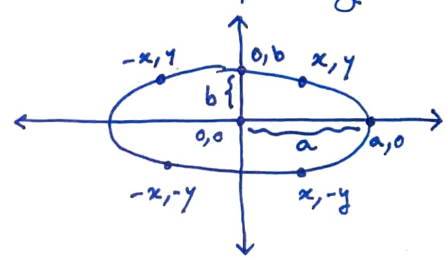
Q. What is a conic section?

→ In geometry, a conic section is a curve obtained as the intersection of a plane with the surface of a cone.

The three types of conic sections are ellipse, parabola and hyperbola.
The circle is just a special case of ellipse. [diagram]

Q. Describe the scan conversion procedure for an ellipse using midpoint analysis method.

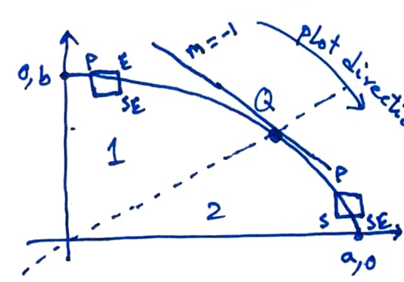
→ The equation an ellipse centered at the origin is given by $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$ where $2a$ and $2b$ are the length of major axis and minor axis respectively.
We assume for now $a > b$.



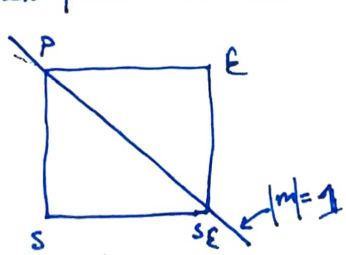
As shown in the figure, an ellipse shows a 4-way symmetry, thus we only need to generate the points for the 1st quadrant, i.e. from $(0,b)$ to $(a,0)$.

Observe that at the begining the curve has gradually increasing x-coordinate ~~and~~ while y-coordinate slowly decreases. ~~Whereas~~ Whereas towards the end of the curve the scenario changes, y-coordinate decreases rapidly while x-coordinate slowly increases.

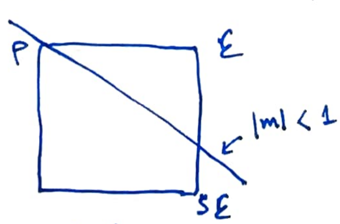
Therefore, in the begining, ~~the~~ we have to choose between E and SE as the next point, while towards the end the choice becomes S or SE.
So there must exists a point Q on the curve where the choice switches from E vs SE to S vs SE.



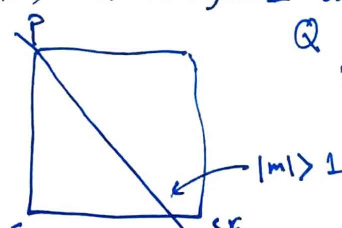
Q must be that point where the tangent has a slope $m = -1$ or $|m| = 1$, since at Q the next point must be exactly SE. We define from $(0,b)$ to Q as region 1 and from Q to $(a,0)$ as region 2.



at Q.



before Q (region 1)



after Q (region 2)

let us rewrite the eqn of the ellipse as $b^2x^2 + a^2y^2 - a^2b^2 = 0$ ... (1)

Let us define, $f(x,y) = b^2x^2 + a^2y^2 - a^2b^2$

On differentiating eqn (1) w.r.t. x we obtain,

$$2b^2x + 2a^2y\frac{dy}{dx} = 0 \quad \Rightarrow \quad \frac{dy}{dx} = -2b^2x/2a^2y = -b^2x/a^2y$$

At point Q ~~at~~ we have $|m| = 1$

$$\therefore \left|\frac{dy}{dx}\right| = 1 \quad \Rightarrow \quad \left|-b^2x/a^2y\right| = 1$$

$$\Rightarrow b^2x/a^2y = 1$$

[since $x, y \geq 0$ and a^2, b^2 always +ve we can remove the modulus operator by discarding the -ve sign]

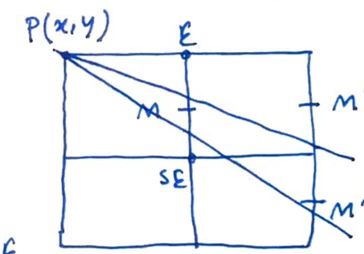
$$\Rightarrow b^2x = a^2y$$

During region 1 (before Q) ~~at~~ we have $|m| < 1$ or $b^2x < a^2y$

and in region 2 (after Q) ...... $|m| > 1$ or $b^2x > a^2y$

Thus to generate points in region 1, we start at $(0, b)$ and continue while $b^2x < a^2y$.

Since in region 1 the choice of next point is between E & SE we define our decision parameter $d = f(M)$ where M is the middle point of line segment joining E & SE points.



If $d = 0$ then M is on the ellipse we can choose either E or SE.

if $d < 0$ then M is inside the ellipse & we choose E as the next point

and if $d > 0$ .... outside .... SE .....

$\boxed{\text{similar to circle}}$

Case East

When next point ~~is~~ choosen is E, the next decision will be taken with $f(M')$

Thus ~~is~~ we calculate

$$\Delta d_E = f(M') - f(M) = f(x+2, y-1/2) - f(x+1, y-1/2)$$

$$= b^2(x+2)^2 + a^2(y-1/2)^2 - a^2b^2 - [b^2(x+1)^2 + a^2(y-1/2)^2 - a^2b^2]$$

$$= b^2[(x+2)^2 - (x+1)^2] = b^2(2x+3)$$

Case South East

Similarly $\Delta d_{SE} = f(M'') - f(M) = f(x+2, y-3/2) - f(x+1, y-1/2)$

$$\vdots$$

$$= b^2\left[(x+2)^2 - (x+1)^2\right] + a^2\left[(y-3/2)^2 - (y-1/2)^2\right]$$

$$= b^2.(2x+3).1 + a^2.(2y-2).(-1)$$

$$= b^2(2x+3) + a^2(2-2y)$$

Since we start at $(0,b)$

$$d_{start} = f(0+1, b-1/2) = b^2.1^2 + a^2(b-1/2)^2 - a^2b^2$$

$$= b^2 + a^2b^2 - a^2b + a^2/4 - a^2b^2$$

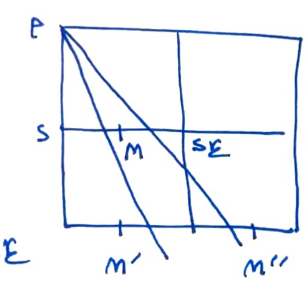
$$= b^2 - a^2b + a^2/4$$

To avoid floating point calculations we simply round-off d_{start} to the nearest integer.

NB We assume a, b all integers. When a is even $a^2/4$ is automatically a ~~perfect integ~~ whole number. When a is odd, we ~~might~~ have a fractional part equal to either $1/4$ or $3/4$ which can be unambiguosly rounded-off to 0 or 1 respectively. This rounding-off does not affect the decision as we had ~~argued~~ argued in the case of a circle.

Now for region 2, we ~~then~~ choose the next point either S or SE.

We similarly define our decision parmeter $d = f(M)$ where M is the midpoint of the S-SE line segment.



Here we have, if $d = 0$ then M is on the ellipse

we can choose either one of S and SE.

if $d < 0$ then M is inside the ellipse & we choose SE

and if $d > 0$ . . . . outside . . . . S

Case South

Here we have $\Delta d_S = f(M') - f(M) = f(x+1/2, y-2) - f(x+1/2, y-1)$

$$\vdots$$

$$= a^2\left[(y-2)^2 - (y-1)^2\right] = a^2(2y-3).(-1) = a^2(3-2y)$$

Case South East

Here we have $\Delta d_{SE} = f(M'') - f(M) = f(x+3/2, y-2) - f(x+1/2, y-1)$

$$\vdots$$

$$= b^2\left[(x+3/2)^2 - (x+1/2)^2\right] + a^2\left[(y-2)^2 - (y-1)^2\right] = b^2(2x+2) + a^2(3-2y)$$

In region 2 we start at the last (x, y) point generated in region 1.

Thus $d_{start} = f(x+1/2, y-1) = b^2(x+1/2)^2 + a^2(y-1)^2 - a^2b^2$

Here also we round-off d_{start} to avoid floating point calculations. We continue until y hits 0 (x-axis).

NB The argument for rounding-off doesn't affect the decision is similar to the argument made for region 1.

Therefore the algorithm for generating an ellipse is as follows:

```
midpoint_ellipse(xc, yc, a, b) {
    // region 1
    x = 0,  y = b
    d = round(b^2 - a^2 b + a^2/4)
    draw_pixel(x, y, xc, yc)
    while (b^2 x < a^2 y) {  // region 1
        if (d <= 0) {   // case East
            d += b^2(2x+3)
        } else {        // case SouthEast
            d += b^2(2x+3) + a^2(2-2y)
            y--
        }
        x++
        draw_pixel(x, y, xc, yc)
    }
    // region 2, start at last x, y
    d = round(b^2(x + 1/2)^2 + a^2(y-1)^2 - a^2 b^2)
    while (y > 0) {  // region 2
        if (d < 0) {  //case South East
            d += b^2(2x+2) + a^2(3-2y)
            x++
        } else {  // case South
            d += a^2(3-2y)
        }
        y--
        draw_pixel(x, y, xc, yc)
    }
}
```

```
draw_pixel(x, y, xc, yc) {
    putpixel(x+xc, y+yc)
    putpixel(-x+xc, y+yc)
    putpixel(-x+xc, -y+yc)
    putpixel(x+xc, -y+yc)
}
```

The arguments x_c, y_c denotes the center of the ellipse. The draw_pixel( ) method uses the 4-way symmetry to ~~gener~~ draw the points on all four quadrants. It also translates each point by (x_c, y_c) to draw the ellipse at the actual intended location.

The equality cases of our decision parameters $(d=0)$ is merged into the cases where we need less computation.

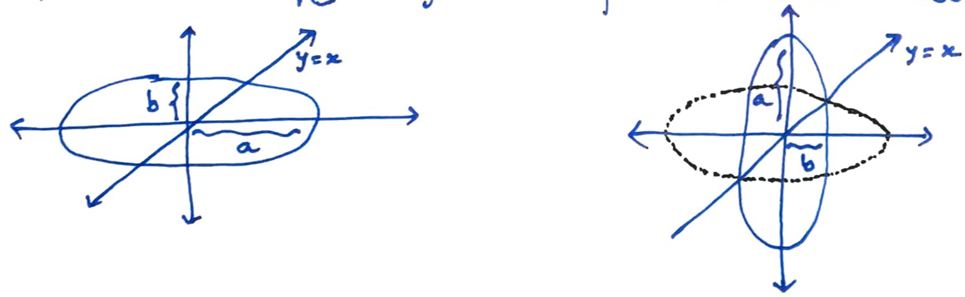
Lastly, our analysis assumed that $a > b$, we can remove this assumption using a reflection symmetry.

Thus If we have $a < b$ then we swap a with b and generate the points using the above algorithm, but draw the pixels by reflecting them about the $x=y$ line, which simply swaps the value of x with y.

Observe that, if we change the equation of the ellipse from

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad \text{to} \quad \frac{x^2}{b^2} + \frac{y^2}{a^2} = 1 \quad (\text{having } a > b)$$

the ellipses are simply reflections of each other about the $y=x$ line.



So, we modify our algorithm by adding a code section at the begining for comparing and swap a, b if required as well as set a flag.

If the flag is set we need to plot the (y, x) instead of (x, y).

[Writing the final algorithm is left as an exercise to the reader.]

NB We can optimize our algorithm by avoiding the multiplications inside the loops. Those multiplication expressions can be easily replaced by variables ~~which~~ whose values are updated incrementally.

[Writing this optimized algorithm is again left as an exercise].

NB Other conic sections can be similary scan converted [left as an exercise]