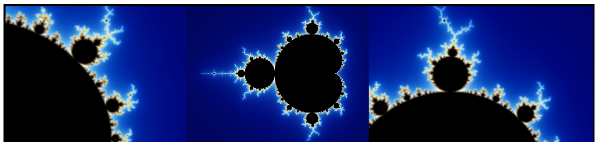
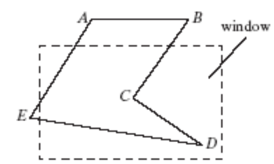


Clipping

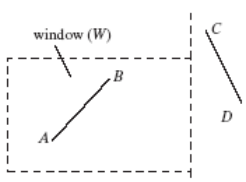
Clipping Lines

- We want to draw only the parts of lines that are inside the world window.
- To do this, we need to replace line portions outside the window by lines along the window boundaries.
- The process is called clipping the lines.



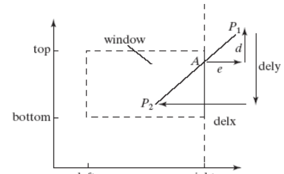
Clipping

- The method we will use is called Cohen-Sutherland line clipping.
- There are 2 trivial cases:
  - a line AB totally inside the window, which we draw all of
  - a line CD totally outside the window, which we do not draw at all.



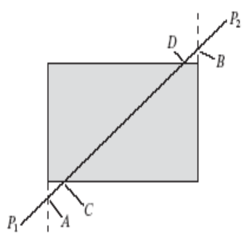
Clipping

- A line consists of 2 endpoints (vertices),
  - P_1 and P_2.
- If we do not have a trivial case, we must alter the vertices to the points where the line intersects the window boundaries (replace P_1 by A).



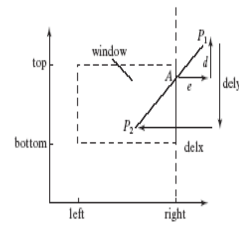
A Line Needing 4 Clips

- Clipping order:
  - Clip from the left, intersection A
    - x = W.l
  - Clip from the right, intersection B
    - x = W.r
  - Clip from the bottom, intersection C
    - y =W.b
  - Clip from the top, intersection D
    - y =W.t



Clipping

- In the diagram, d/dely = e/delx (similar triangles).
- Also,
  - delx = p_1.x – p_2.x,
  - dely = p_1.y – p_2.y and
  - e = p_1.x – W.r.
- Obviously, A.x = W.r.
- A.y=?



Cohen Sutherland Algorithm

- www.cs.mtsu.edu/~cen/private/lectures/ClippingAlg.html
- Determine if the line needs clipping
  - Compare TFTT of P1 and P2
- If it does, choose which of the 4 sides to clip against
  - Determine which of the two points need clipping
  - Determine which side does clipping occur
  - Test all 4 sides to make sure the line is fully clipped

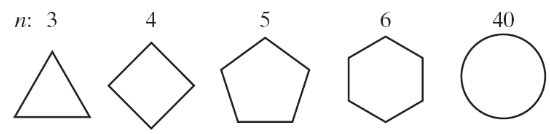
Clipping by hand…

- Given a window (50,120,0,100)
- Trace the algorithm to clip the following line segments:
  - P1 (50,40), p2 (100,20)
  - P1 (10,170), p2 (100,0)
  - P1 (20,-10), p2 (200,200)
- Give the new endpoints for any clipped segments

Drawing Regular Polygons, Circles, and Arcs

- A polygon is **simple** if no two of its edges cross each other. More precisely, only adjacent edges can touch and only at their shared endpoint.
- A polygon is **regular** if it is simple, if all its sides have equal length, and if adjacent sides meet at equal interior angles.
- We give the name ***n*-gon** to a regular polygon having *n* sides; familiar examples are the 4-gon (a square), an 8-gon (a regular octagon) and so on.

Regular Polygons



Drawing Circles and Arcs

- Two methods:
  - **The center is given, along with a point on the circle.**
    - Here drawCircle(IntPoint center, int radius) can be used as soon as the radius is known.
    - If *c* is the center and *p* is the given point on the circle, the radius is simply the distance from *c* to *p*, found using the usual Pythagorean Theorem.

Drawing Circles and Arcs

- **Three points are given through which the circle must pass.**
  - It is known that a unique circle passes through any three points that don't lie in a straight line.
  - Finding the center and radius of this circle is discussed in Chapter 4.

Parametric Curves

- Three forms of equation for a given curve:
  - Explicit: e.g., $y = m*x + b$
  - Implicit: $F(x, y) = 0$; e.g., $y - m*x - b = 0$
  - Parametric: $x = x(t)$, $y = y(t)$

  t is a parameter; frequently, $0 \le t \le 1$.

  Example: Line:

  $F(x, y) = A_x + (B_x - A_x) * t$
  - A and B are 2D points with x and y values. A and B are on the straight line
  - The parametric form is

    $x(t) = A_x + (B_x - A_x)*t$

    $y(t) = A_y + (B_y - A_y)*t$

Specific Parametric Forms

- line:

  $x(t) = A_x + (B_x - A_x)*t$, $t>=0$

  $y(t) = A_y + (B_y - A_y)*t$
- circle:

  $x(t) = r*\cos(t)$, $0 <= t <= 2\pi$

  $y(t) = r*\sin(t)$
- ellipse:

  $x(t) = W*\cos(t)$, $0 <= t <= 2\pi$

  $y(t) = H*\sin(t)$

  - W and H are half-width and half-height.

Finding Implicit Form from Parametric Form

- Combine the x(t) and y(t) equations to eliminate t.
- Example:

  ellipse:

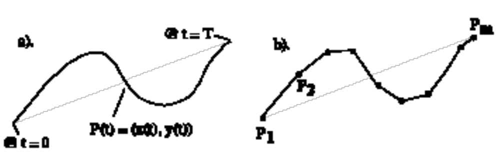
  $x = W*\cos(t)$

  $y = H*\sin(t)$
  - $x^2 = W^2\cos^2(t)$, $y^2 = H^2\sin^2(t)$
  - Dividing by the W or H factors and adding gives

    $(x/W)^2 + (y/H)^2 = 1$, the implicit form.

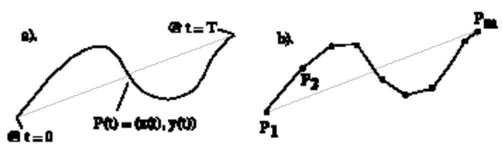
Drawing Parametric Curves

- For a curve *C* with the parametric form *P*(*t*) = (*x*(*t*), *y*(*t*)) as *t* varies from 0 to *T*, we use **samples** of *P*(*t*) at closely spaced instants.



Drawing Parametric Curves

- The position $P_i = P(t_i) = (x(t_i), y(t_i))$ is calculated for a sequence $\{t_i\}$ of times.
- The curve $P(t)$ is approximated by the polyline based on this sequence of points P_i.



Drawing Parametric Curves

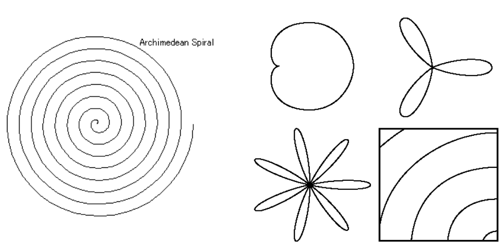
- Code:

```
// draw the curve (x(t), y(t)) using
// the array t[0],..,t[n-1] of sample times
glBegin(GL_LINES);
  for(int i = 0; i < n; i++)
    glVertex2f((x(t[i]), y(t[i]));
glEnd();
```

Middle Tennessee State University

Shapes as Parametric Curves

- For drawing purposes, parametric forms circumvent all of the difficulties of implicit and explicit forms.
- Cardioid, 2 rose curves, Archimedean spiral



Middle Tennessee State University

Polar Coordinates Parametric Form

- General form: $x = f(\theta)*\cos(\theta)$
  $y = f(\theta)*\sin\theta$
- cardioid: $f(\theta) = K*(1 + \cos(\theta))$, $0 \le \theta \le 2\pi$
- rose: $f(\theta) = K*\cos(n*\theta)$, $0 \le \theta \le 2n\pi$, where n is number of petals (n odd) or twice the number of petals (n even)
- spirals: Archimedean, $f(\theta) = K\theta$
  Logarithmic, $f(\theta) = Ke^{a\theta}$
- K is a scale factor for the curves.

Middle Tennessee State University

Polar coordinates Parametric Form

– conic sections (ellipse, hyperbola, circle, parabola): $f(\theta) = (1 \pm e \cos \theta)^{-1}$

- e is eccentricity:
  - 1 : parabola;
  - 0 : circle;
  - between 0 and 1, ellipse;
  - greater than 1, hyperbola

Example for Ellipse