Slope Method:

1. Compute dx: $dx = x_2 - x_1$
2. Compute dy: $dy = y_2 - y_1$
3. Compute m: $m = {}^{dy}/_{dx}$
4. Compute b: $b = y_1 - m * x_1$
5. Set (x, y) equal to the lower left hand endpoint and set x_{end} equal to the largest value of x. If dx < 0, then $x = x_2, y = y_2 \; and \; x_{end} = x_1$. If dx > 0, then $x = x_1, y = y_1 \; and \; x_{end} = x_2$.
6. Test to determine whether the entire line has been drawn. If $x > x_{end}$, stop.
7. Plot a point at the current (x, y) coordinates.
8. Increment x: x = x+1
9. Compute the next value of y from the equation, y = mx + b
10. Go to step 6.

DDA Algorithm:

1. Starting point (x_1, y_1) and ending point (x_2, y_2)
2. Let (x_i, y_i) be any point on the line
3. Slope, m = ${}^{dy}/_{dx}$ where, $dy = y_{i+1} - y_i$ and $dx = x_{i+1} - x_i$
4. From above equation, we get

   $$m = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

   ⇨ $y_{i+1} = y_i + mdx$ or, $x_{i+1} = x_i + {}^{dy}/_{m}$
5. If |m| <= 1, $x = x_1$, $y = y_1$ and set dx = 1

   That is, $x_{i+1} = x_i + 1$ and $y_{i+1} = y_i + m$

   Else, |m|>1, $x = x_1$, $y = y_1$ and set dy = 1

   That is, $x_{i+1} = x_i + {}^{1}/_{m}$ and $y_{i+1} = y_i + 1$
6. Continue until x reaches x_2 for |m| <= 1 case or, y reaches y_2 for |m|>1 case

LAB 1 Assignment: Draw the following shape using DDA Algorithm.

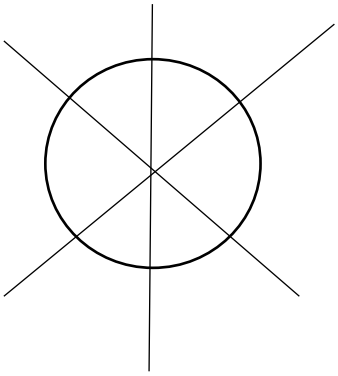
Bresenham's Circle Algorithm:

1. Take center and take radius of the circle
2. Set initial point (x, y) at (0, r) where r being the radius
3. Compute d: d = 3 – 2r
4. Test to determine whether the entire circle is drawn. If x > y, then stop
5. Plot (x, y)
6. If d < 0, then d = d+4x+6, x = x+1

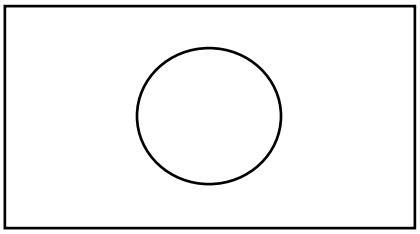
   Else d = d+4(x-y)+10, x = x+1, y = y-1
7. Go to step 4

Points ➔

```
glBegin(GL_POINTS);
glVertex2i(xc+x,yc+y);
glVertex2i(xc+y,yc+x);
glVertex2i(xc-y,yc+x);
glVertex2i(xc-x,yc+y);
glVertex2i(xc-x,yc-y);
glVertex2i(xc-y,yc-x);
glVertex2i(xc+y,yc-x);
glVertex2i(xc+x,yc-y);
glEnd();
```



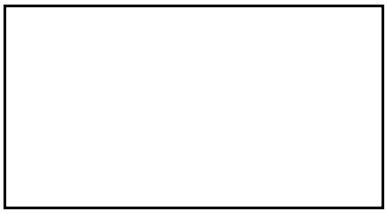
LAB 2 Assignment: Draw the following shape using DDA and Bresenham's circle Algorithm.



Bresenham's Line Algorithm: (for 0<m<1)

1. Compute the initial values:
   dx = x_2 - x_1, dy = y_2 - y_1, Inc_1= 2dy, d = $Inc_1 - dx$, Inc_2= 2(dy – dx)
2. Set (x, y) equal to the lower left-hand endpoint and x_{end} equal to the largest value of x. If dx < 0, then x = x_2, y = y_2, x_{end} = x_1. If dx > 0, then x = x_1, y = y_1, x_{end} = x_2.
3. Plot a point at the current (x, y) coordinates.
4. Test to see whether the entire line has been drawn. If x = x_{end}, stop.
5. Compute the location of the next pixel. If d < 0, then d = d + Inc_1. If d ≥ 0, then d = d + Inc_2 and then y = y + 1.
6. Increment x: x = x +1.
7. Plot a point at the current (x, y) coordinates.
8. Go to step 4.

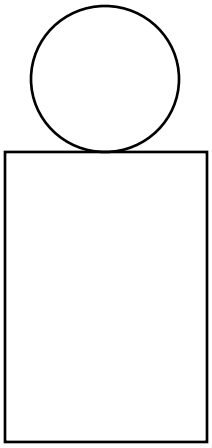
LAB 3 Assignment: Draw the following shape using Bresenham's Line Algorithm.



Midpoint Circle Algorithm:

1. Take center and radius of the circle
2. Set initial point (x, y) at (0, r) where r being the radius
3. Compute p: p = 1 – r
4. Test to determine whether the entire circle is drawn. If x > y, then stop
5. Plot (x, y)
6. If p < 0, then p = p+2x+3, x = x+1
   Else p = p+2(x-y)+5, x = x+1, y = y-1
7. Go to step 4

LAB 4 Assignment: Draw the following shape using Bresenham's Line and Midpoint Circle Algorithm.



Midpoint Ellipse Algorithm:

1. Take the center (h, k), major axis (a) and minor axis (b) as input
2. Initialize x = 0, y = b
3. Compute the following values:

   aa = a*a, bb = b*b, aa2 = aa*2, bb2 = bb*2

   fx = 0, fy = aa2*b

   p = bb – aa*b + 0.25*aa
4. Test whether fx < fy. If no, then stop. [where slope < 1]
5. Plot (x, y)
6. Increment x: x= x + 1

   Update fx: fx = fx + bb2
7. If p < 0, then p = p + fx + bb

   Else y = y – 1, fy = fy – aa2, p = p + fx +bb – fy
8. Go to 4
9. p =bb(x + 0.5)(x + 0.5) + aa(y - 1)(y - 1) – aa*bb
10. Test whether y > 0. If no, then stop.
11. Plot(x, y)
12. Decrement y: y = y – 1

    Update fy: fy = fy – aa2

Part 1

Part 2

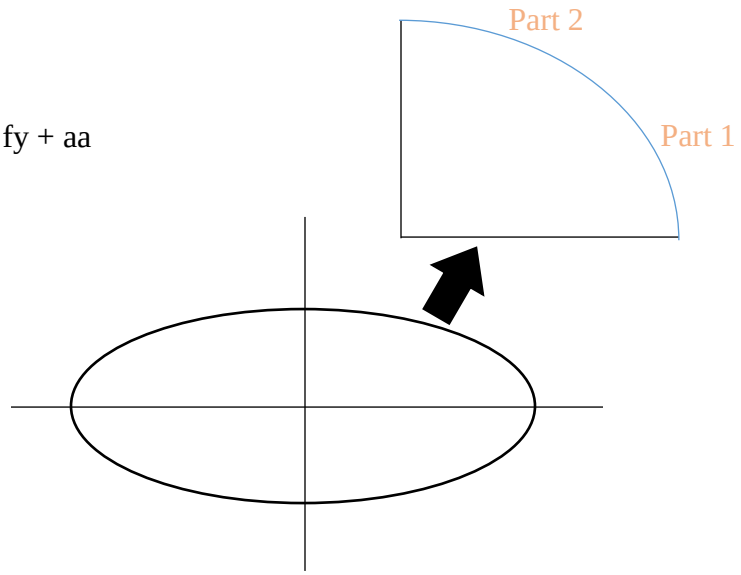
13. If $p \geq 0$, then $p = p - fy + aa$

    Else $x = x + 1$, $fx = fx + bb2$, $p = p + fx - fy + aa$

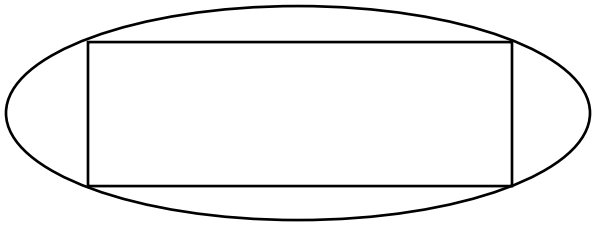
14. Go to 10

Points ➔

```
glBegin(GL_POINTS);
glVertex2i(h+x,k+y);
glVertex2i(h-x,k+y);
glVertex2i(h-x,k-y);
glVertex2i(h+x,k-y);
glEnd();
```



LAB 5 Assignment: Draw the following shape using Midpoint Ellipse Algorithm.



Constraints:

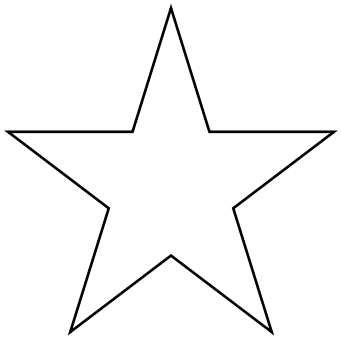
i) Four corner points of the rectangle should touch the circumference of the enclosing ellipse.
ii) Rectangle should be placed at the middle portion of the ellipse
iii) You can draw the rectangle with an algorithm of your own choosing

C Curve:

```
C_curve(float x, y, len, alpha, int n)
{
        If (n > 0)
        {
                len = len/sqrt(2.0);
                C_curve(x, y, len, alpha + 45, n - 1);
                x = x + len*cos(alpha + 45);
                y = y + len*sin(alpha + 45);
                C_curve(x, y, len. alpha – 45. n - 1);
        }
```

```
        else
            line(x, y, x + len*cos(alpha), y + len*sin(alpha));
}
```

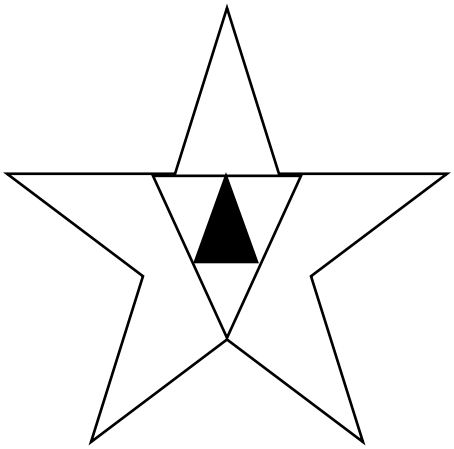
LAB 6 Assignment: Draw the following shape using C curve.



Sierpinski Gasket:

```
S_Gasket(float x1, y1, x2, y2, x3, y3, int n)
{
    float x12, y12, x13, y13, x23, y23;
    if (n > 0)
    {
        x12 = (x1 + x2) / 2;
        y12 = (y1 + y2) / 2;
        x13 = (x1 + x3) / 2;
        y13 = (y1 + y3) / 2;
        x23 = (x2 + x3) / 2;
        y23 = (y2 + y3) / 2;
        S_Gasket(x1, y1, x12, y12, x13, y13, n - 1);
        S_Gasket(x12, y12, x2, y2, x23, y23, n - 1);
        S_Gasket(x13, y13, x23, y23, x3, y3, n - 1);
    }
    else
        triangle(x1, y1, x2, y2, x3, y3);
}
```

LAB 7 Assignment: Draw the following shape using C curve and sierpinski gasket.



Constraints:

i) The triangle inside the star should touch each other at the intersecting points / areas.