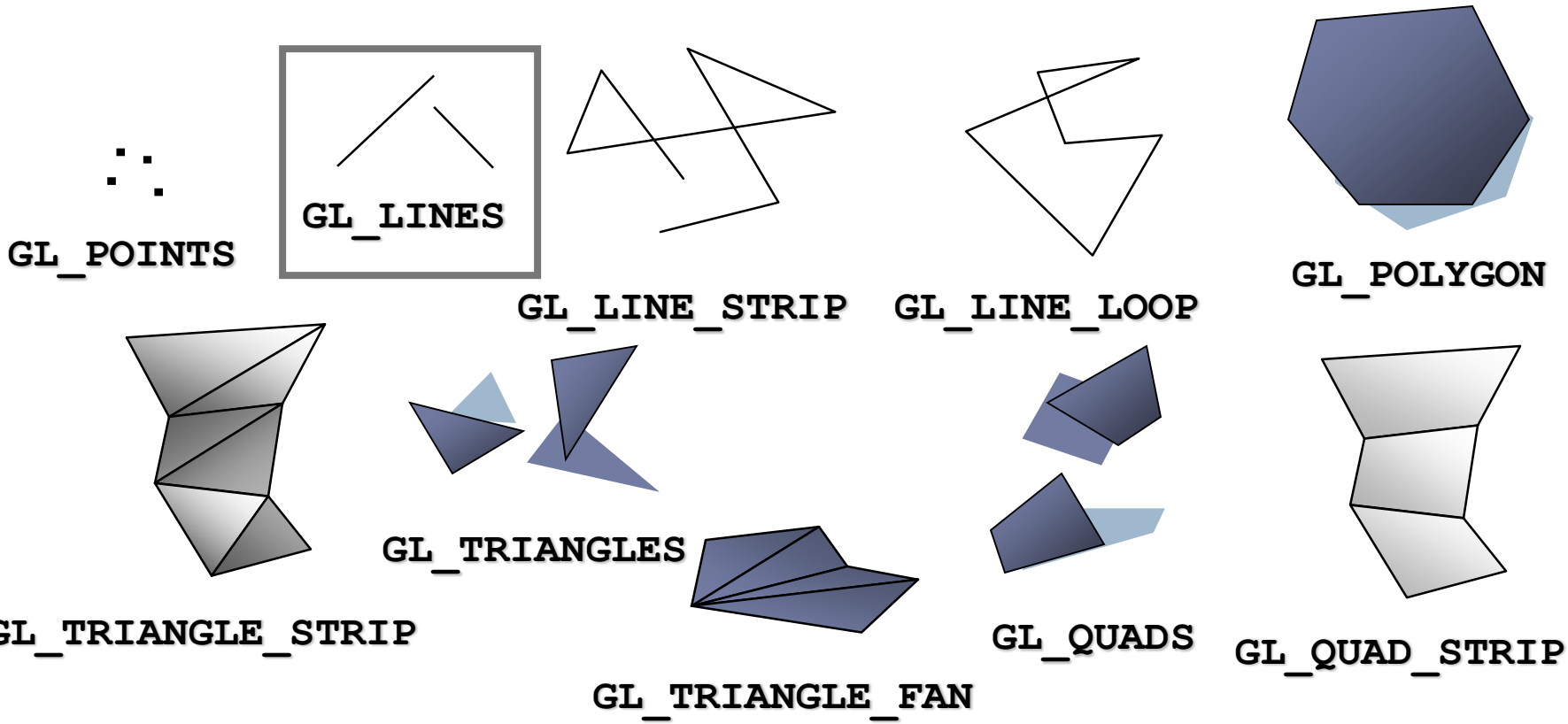
OpenGL Geometric Primitives

- All geometric primitives are specified by vertices



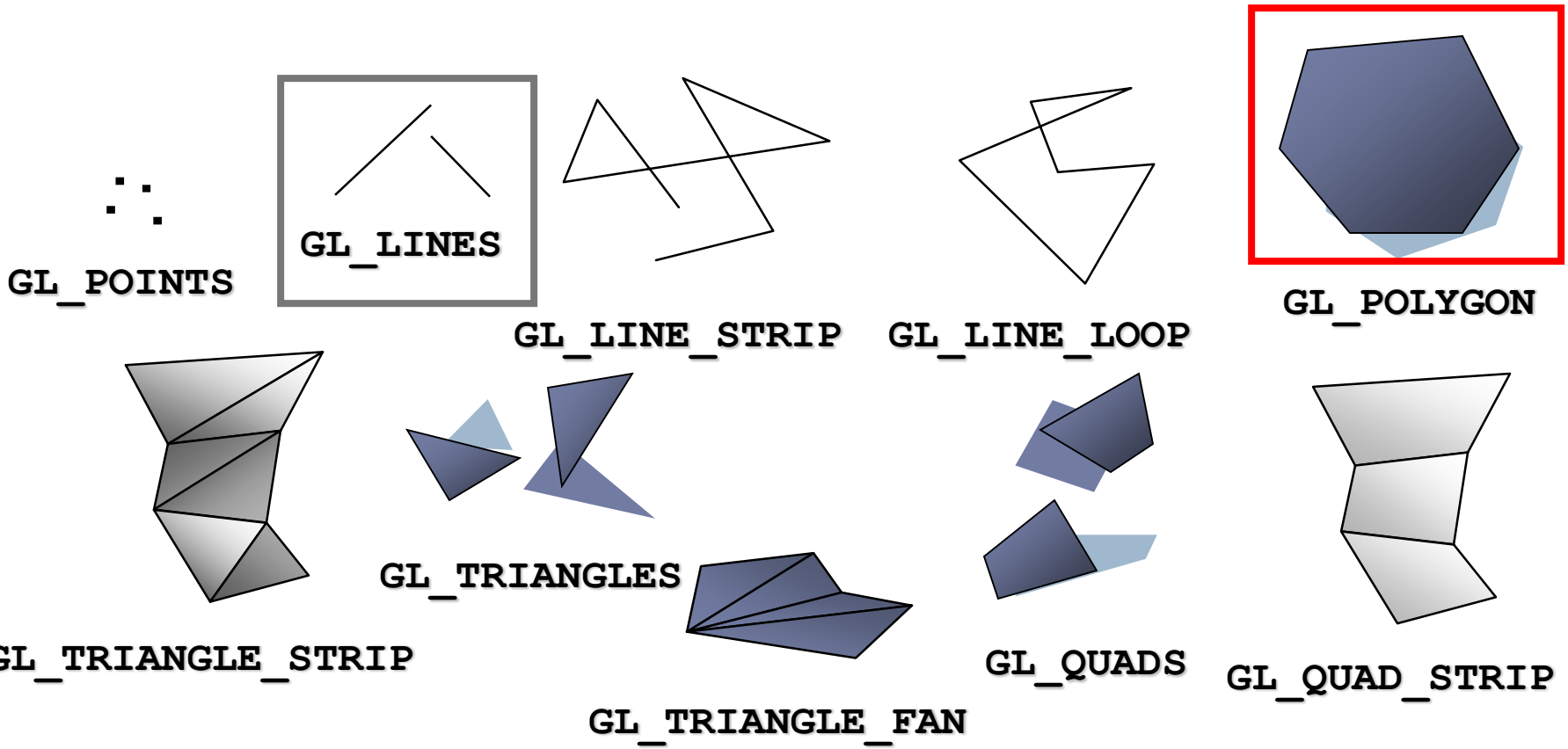
OpenGL Geometric Primitives

- All geometric primitives are specified by vertices



OpenGL Geometric Primitives

- All geometric primitives are specified by vertices



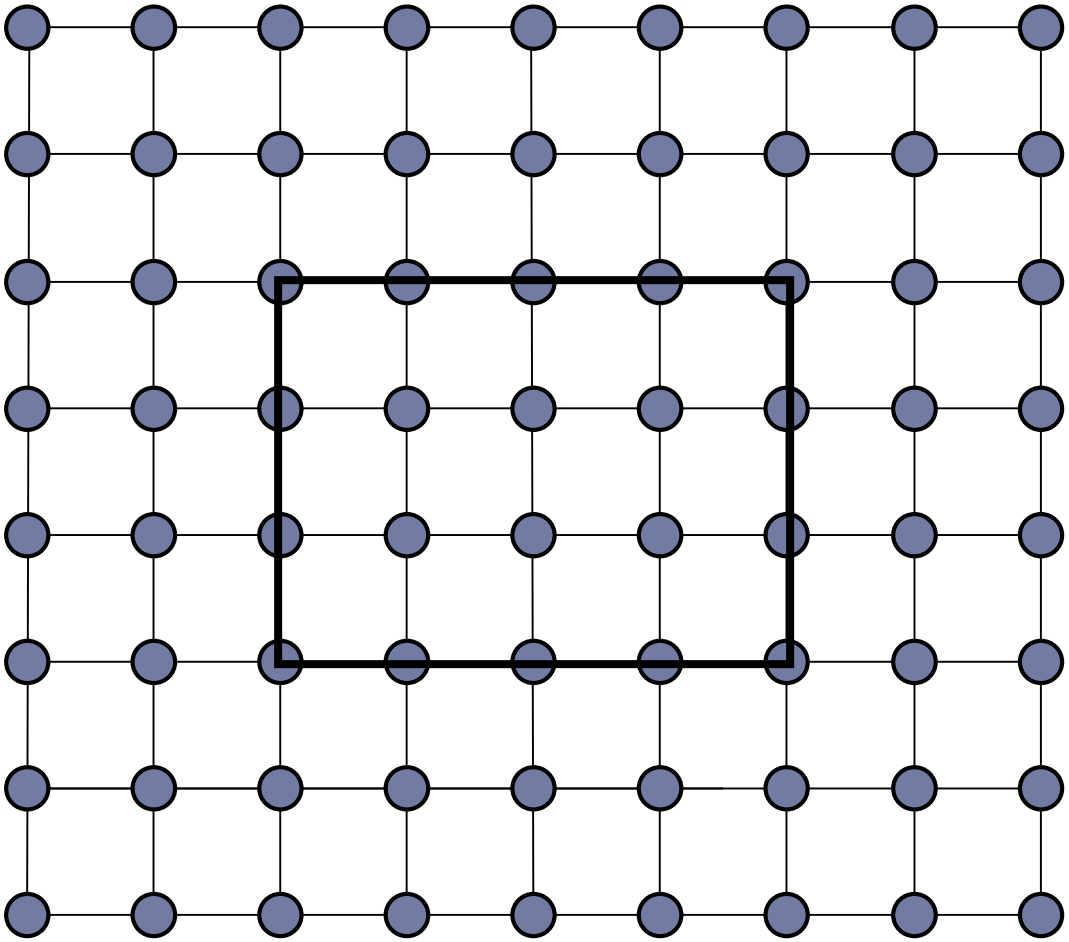
Outline

- Methods for drawing polygons or curved objects
  - Scanline conversion of polygons
  - Boundary-fill
  - Flood-fill
- Required readings:

  - HB 4-9 to 4-14

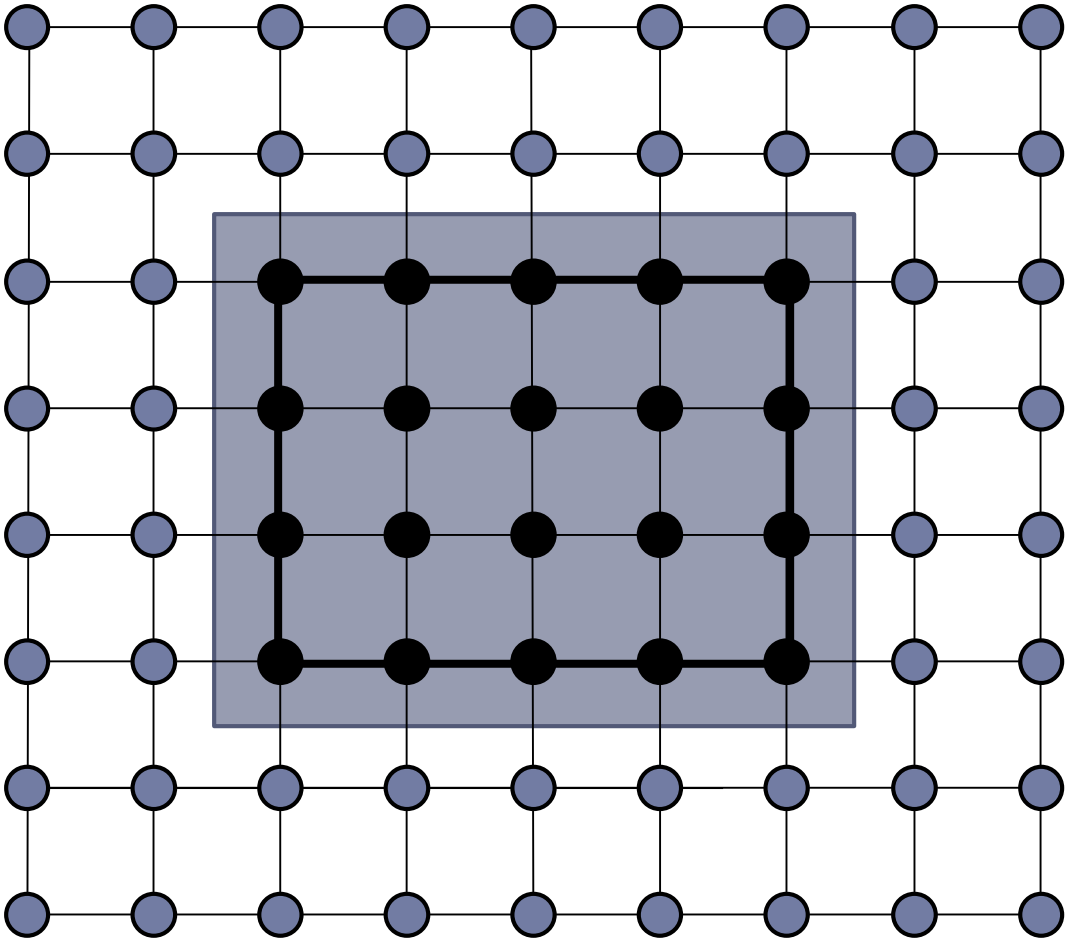
Drawing Rectangles

Which pixels should be filled?



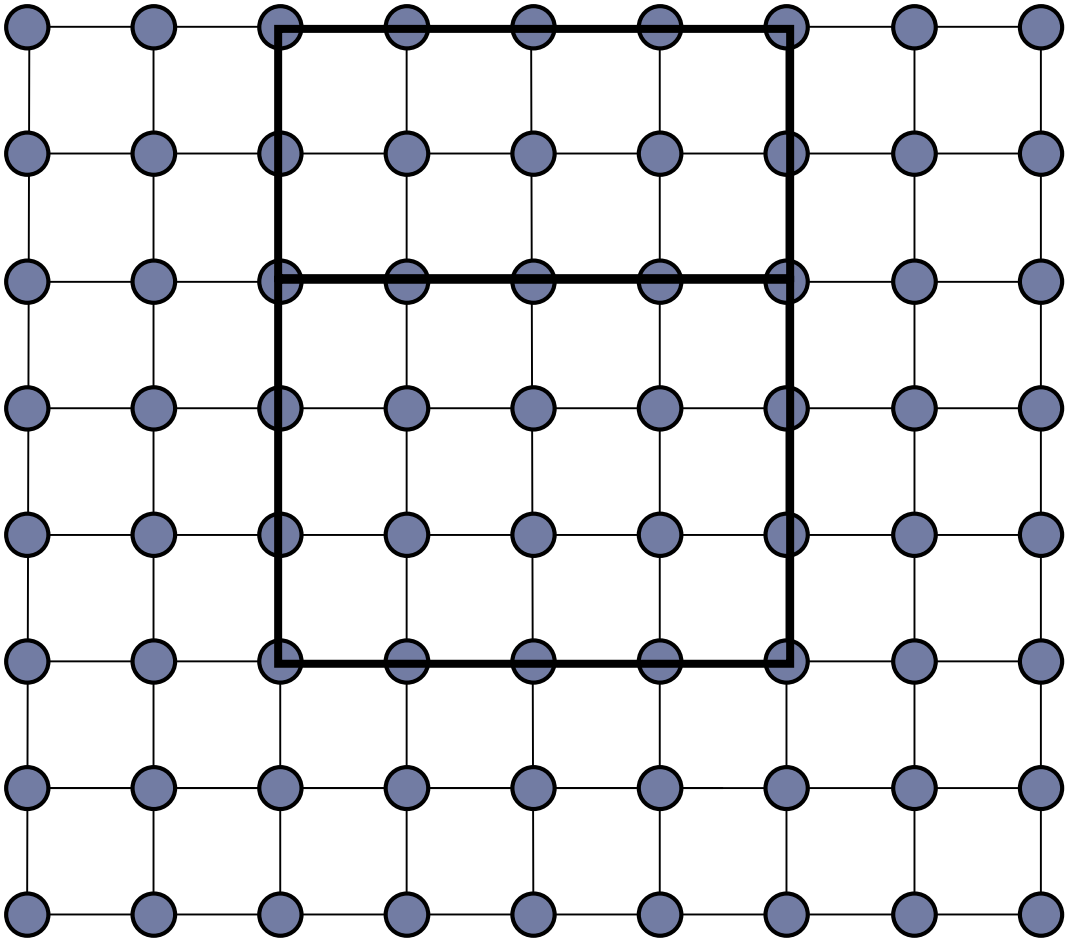
Drawing Rectangles

Is this correct?



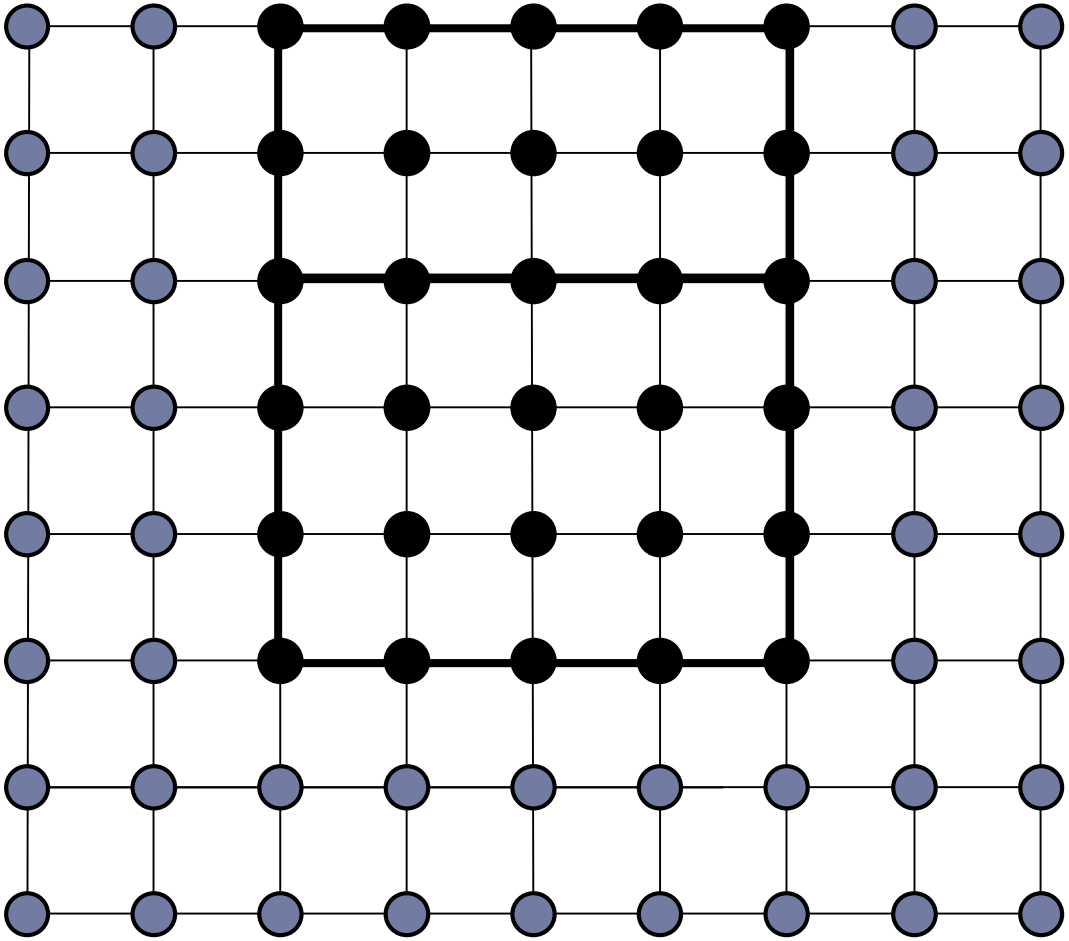
Drawing Rectangles

What if two rectangles overlap?



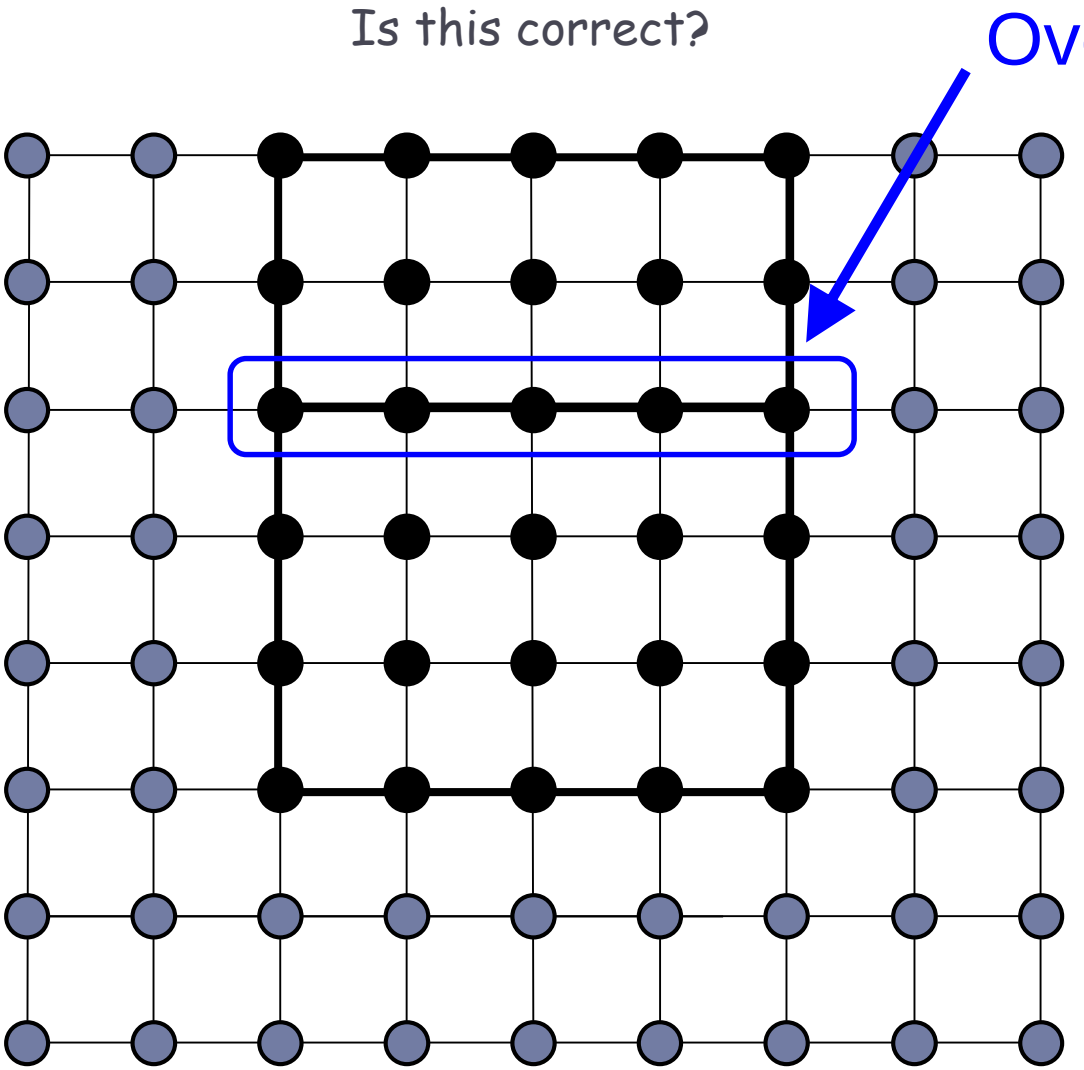
Drawing Rectangles

Is this correct?



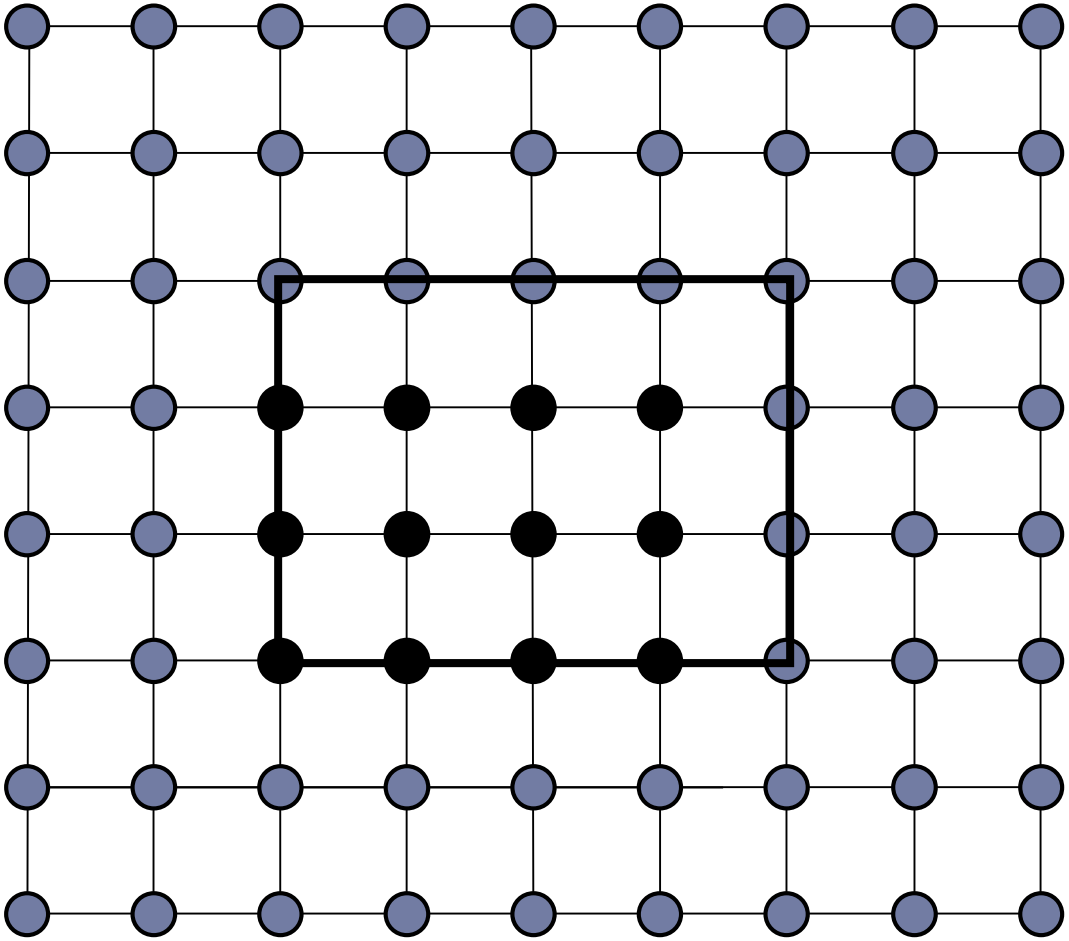
Drawing Rectangles

Is this correct?



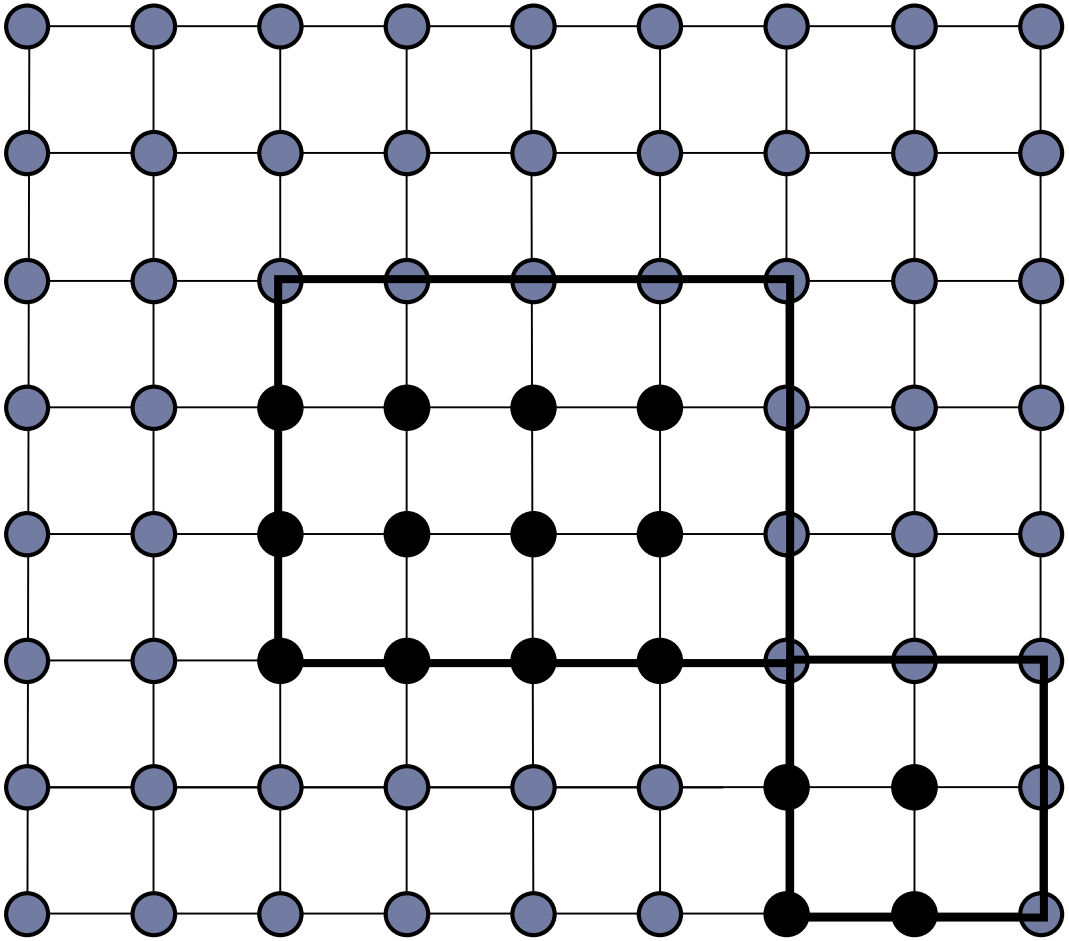
Drawing Rectangles

Solution: Exclude pixels on top and right



Drawing Rectangles

Artifacts are possible

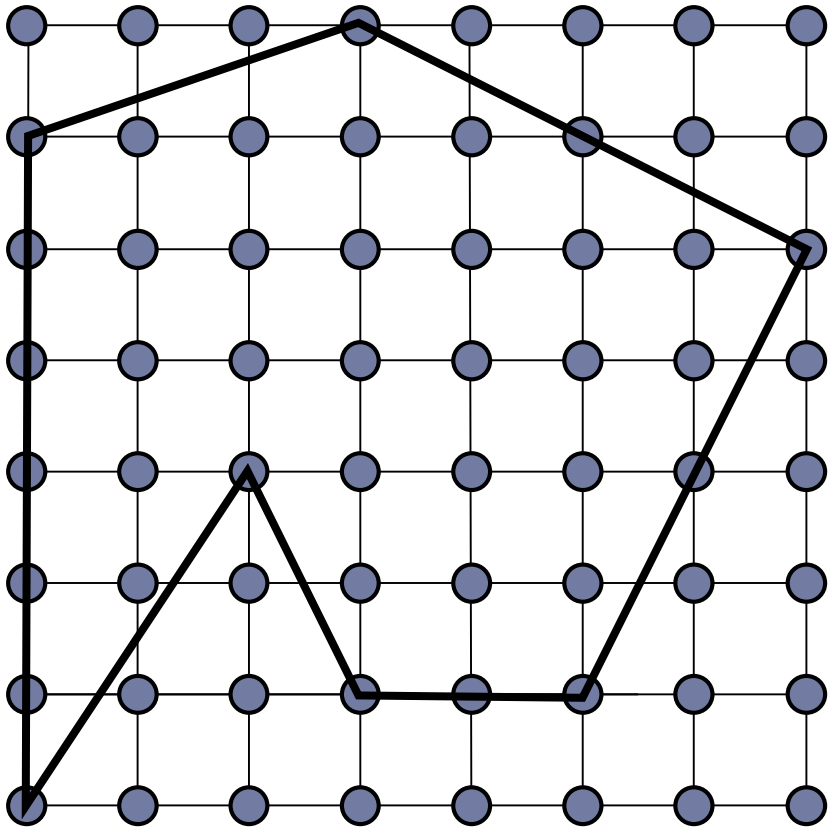


General Polygons – Basic Idea

- Intersect scan lines with edges
- Find ranges along *x*
- Fill interior of those ranges

Don't fill top/right

Edges may NOT match with line drawing algorithm



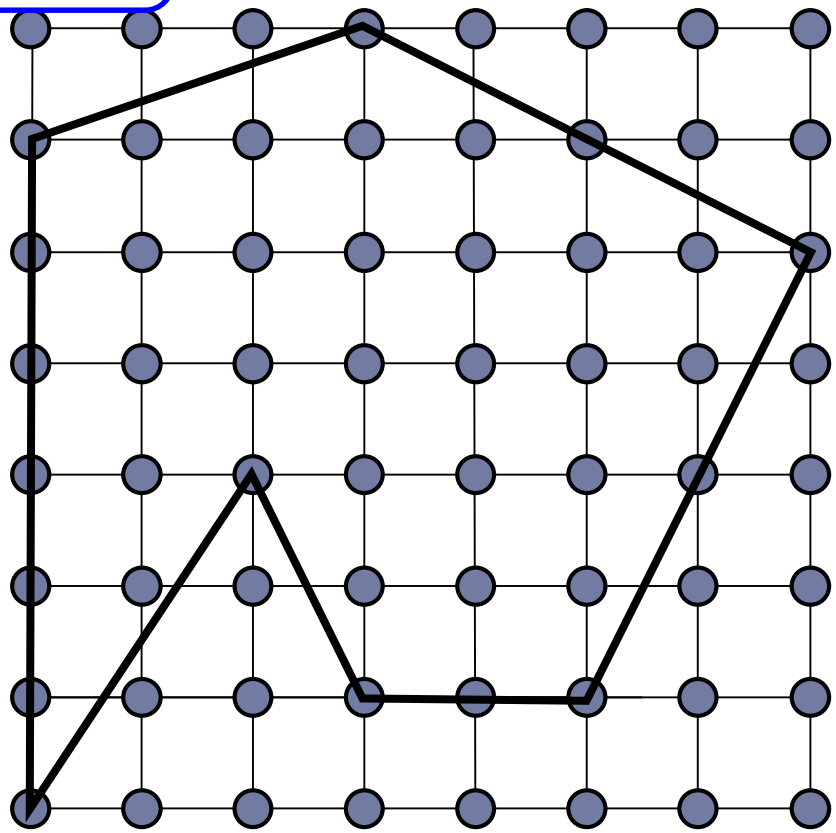
General Polygons – Basic Idea

- Intersect scan lines with edges
- Find ranges along x
- Fill interior of those ranges

Use coherence to speed up

Don't fill top/right

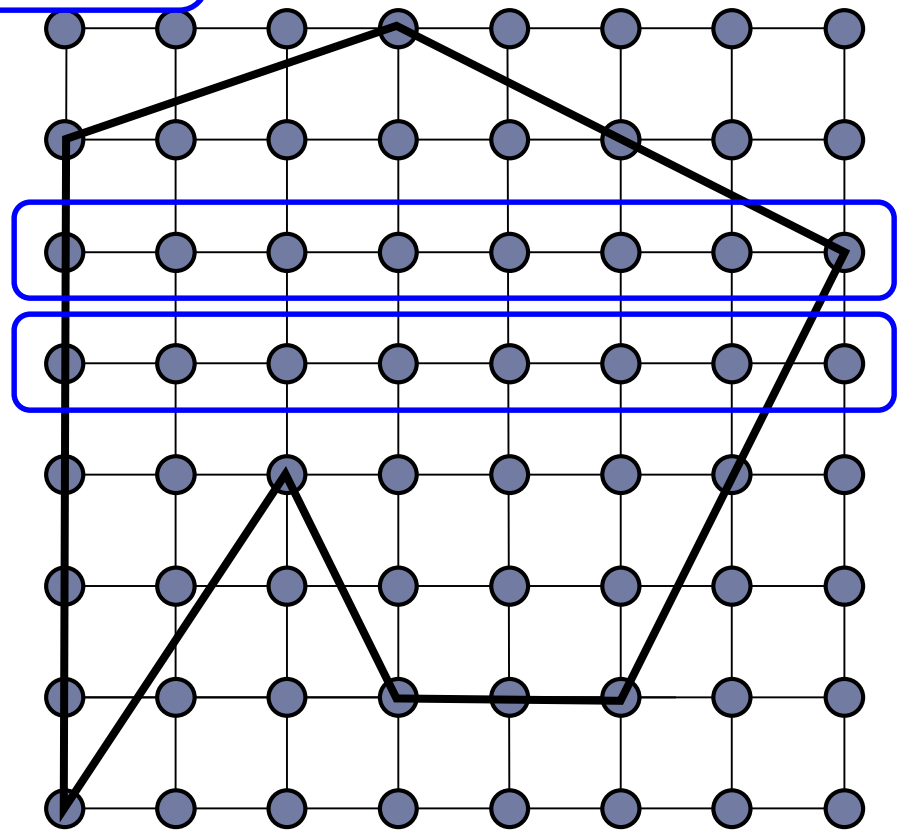
Edges may NOT match with line drawing algorithm



General Polygons – Basic Idea

- Intersect scan lines with edges
- Find ranges along x
- Fill interior of those ranges

Use coherence to speed up

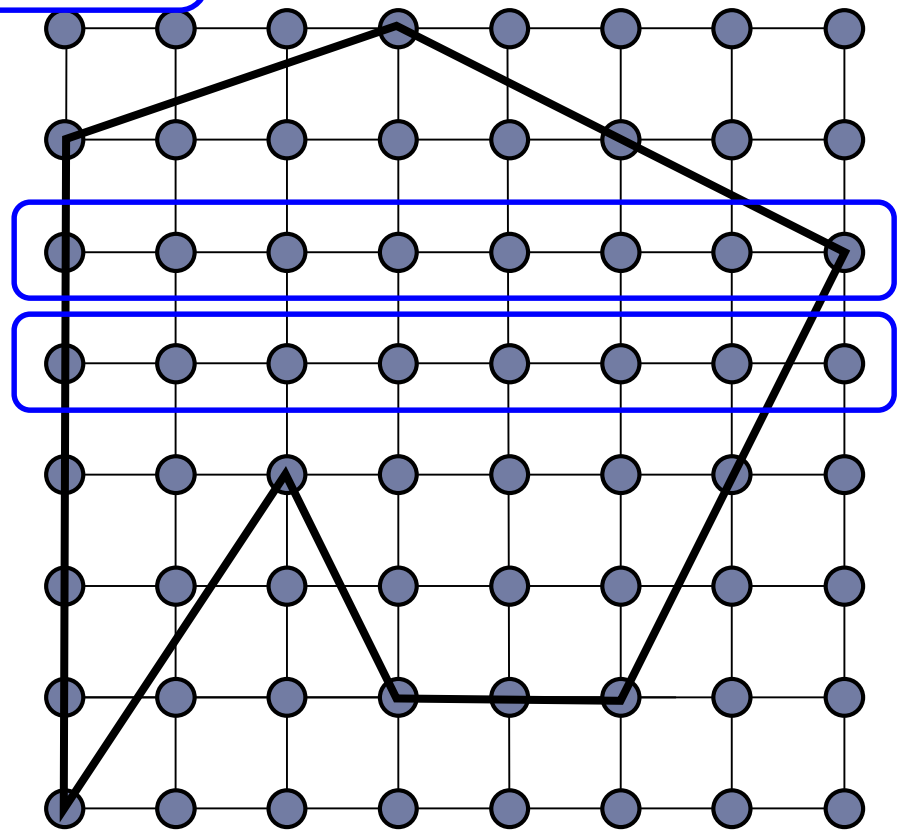


Edges intersecting one scan line are mostly same as those intersecting previous scan line

General Polygons – Basic Idea

- Intersect scan lines with edges
- Find ranges along x
- Fill interior of those ranges

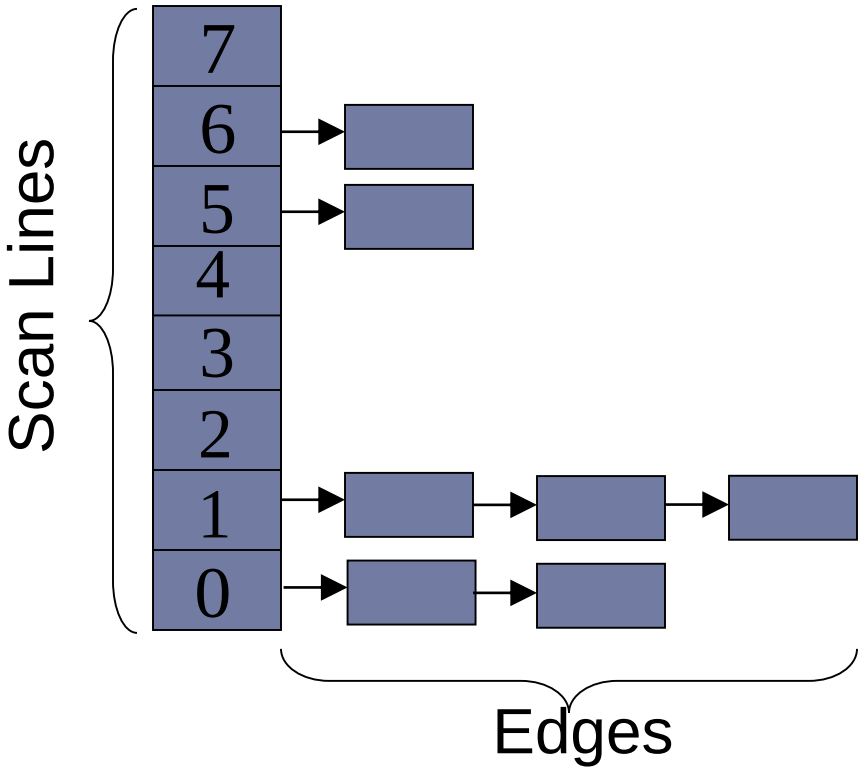
Use coherence to speed up



The x-value of an intersection with one scan line is close to the intersection with the previous one

General Polygons – Data Structures

- Active Edge Table:

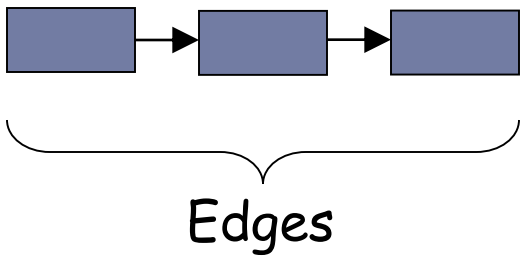


Store a linked-list per scan-line.

Insert edges into table at scan-line associated with lowest end-point.

General Polygons – Data Structures

- Active Edge List:



List of all edges intersecting current scan-line sorted by their *x*-values

General Polygons – Data Structures

- Edge:

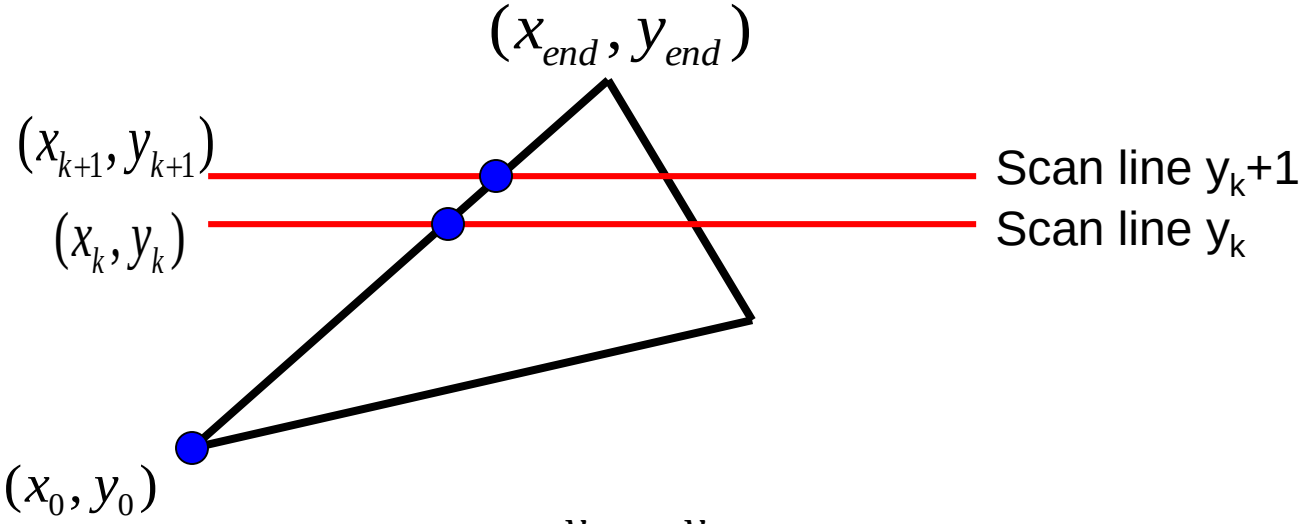
Edge
maxY
currentX
xIncr

maxY: highest *y*-value

currentX: *x*-value of end-point with lowest *y*-value

xIncr: 1 / slope

Scan line intersection

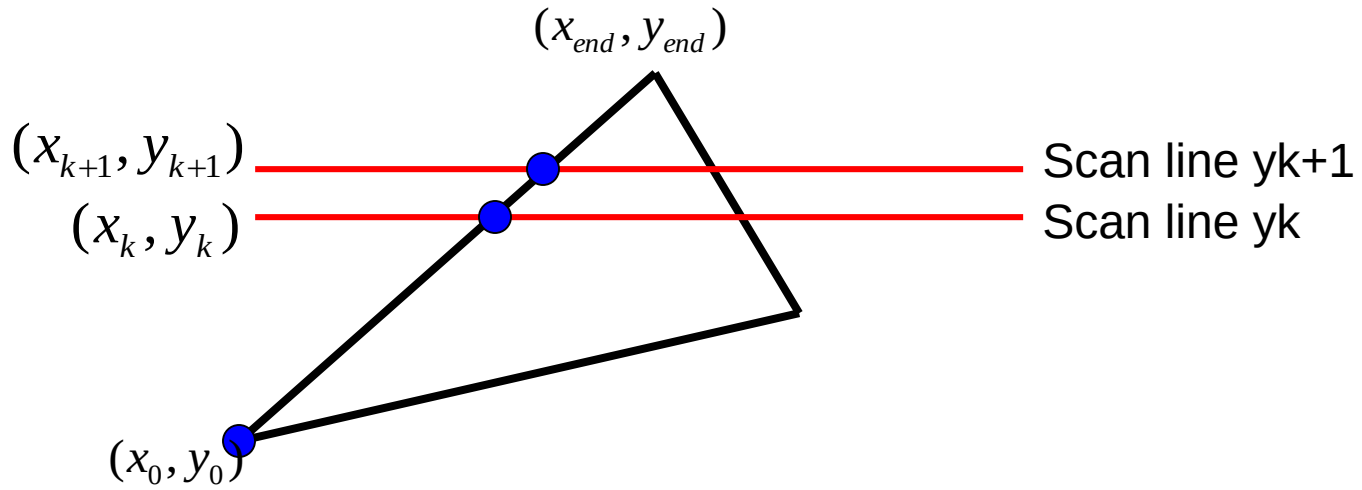


$$m = \frac{y_{end} - y_0}{x_{end} - x_0}$$

$$x_{k+1} = x_k + \frac{1}{m}$$

$$x_k = x_0 + \frac{k}{m}$$

Scan line intersection



$$m = \frac{y_{end} - y_0}{x_{end} - x_0}$$

$$x_{k+1} = x_k + \frac{1}{m}$$

$$x_k = x_0 + \frac{k}{m}$$

General Polygons – Data Structures

- Edge:

Edge
maxY
currentX
xIncr

maxY: highest *y*-value

currentX: *x*-value of end-point with lowest *y*-value

xIncr: 1 / slope

Horizontal edges will not be used!!!

General Polygons – Algorithm

line = 0

While (*line* < height)

  Add edges to Active Edge List from Active Edge Table starting at *line*

  Remove edges that end at *line*

  Fill pixels

  Increment *x*-values on edges in Active Edge List

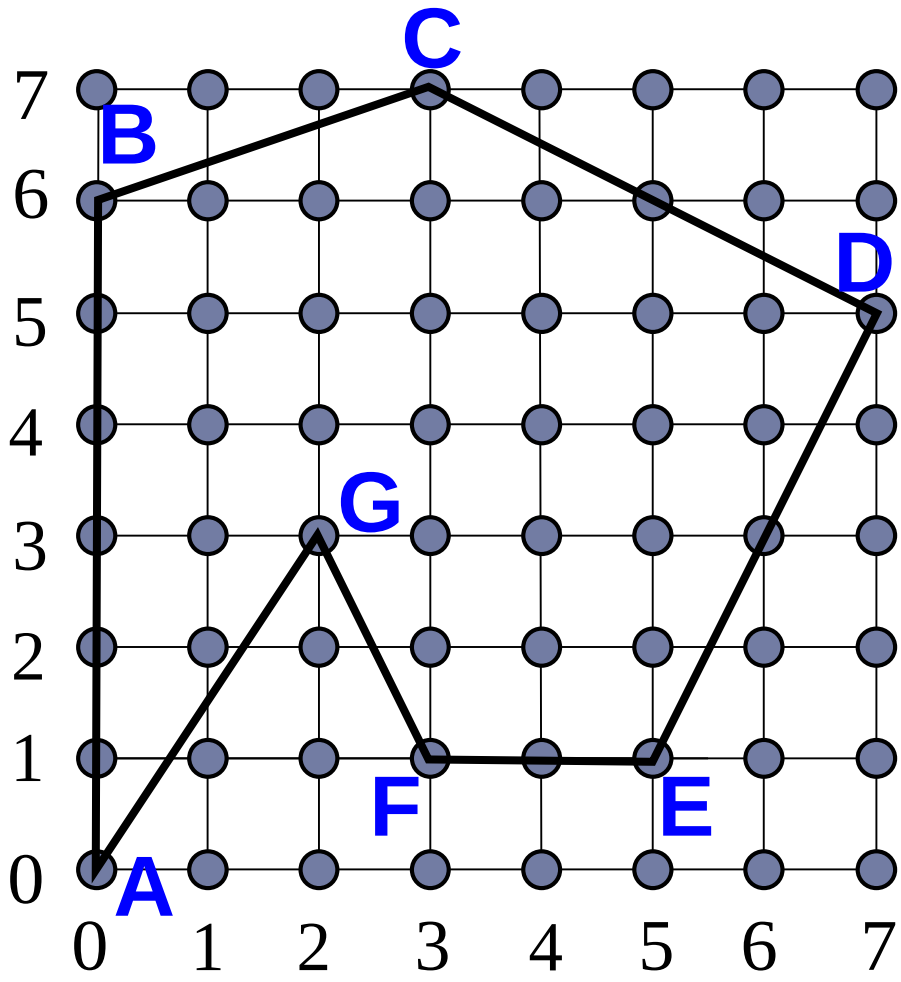
  Increment *line*

General Polygons – Example

Active Edge Table

7
6
5
4
3
2
1
0

Active Edge List

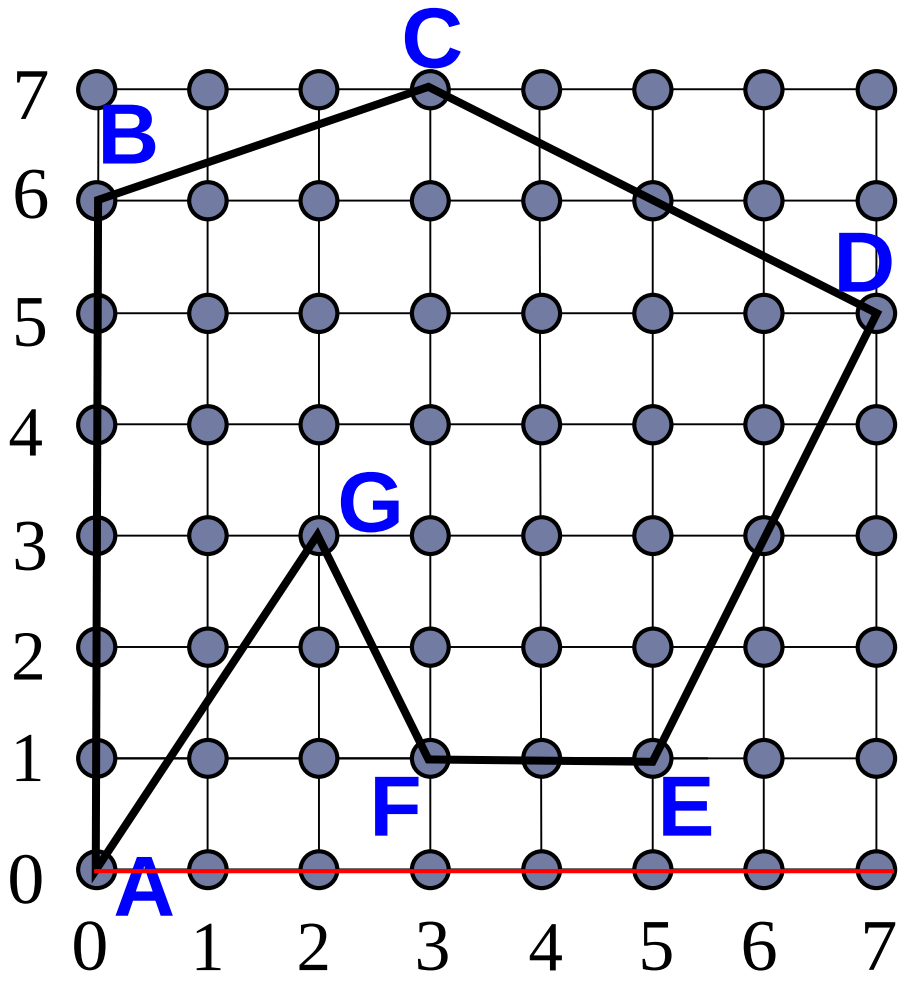


General Polygons – Example

Active Edge Table

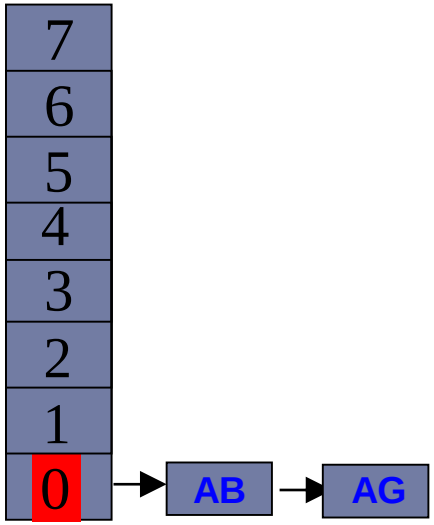
7
6
5
4
3
2
1
0

Active Edge List

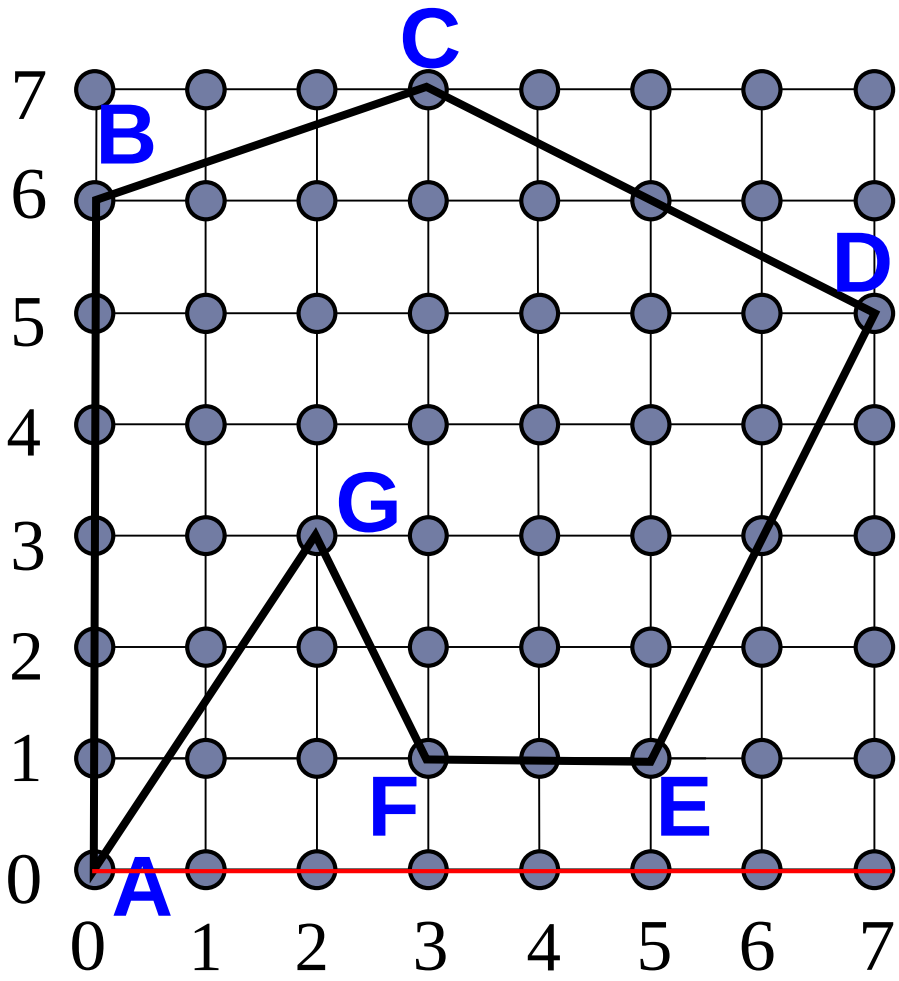


General Polygons – Example

Active Edge Table

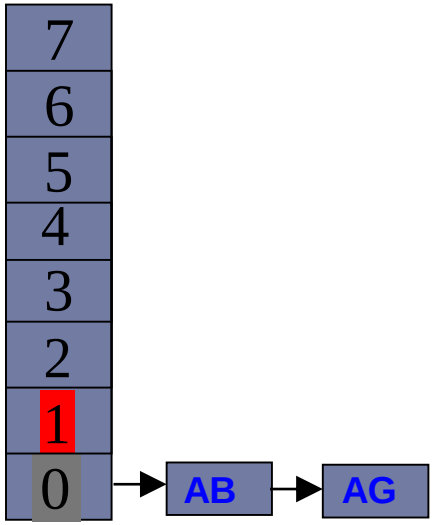


Active Edge List

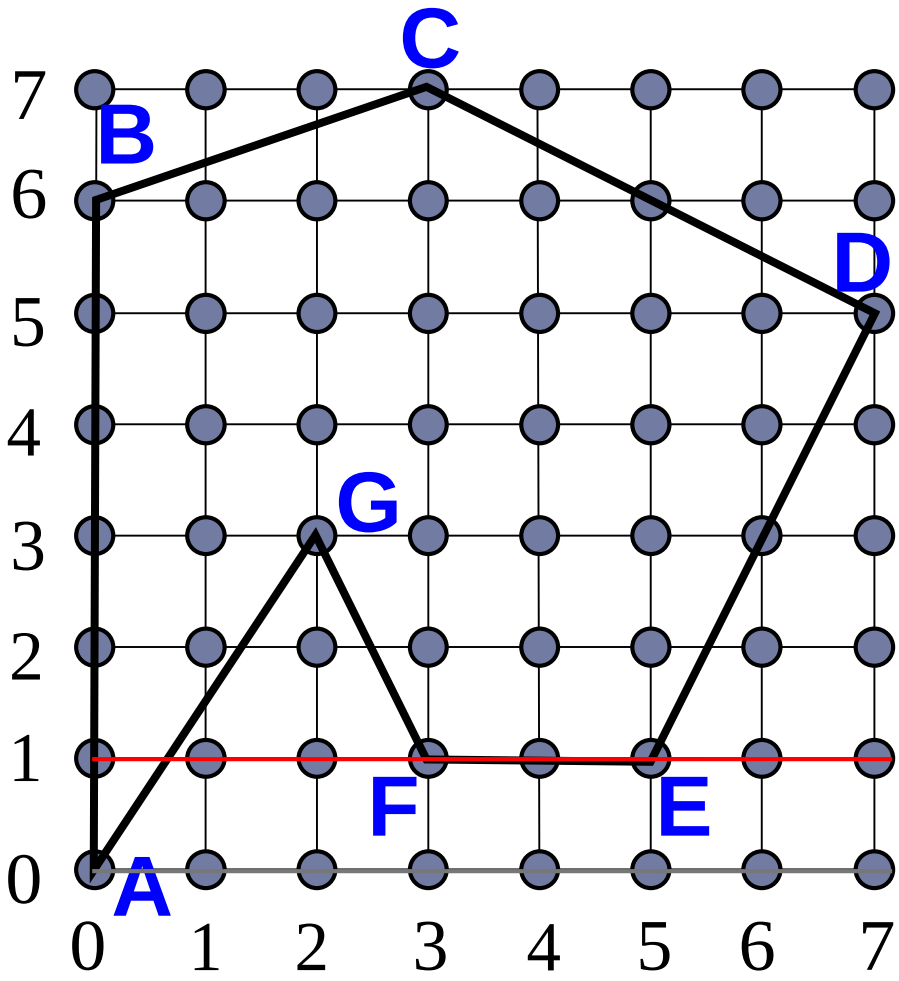


General Polygons – Example

Active Edge Table

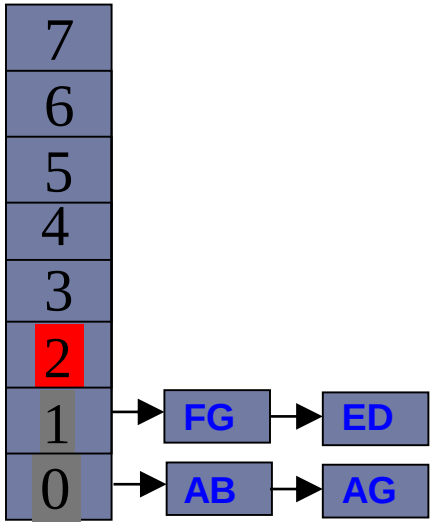


Active Edge List

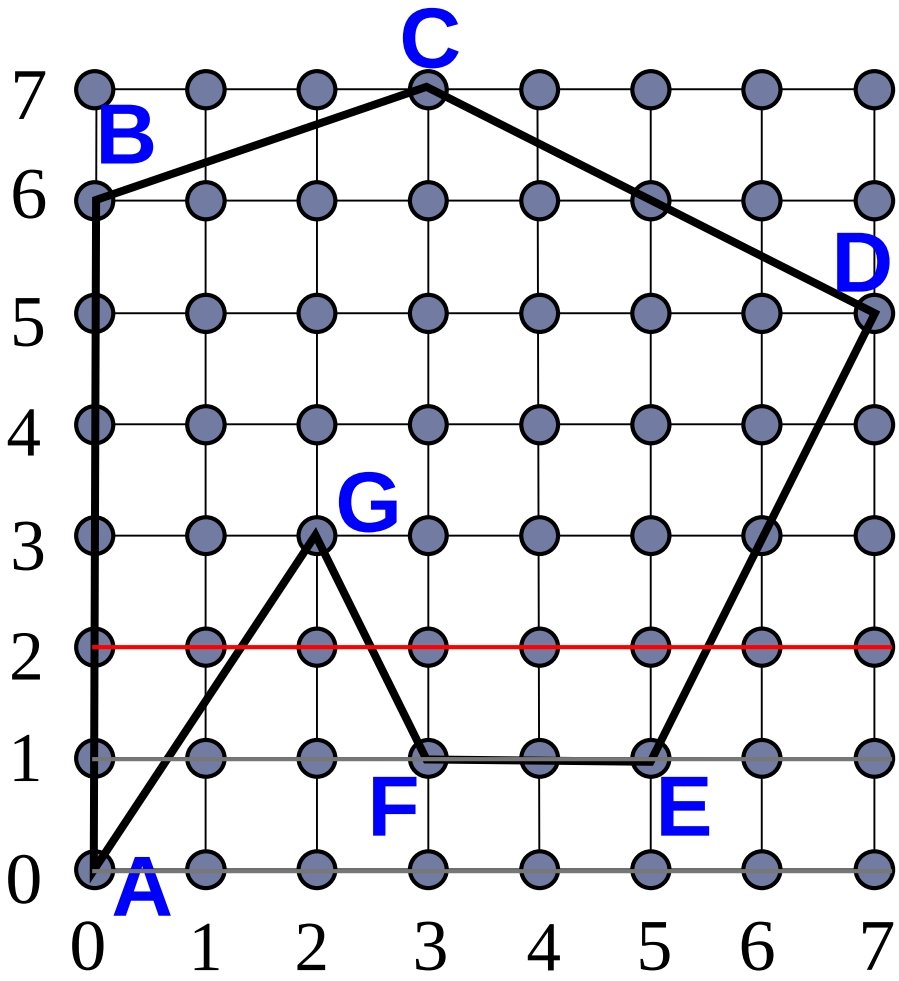


General Polygons – Example

Active Edge Table

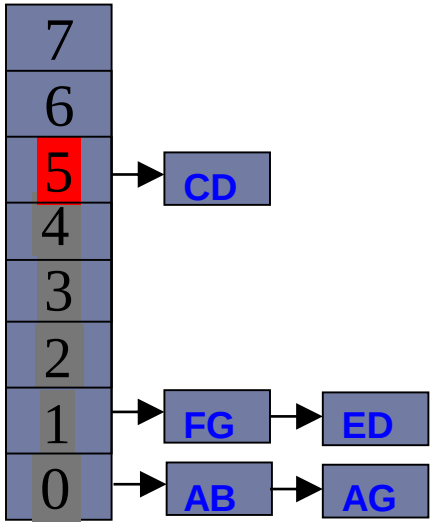


Active Edge List

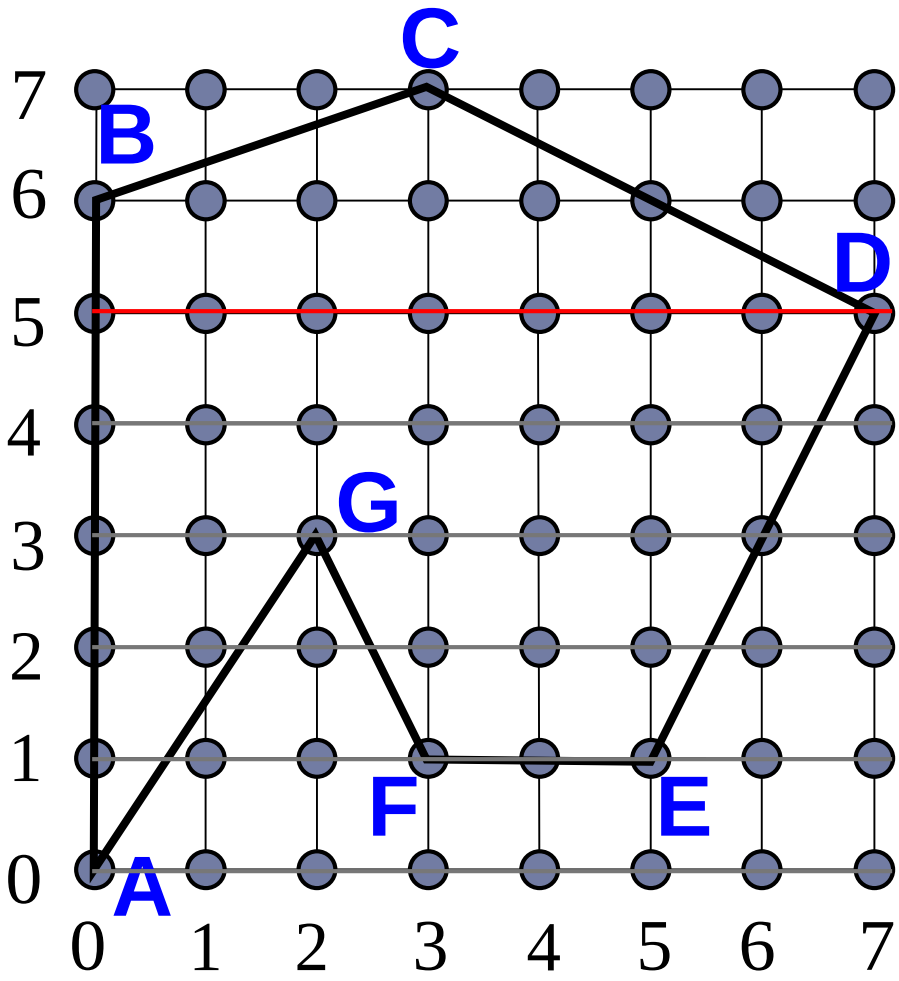


General Polygons – Example

Active Edge Table

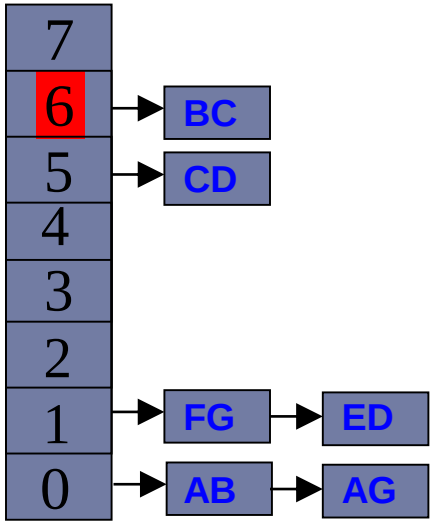


Active Edge List

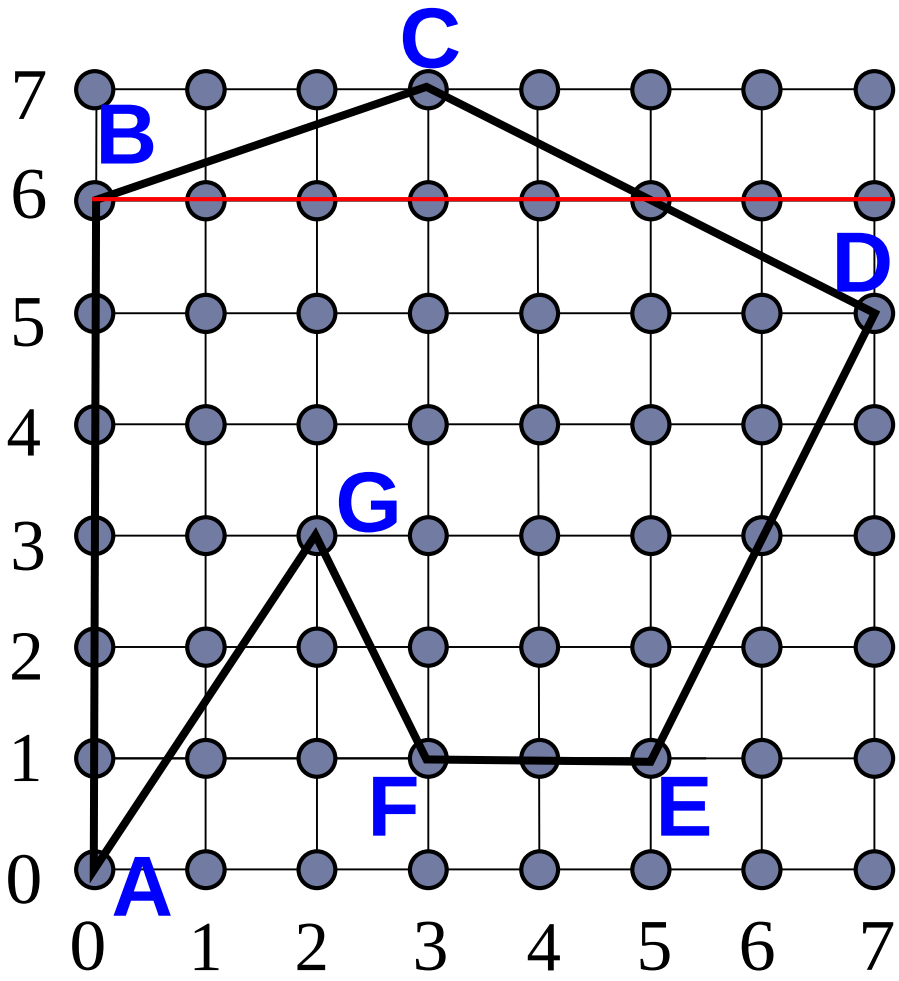


General Polygons – Example

Active Edge Table

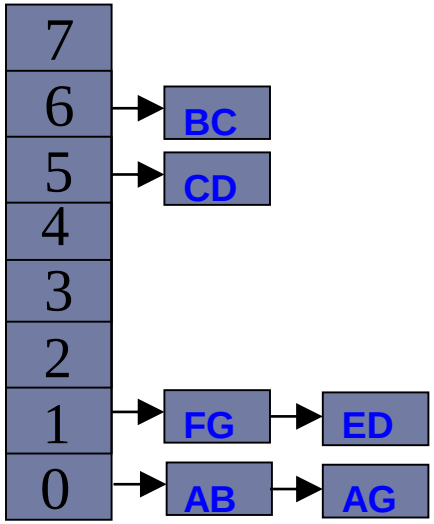


Active Edge List



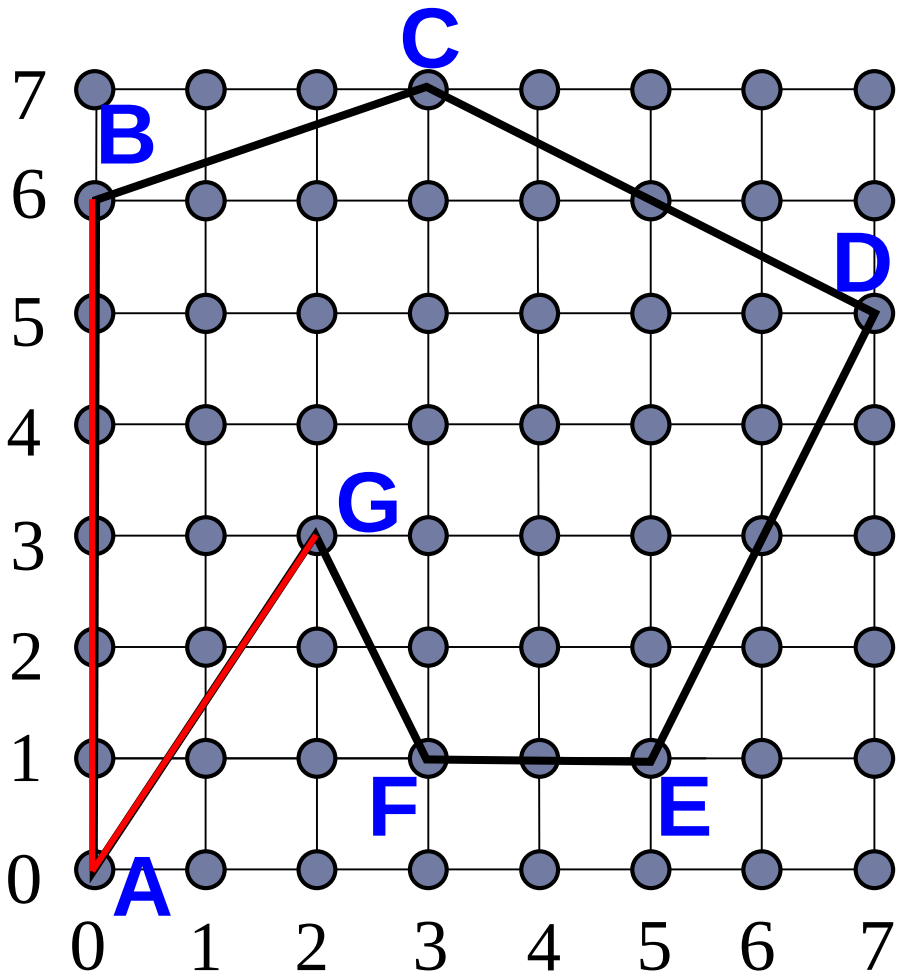
General Polygons – Example

Active Edge Table



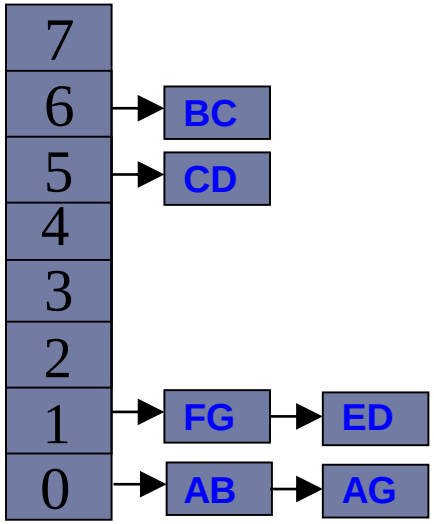
Active Edge List

	AB	AG
maxY	6	3
currentX	0	0
xIncr	0	⅔

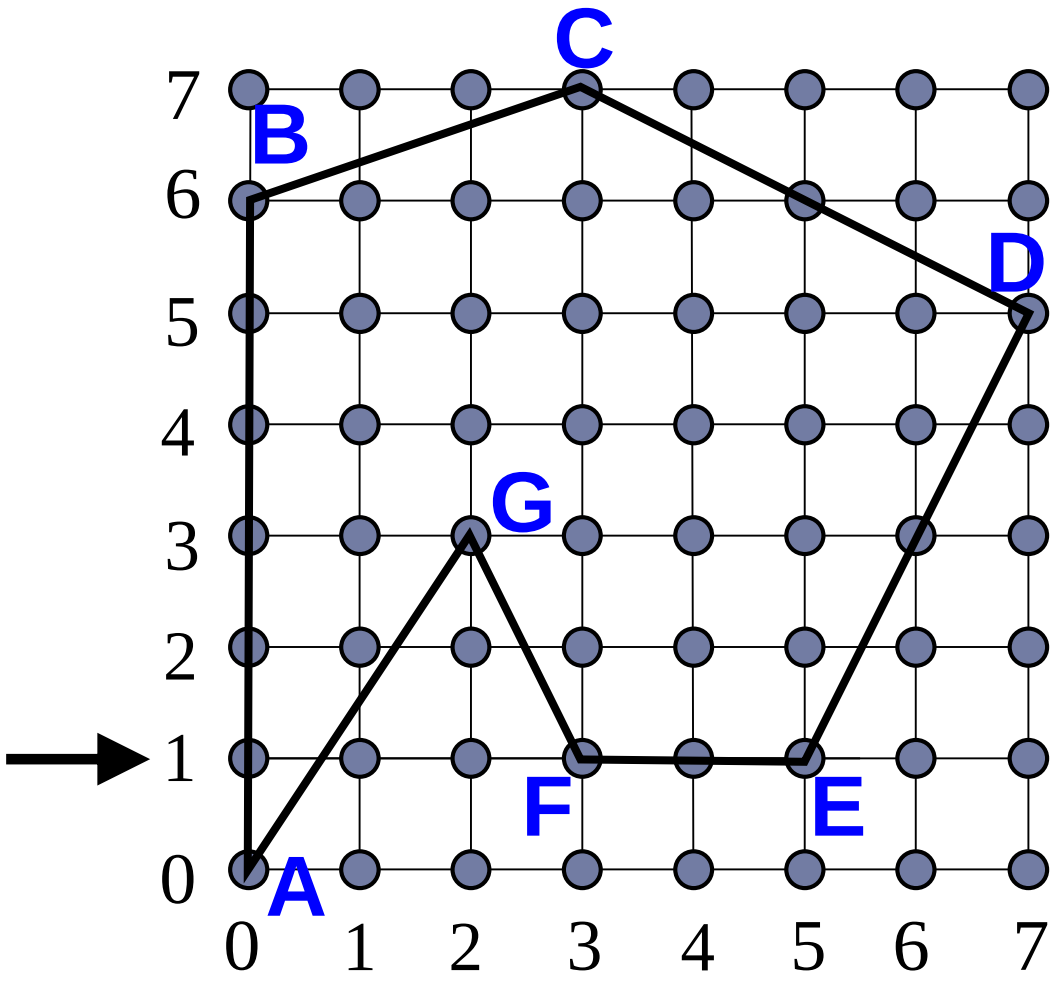


General Polygons – Example

Active Edge Table

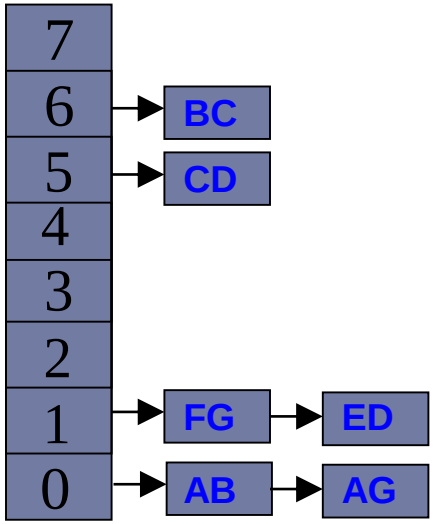


Active Edge List?



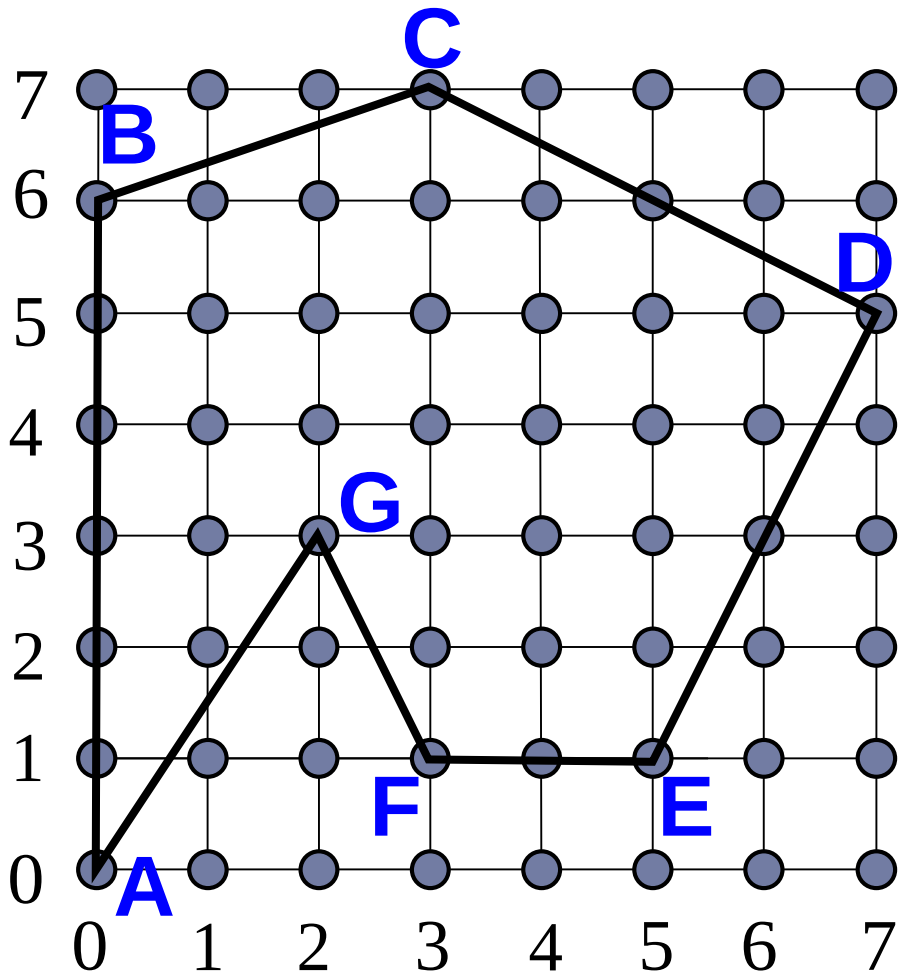
General Polygons – Example

Active Edge Table



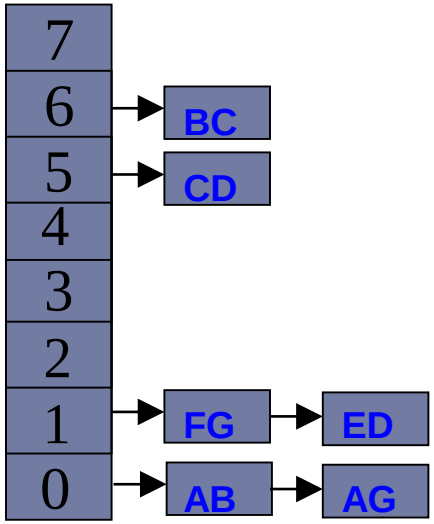
Active Edge List

	FG	ED
maxY	3	5
currentX	3	5
xIncr	$-1/2$	$1/2$



General Polygons – Example

Active Edge Table



Active Edge List

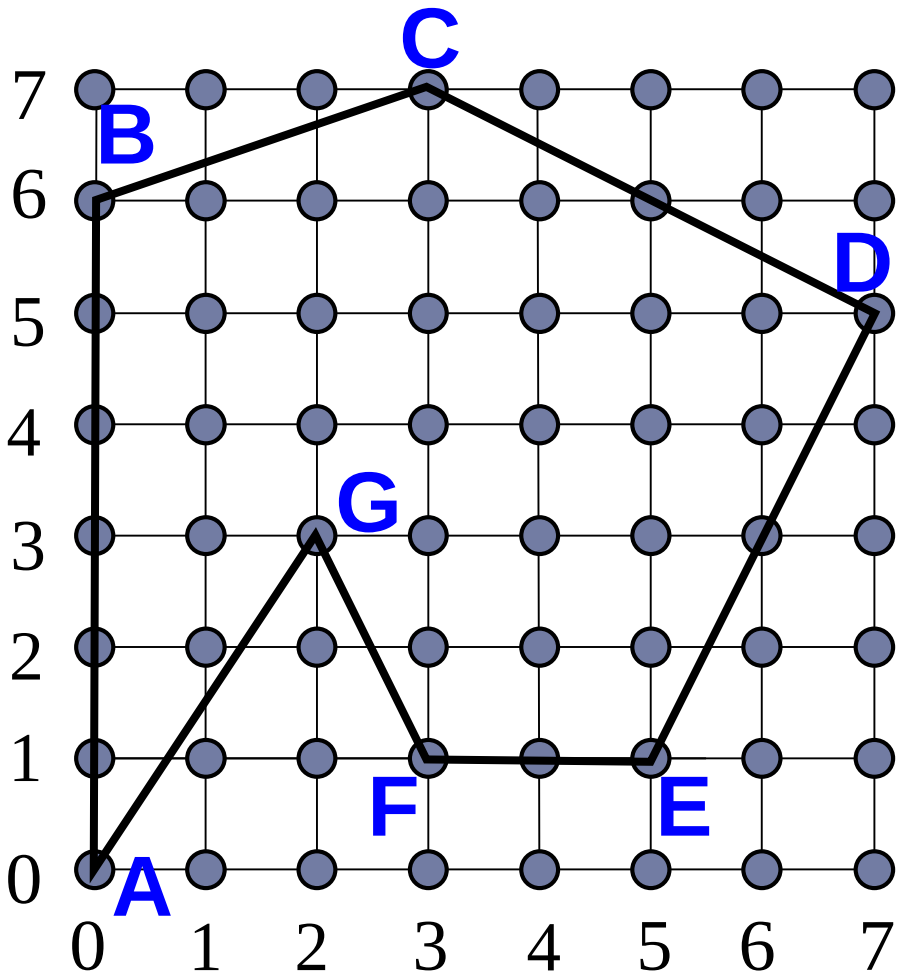
FG	ED
3	5
3	5
$-1/2$	$1/2$

maxY

currentX

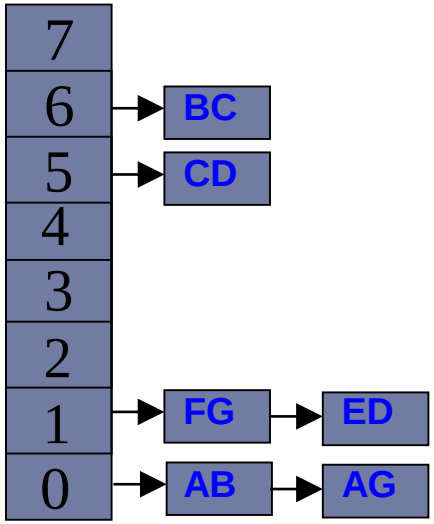
xIncr

Is it correct?



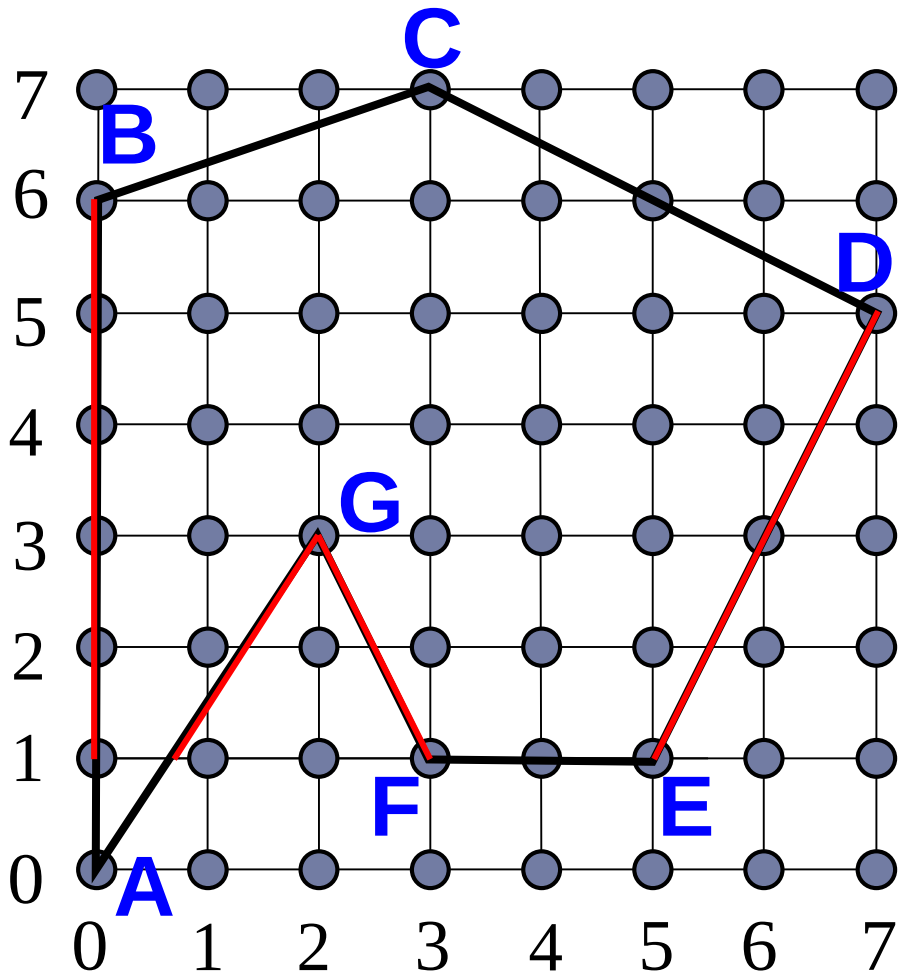
General Polygons – Example

Active Edge Table



Active Edge List

	AB	AG	FG	ED
maxY	6	3	3	5
currentX	0	$\frac{2}{3}$	3	5
xIncr	0	$\frac{2}{3}$	$-\frac{1}{2}$	$\frac{1}{2}$



General Polygons – Algorithm

line = 0

While (*line* < height)

Add edges to Active Edge List from Active Edge Table starting at *line*

Remove edges that end at *line*

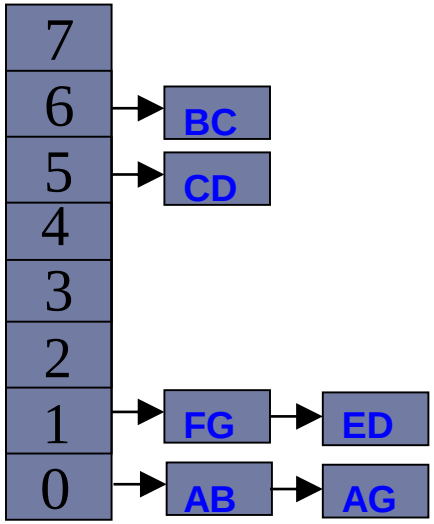
Fill pixels

Increment *x*-values on edges in Active Edge List

Increment *line*

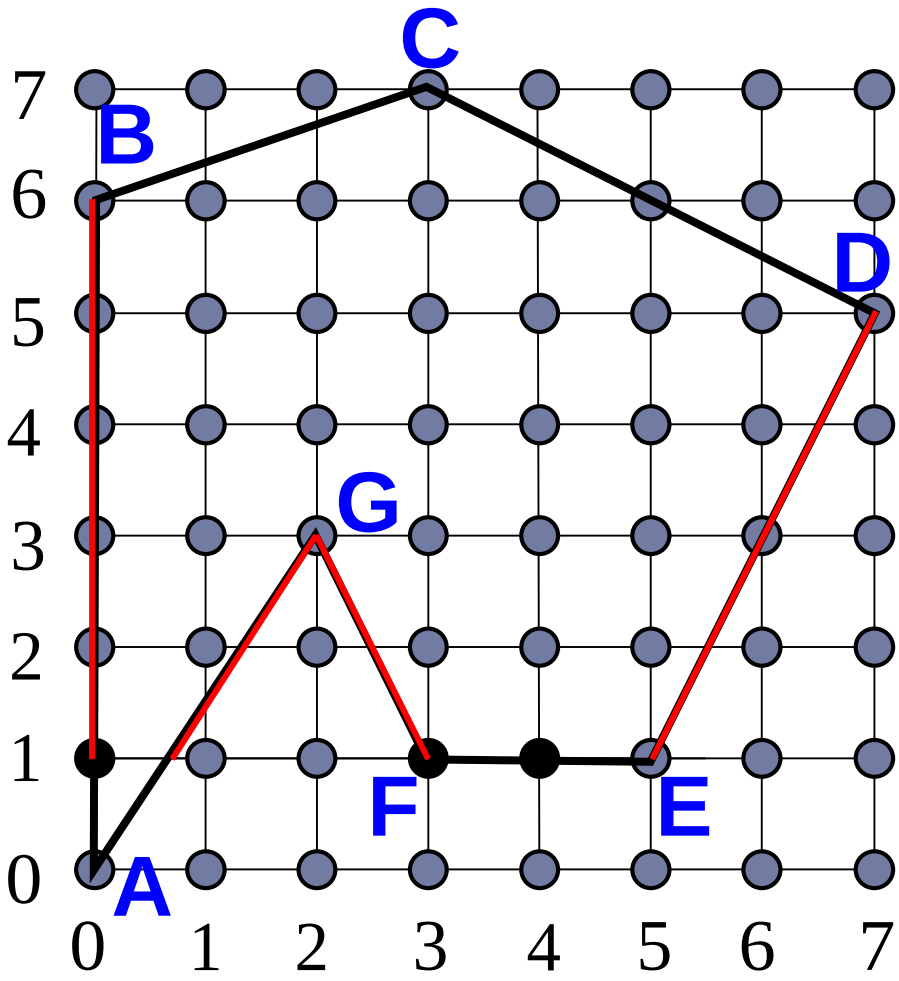
General Polygons – Example

Active Edge Table



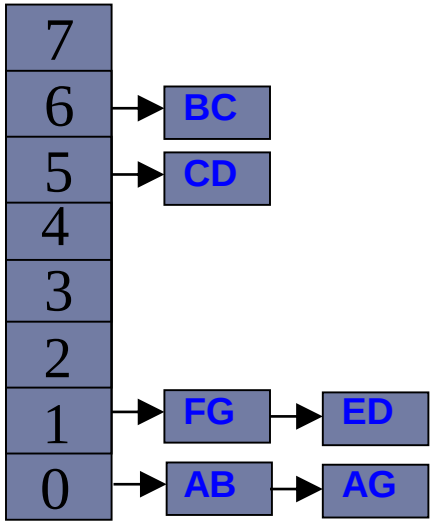
Active Edge List

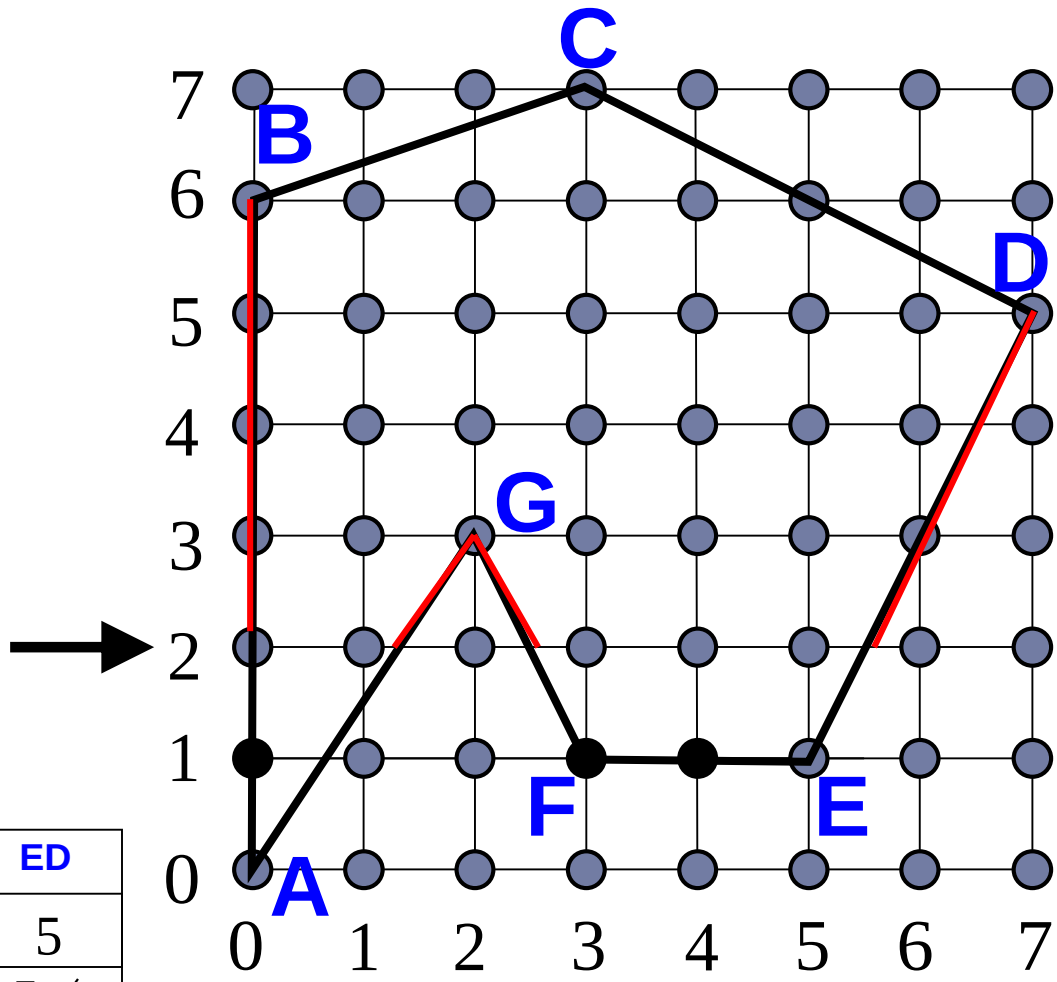
	AB	AG	FG	ED
maxY	6	3	3	5
currentX	0	$\frac{2}{3}$	3	5
xIncr	0	$\frac{2}{3}$	$-\frac{1}{2}$	$\frac{1}{2}$



General Polygons – Example

Active Edge Table



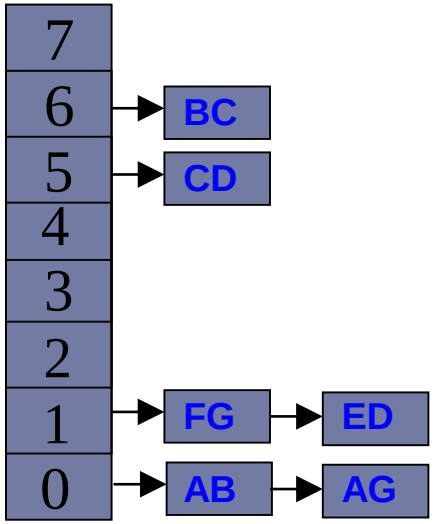


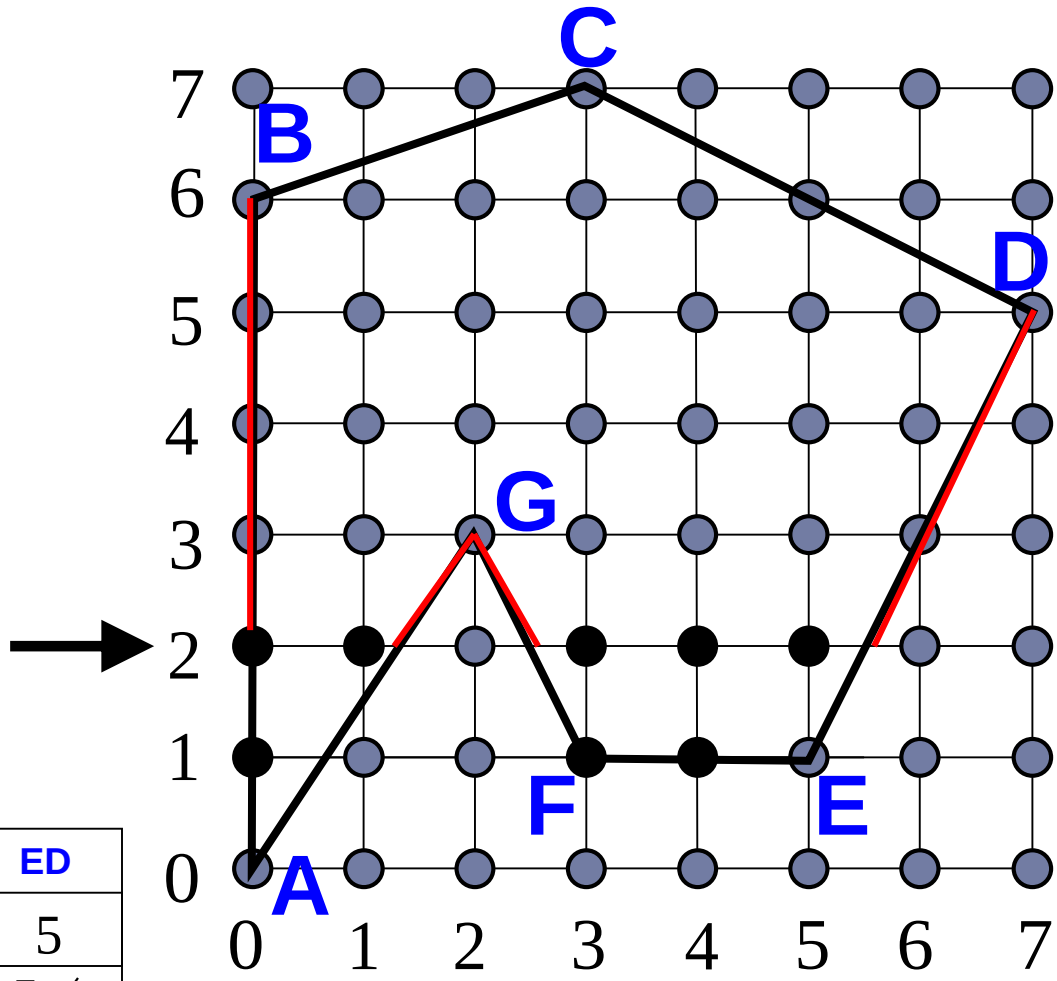
Active Edge List

	AB	AG	FG	ED
maxY	6	3	3	5
currentX	0	$\frac{4}{3}$	$2\frac{1}{2}$	$5\frac{1}{2}$
xIncr	0	$\frac{2}{3}$	$-\frac{1}{2}$	$\frac{1}{2}$

General Polygons – Example

Active Edge Table



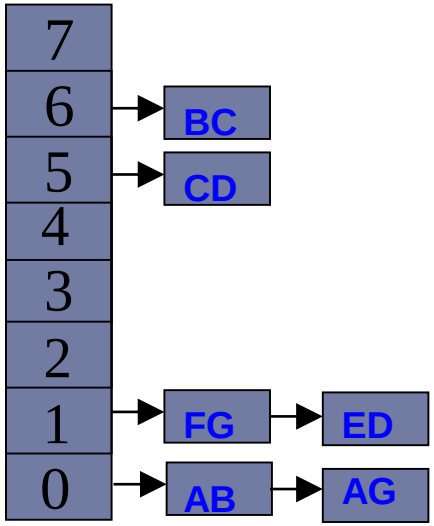


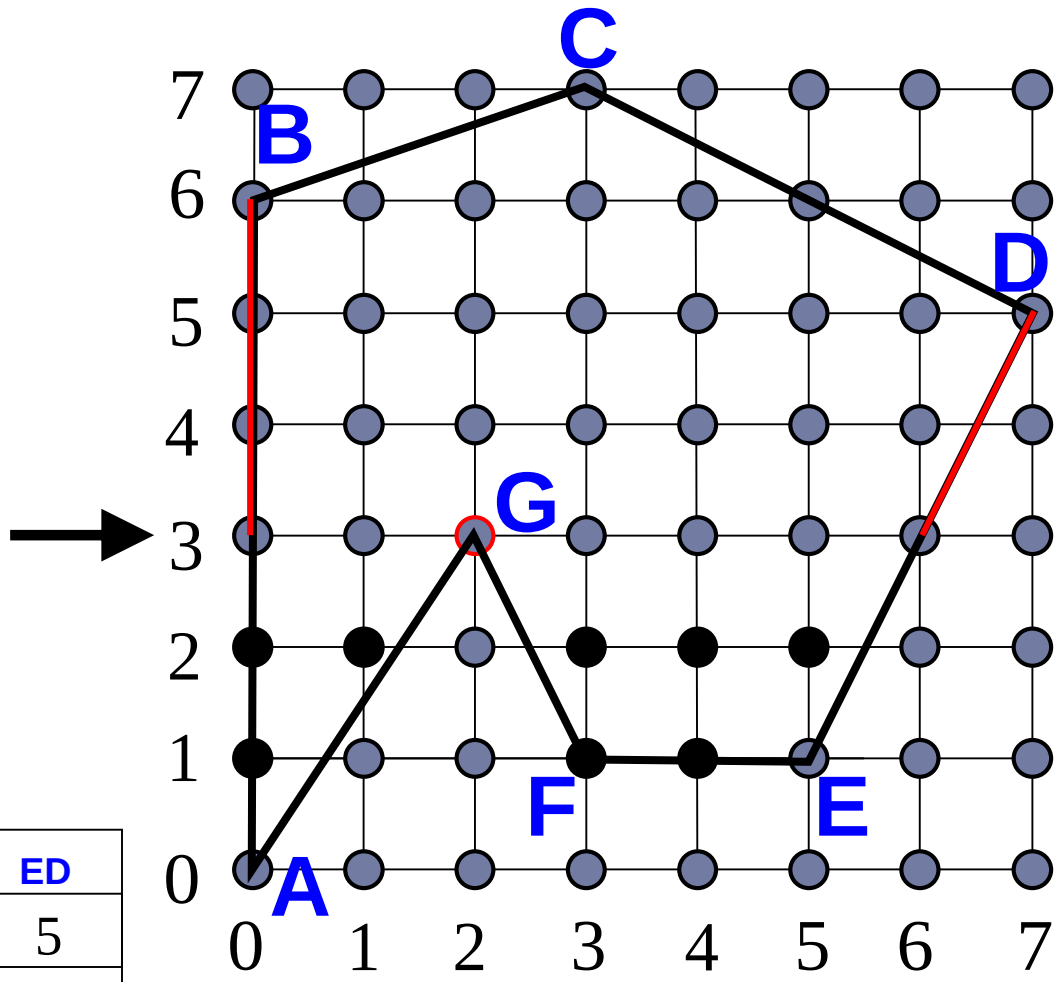
Active Edge List

	AB	AG	FG	ED
maxY	6	3	3	5
currentX	0	$\frac{4}{3}$	$2\frac{1}{2}$	$5\frac{1}{2}$
xIncr	0	$\frac{2}{3}$	$-\frac{1}{2}$	$\frac{1}{2}$

General Polygons – Example

Active Edge Table





Active Edge List

	AB	AG	FG	ED
maxY	6	3	3	5
currentX	0	2	2	6
xIncr	0	$\frac{2}{3}$	$-\frac{1}{2}$	$\frac{1}{2}$

General Polygons – Algorithm

line = 0

While (*line* < height)

Add edges to Active Edge List from Active Edge Table starting at *line*

Remove edges that end at *line*

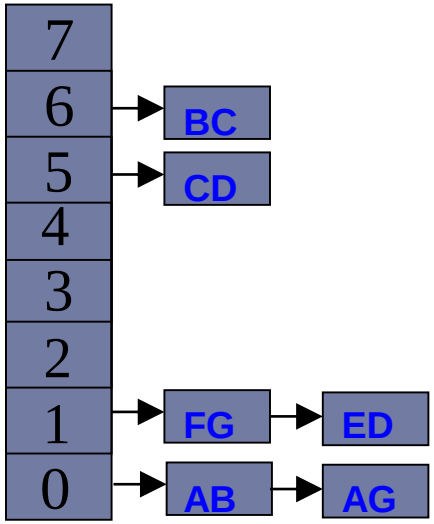
Fill pixels

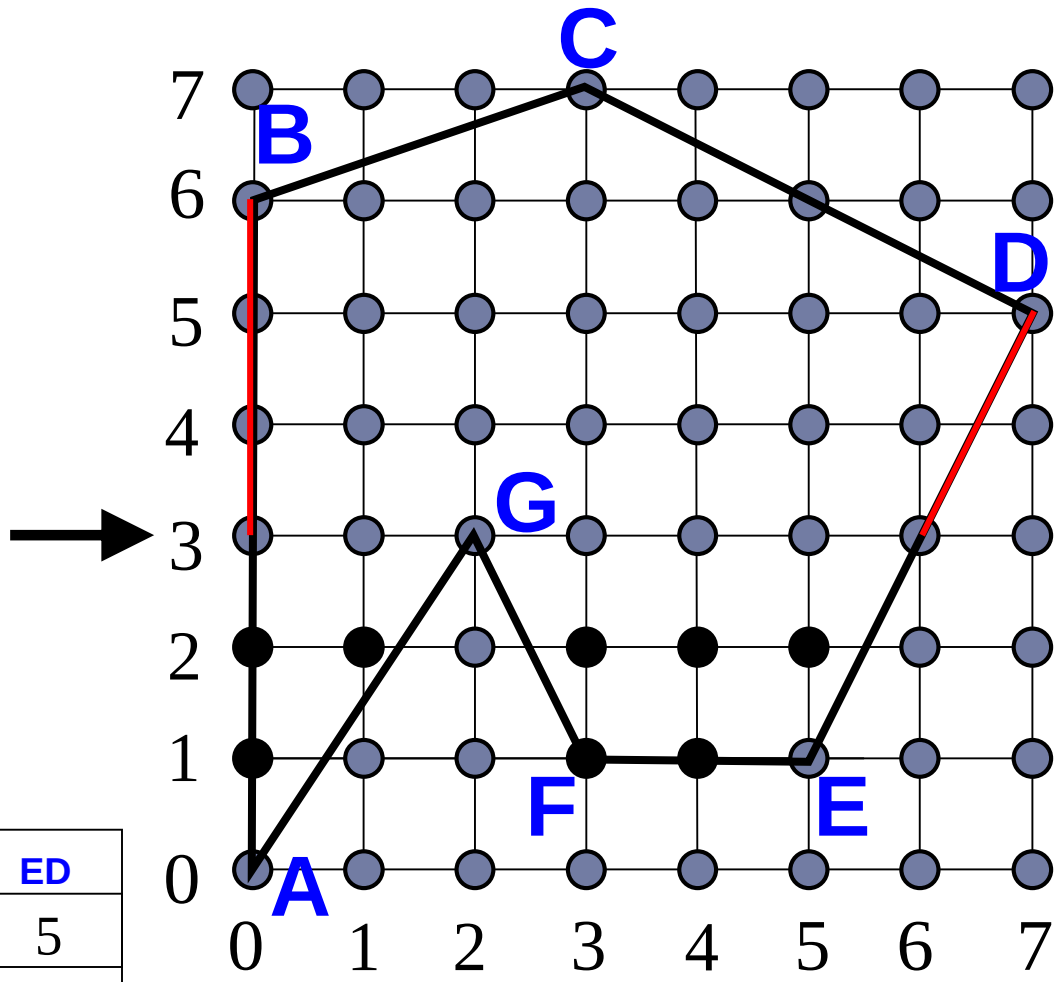
Increment *x*-values on edges in Active Edge List

Increment *line*

General Polygons – Example

Active Edge Table



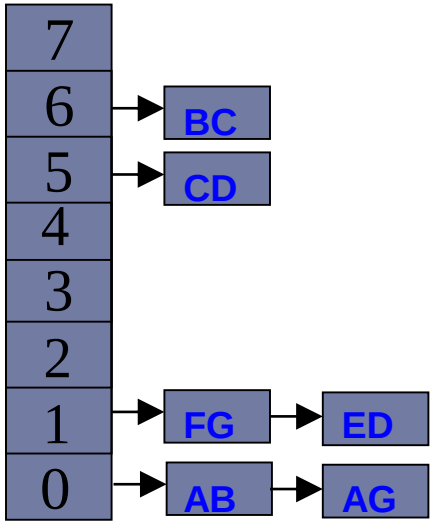


Active Edge List

	AB	~~AG~~	~~FG~~	ED
maxY	6	~~3~~	~~3~~	5
currentX	0	~~2~~	~~2~~	6
xIncr	0	~~$\frac{2}{3}$~~	~~$-\frac{1}{2}$~~	$\frac{1}{2}$

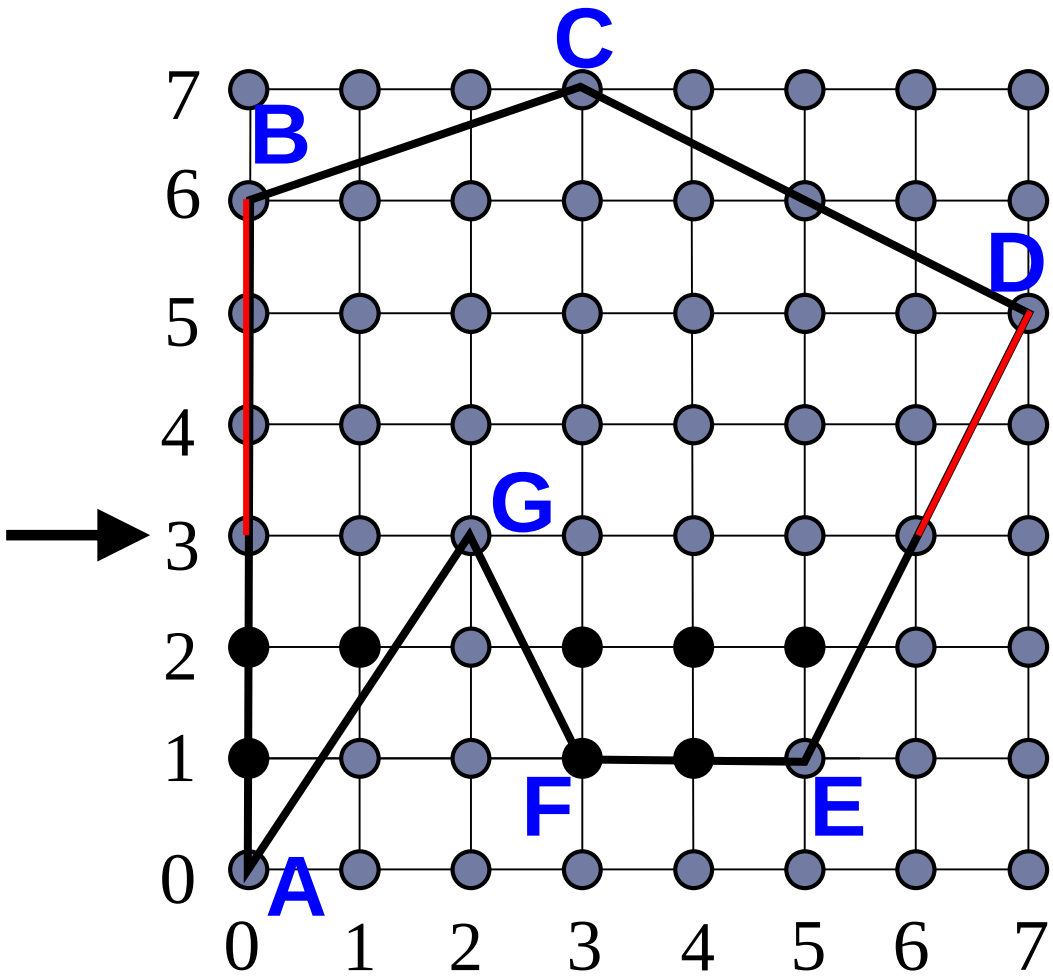
General Polygons – Example

Active Edge Table



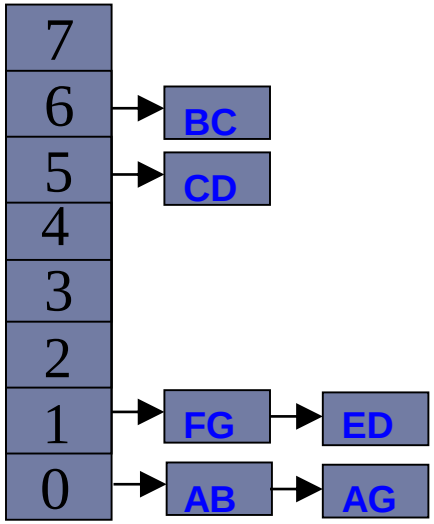
Active Edge List

	AB	ED
maxY	6	5
currentX	0	6
xIncr	0	½



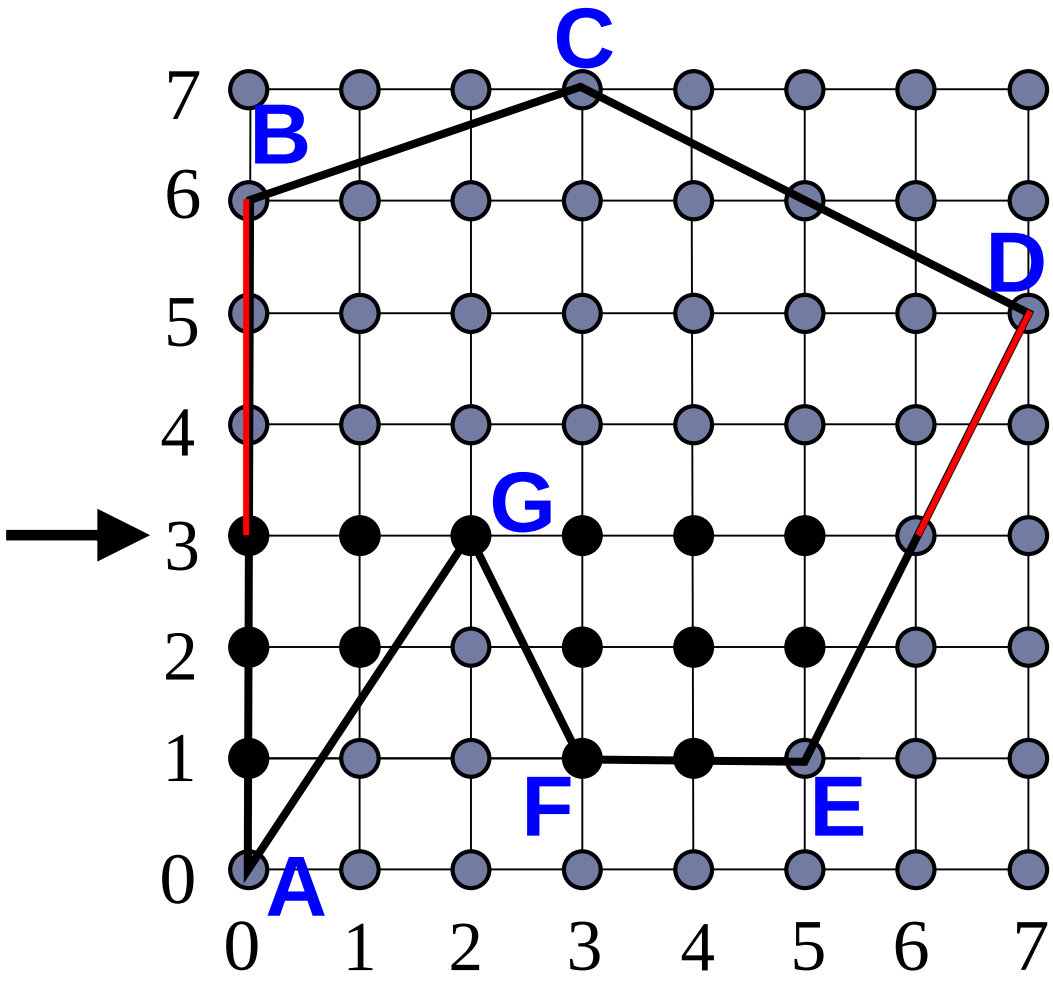
General Polygons – Example

Active Edge Table



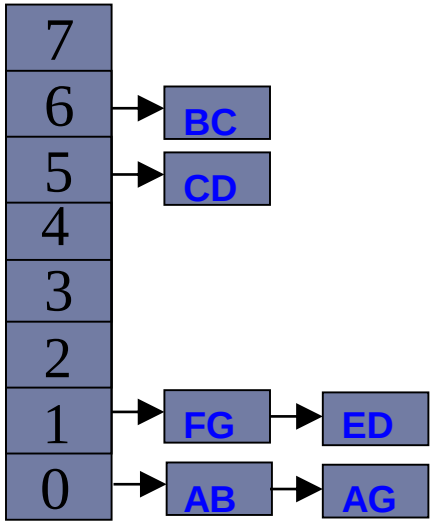
Active Edge List

	AB	ED
maxY	6	5
currentX	0	6
xIncr	0	½



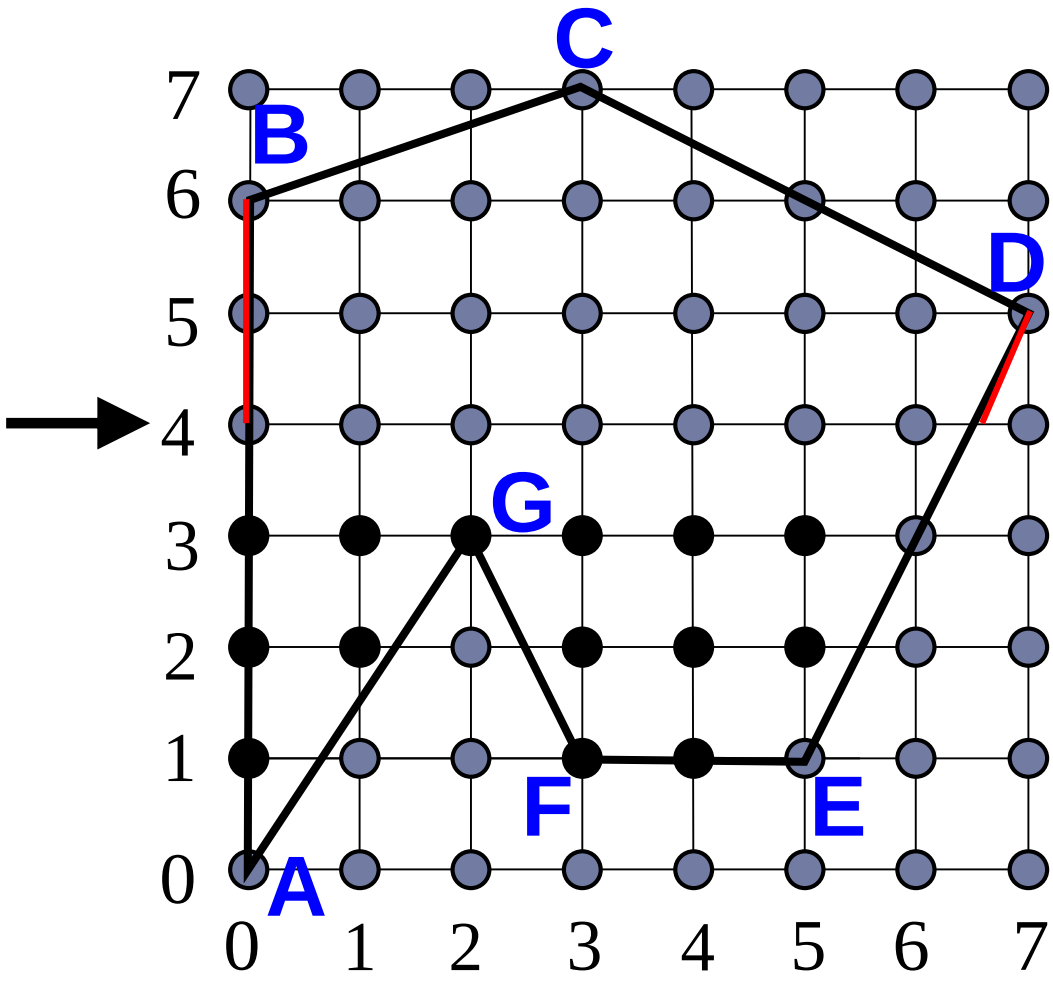
General Polygons – Example

Active Edge Table



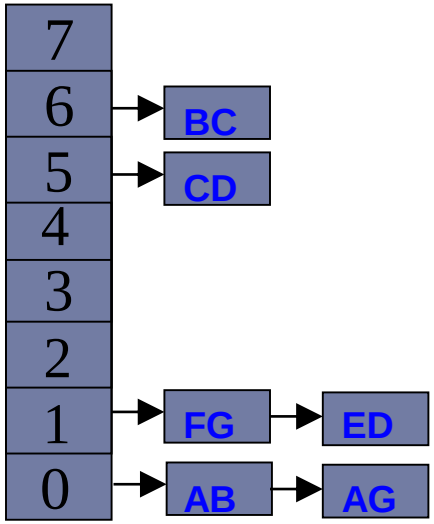
Active Edge List

	AB	ED
maxY	6	5
currentX	0	6½
xIncr	0	½



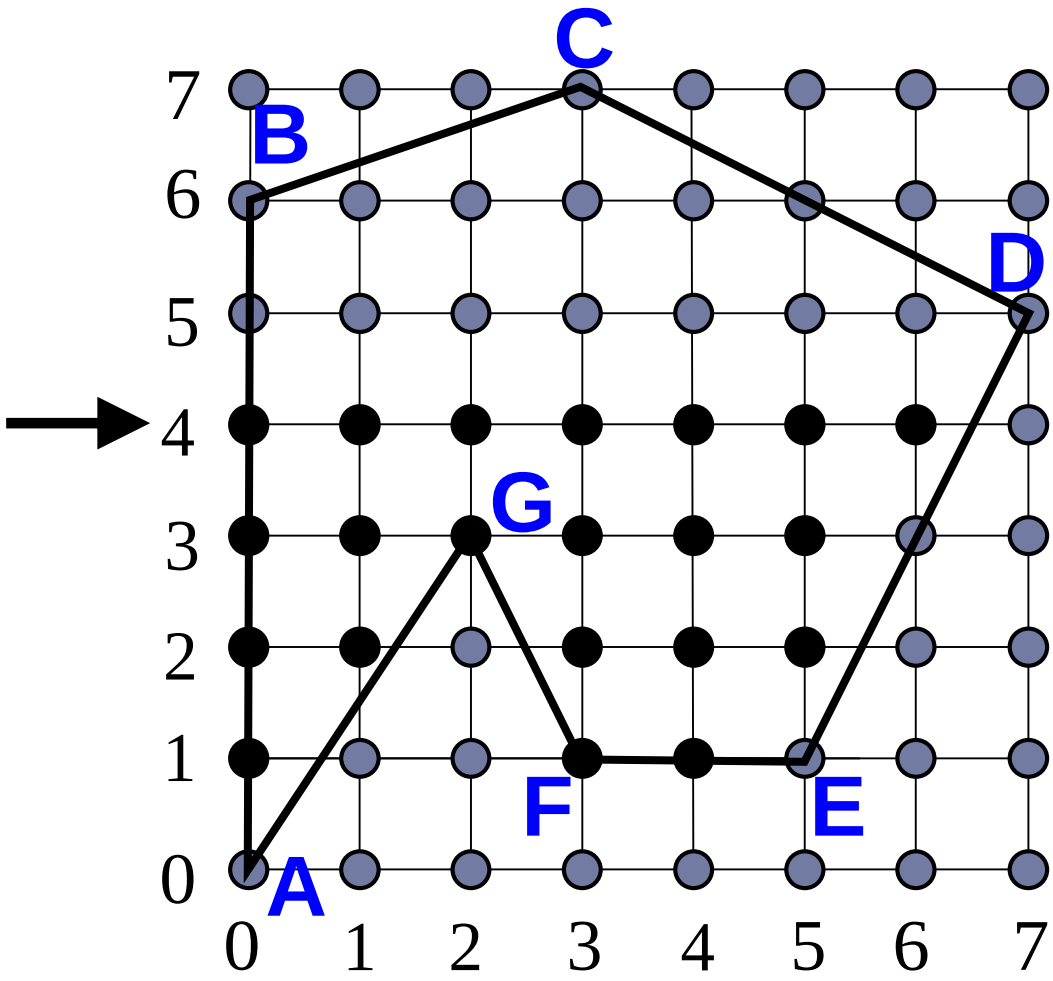
General Polygons – Example

Active Edge Table



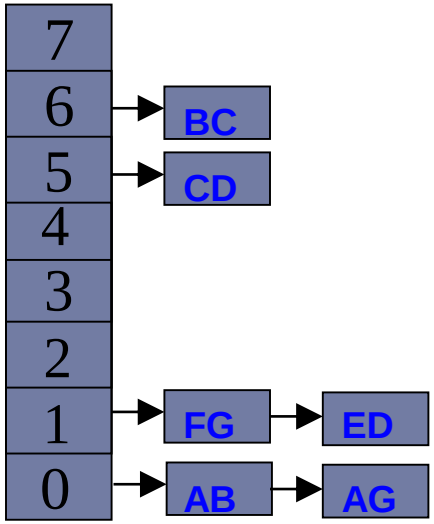
Active Edge List

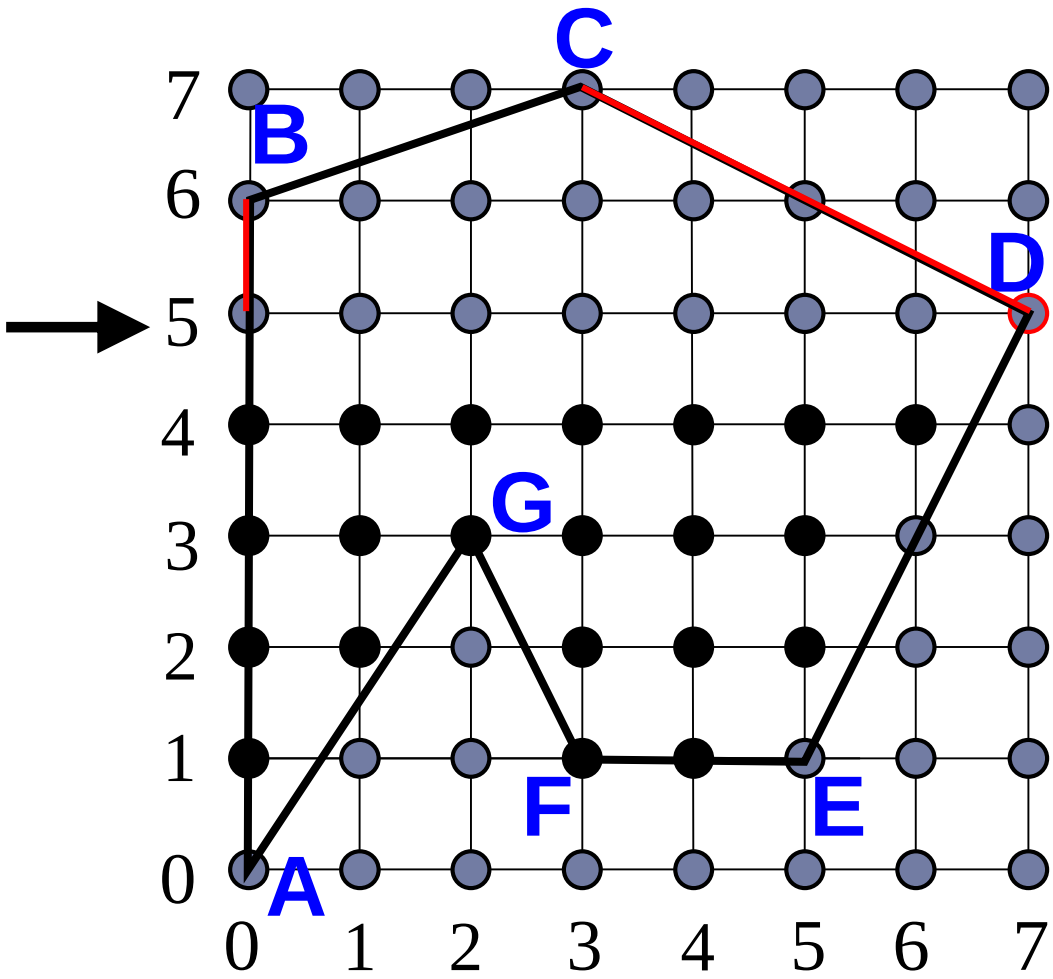
	AB	ED
maxY	6	5
currentX	0	6½
xIncr	0	½



General Polygons – Example

Active Edge Table



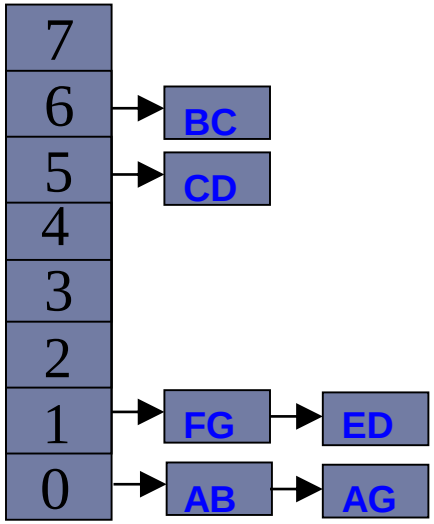


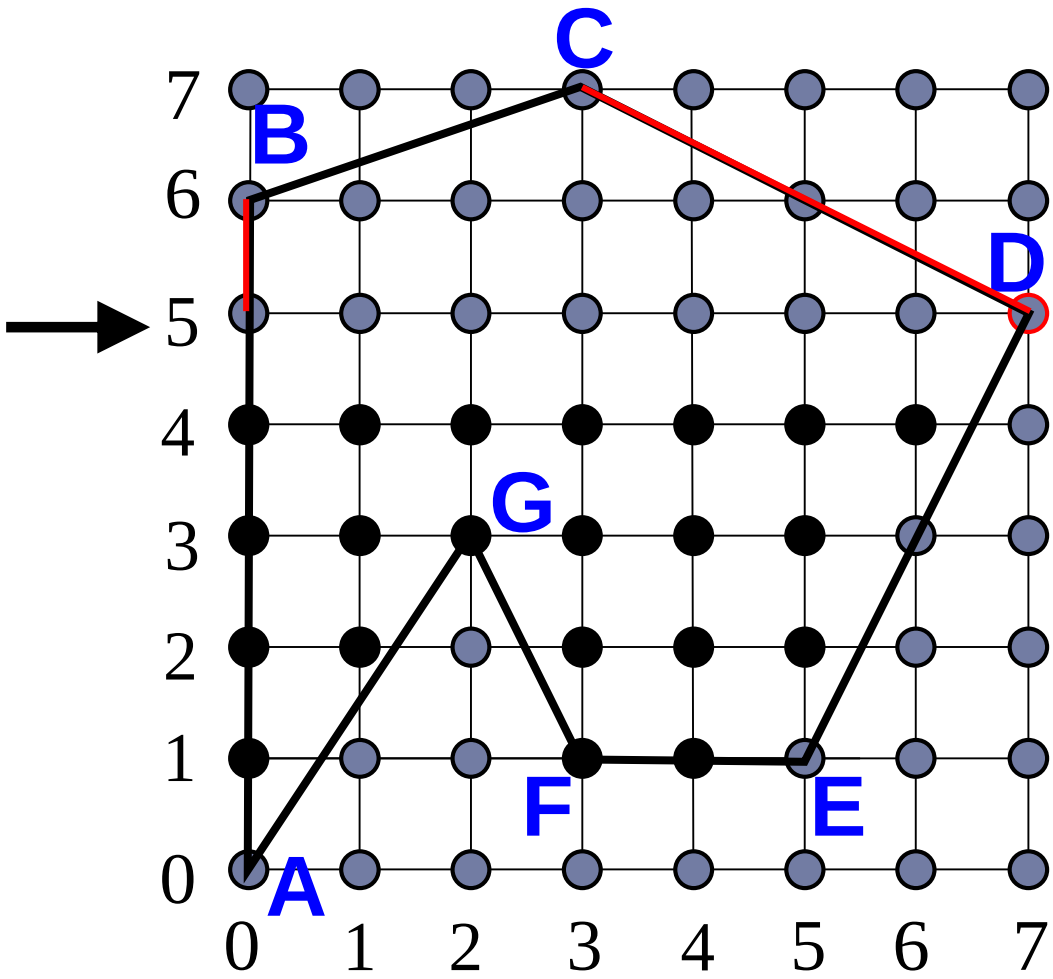
Active Edge List

	AB	ED	CD
maxY	6	5	7
currentX	0	7	7
xIncr	0	½	–2

General Polygons – Example

Active Edge Table



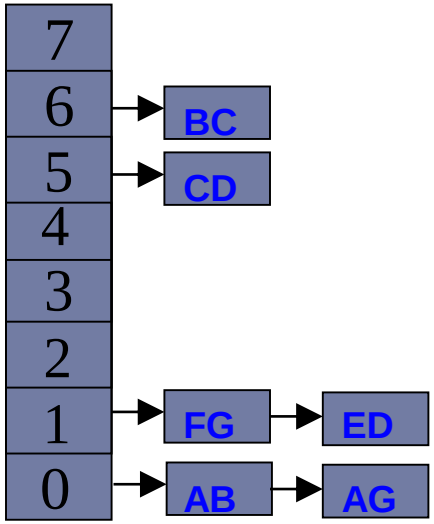


Active Edge List

	AB	ED	CD
maxY	6	5	7
currentX	0	6½	7
xIncr	0	½	−2

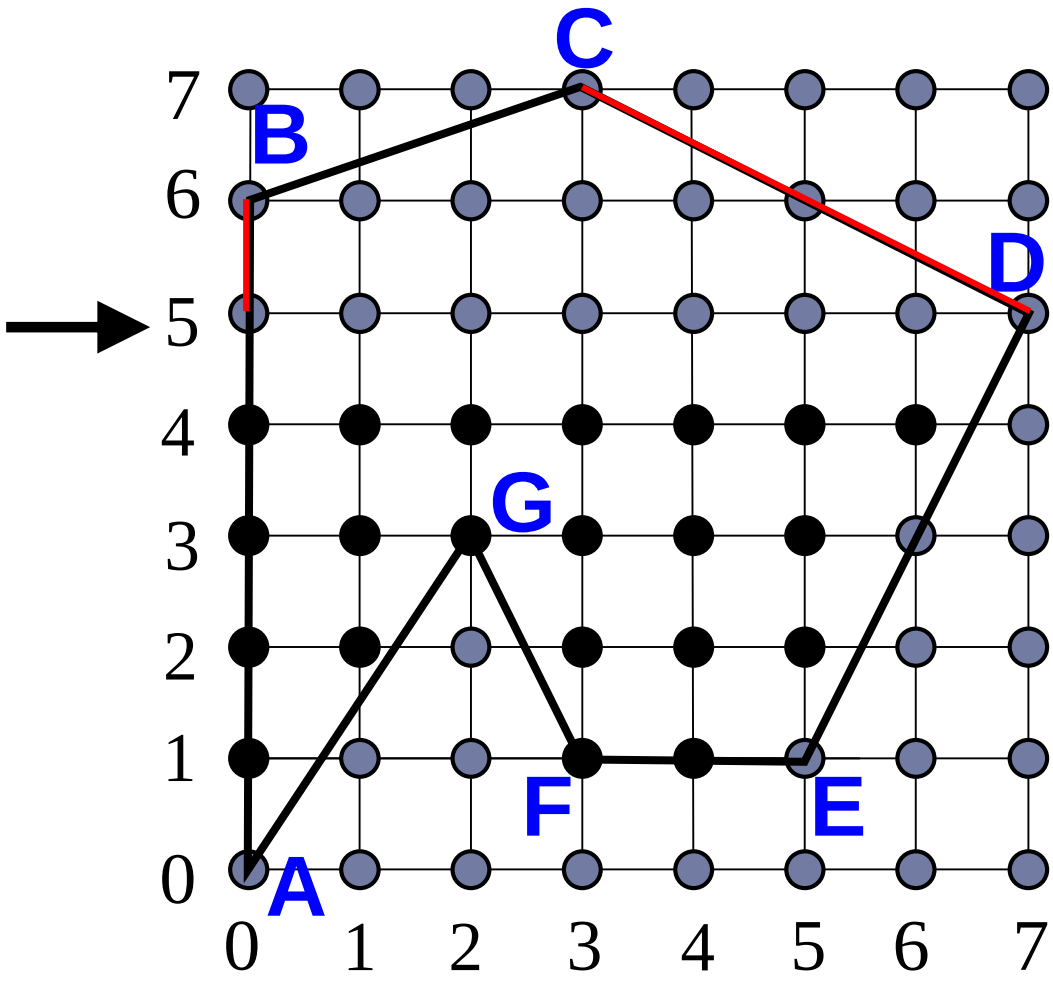
General Polygons – Example

Active Edge Table



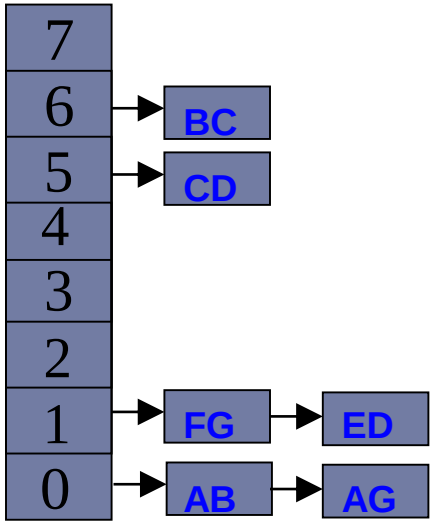
Active Edge List

	AB	CD
maxY	6	7
currentX	0	7
xIncr	0	–2



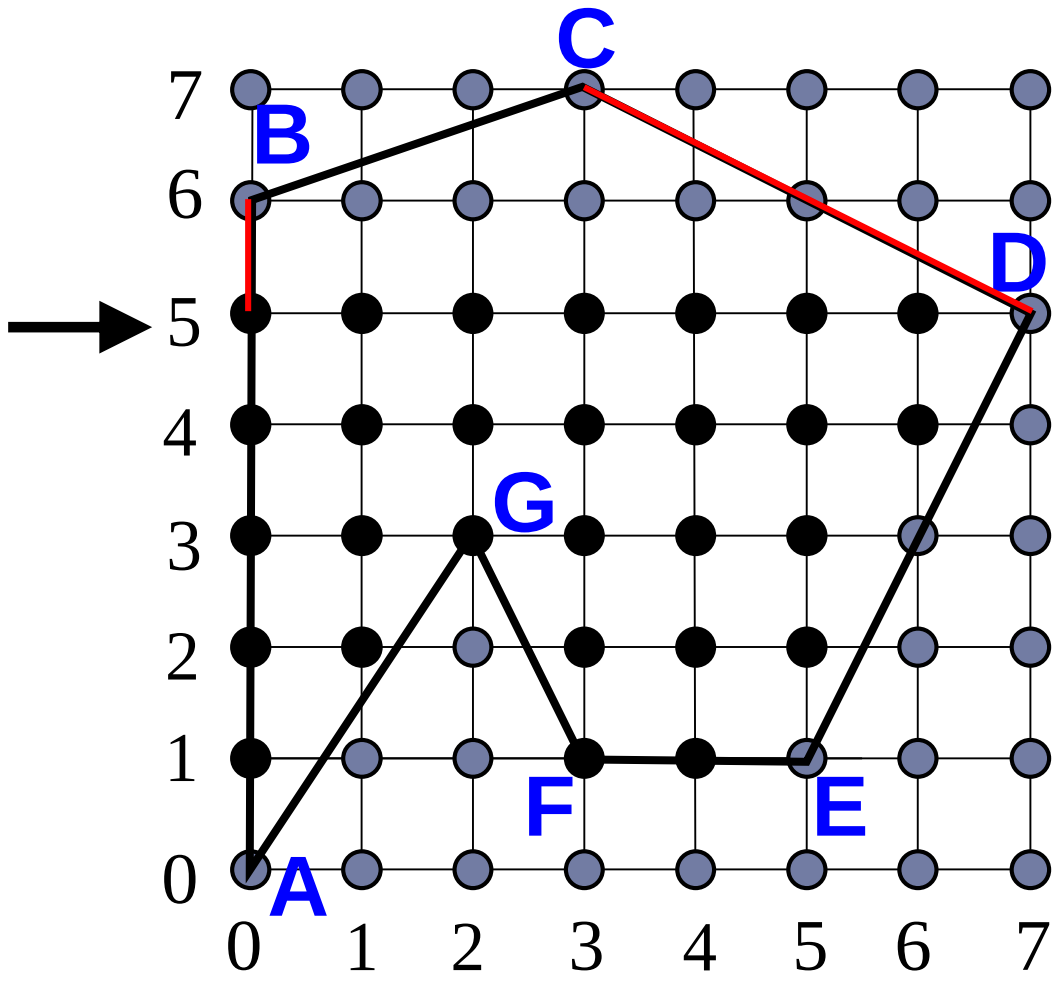
General Polygons – Example

Active Edge Table



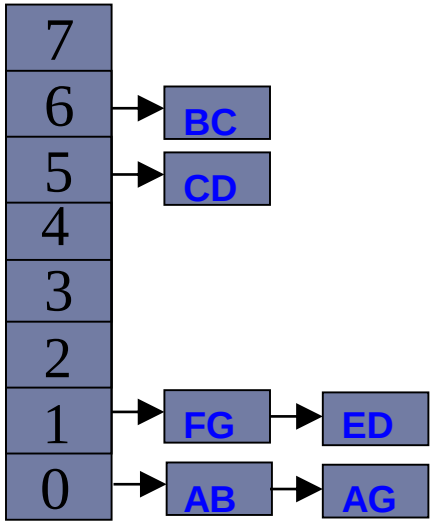
Active Edge List

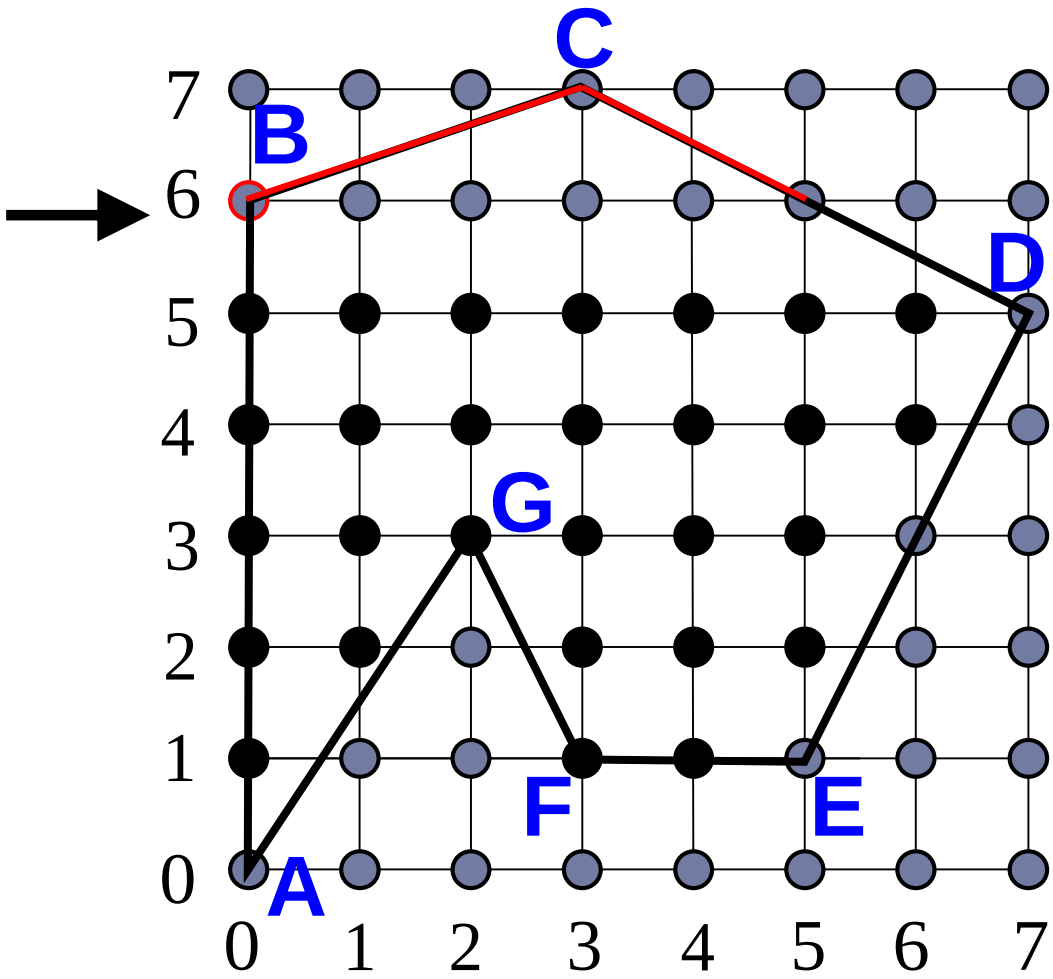
	AB	CD
maxY	6	7
currentX	0	7
xIncr	0	–2



General Polygons – Example

Active Edge Table



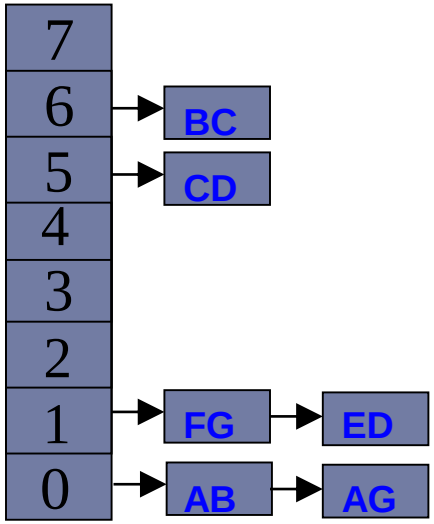


Active Edge List

	AB	BC	CD
maxY	6	7	7
currentX	0	0	5
xIncr	0	3	–2

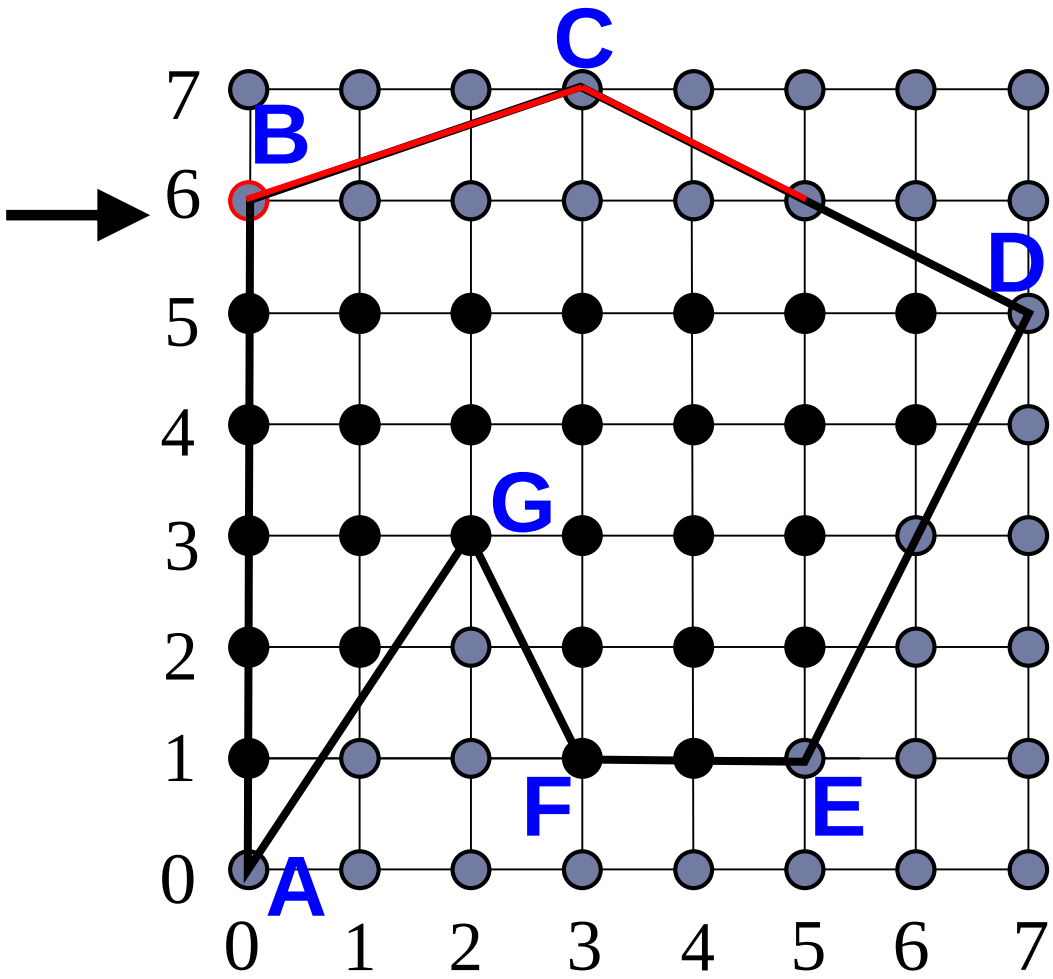
General Polygons – Example

Active Edge Table



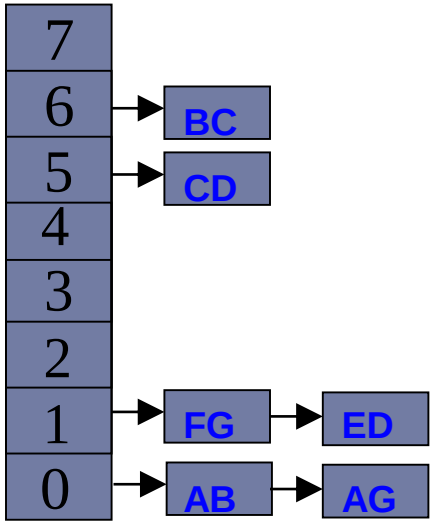
Active Edge List

	AB	BC	CD
maxY	6	7	7
currentX	0	0	5
xIncr	0	3	−2



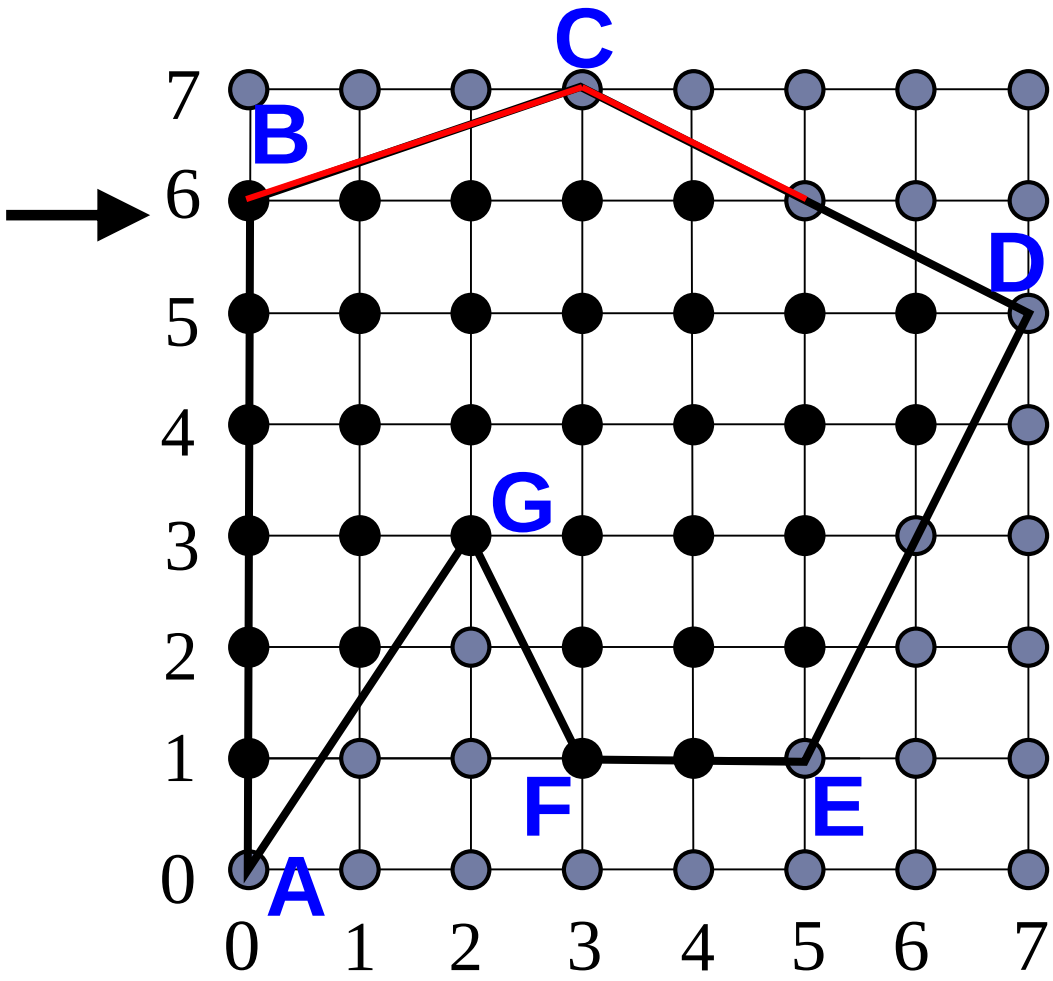
General Polygons – Example

Active Edge Table



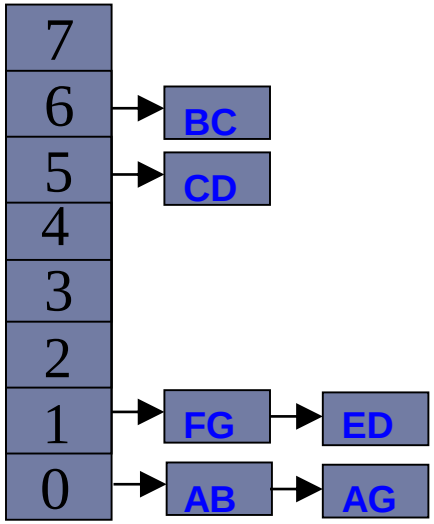
Active Edge List

	BC	CD
maxY	7	7
currentX	0	5
xIncr	3	–2

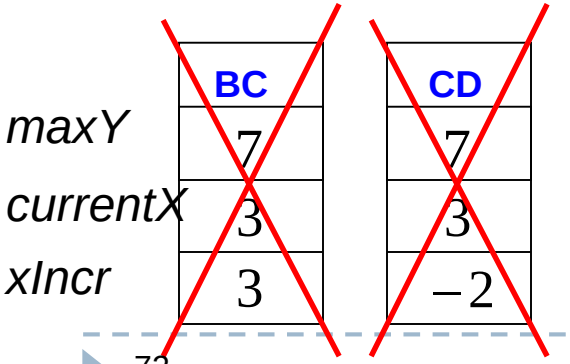


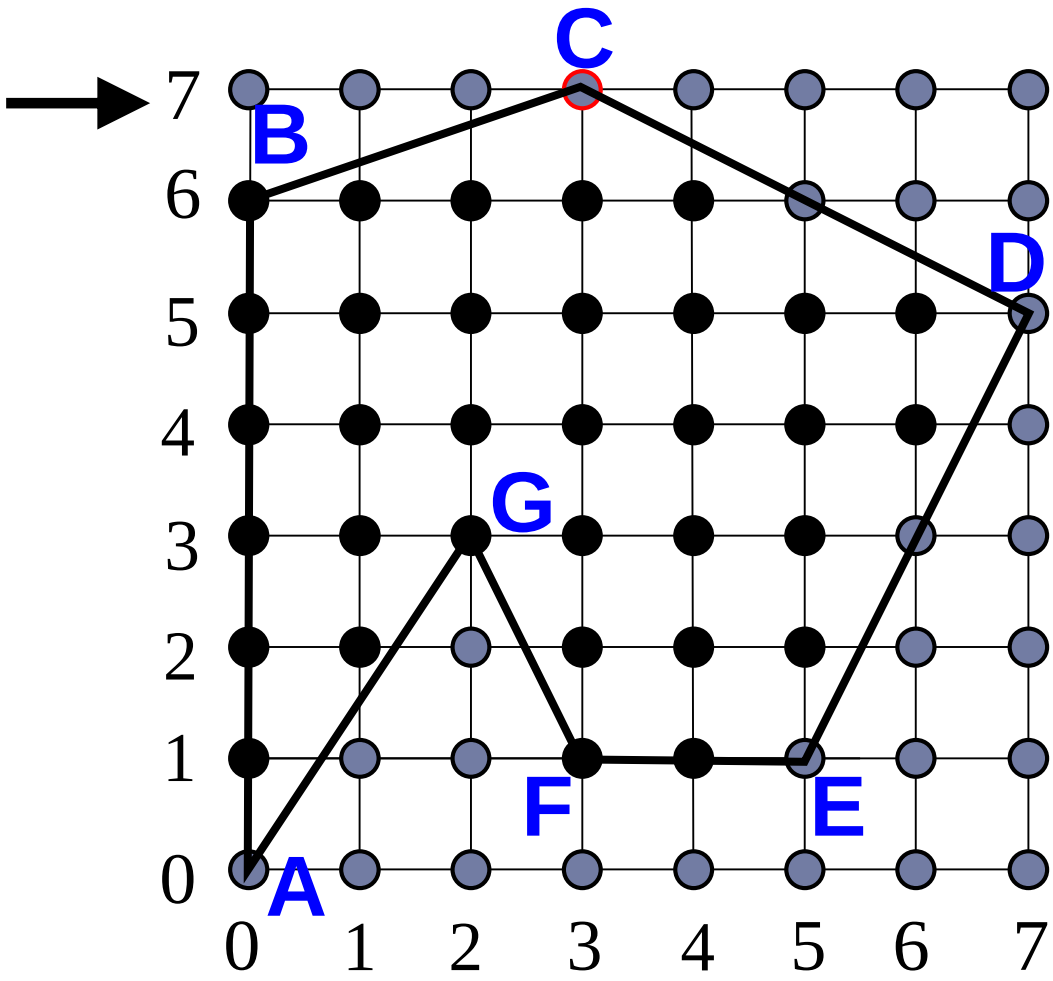
General Polygons – Example

Active Edge Table



Active Edge List





General Polygons – Algorithm

line = 0

While (*line* < height)

Add edges to Active Edge List from Active Edge Table starting at *line*

Remove edges that end at *line*

Fill pixels

Increment *x*-values on edges in Active Edge List

Increment *line*