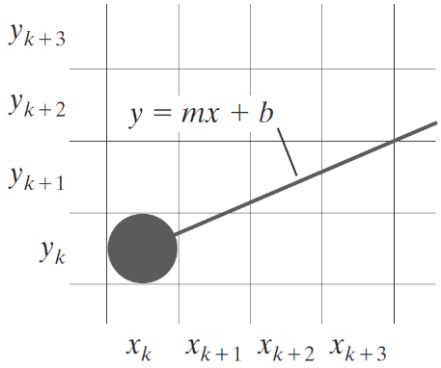
Bresenham's Line Algorithm

Bresenham's line algorithm is an accurate and **efficient raster line-generating algorithm**, developed by **Bresenham**, that uses only incremental **integer calculations**.

In addition, Bresenham's line algorithm can be adapted to display circles and other curves.



We summarize Bresenham line drawing for a line with a **positive slope less than 1** in the following outline of the algorithm. The constants $2\Delta y$ and $2\Delta y - 2\Delta x$ are calculated once for each line, so the arithmetic involves **only integer addition and subtraction** of these two constants.

An implementation of Bresenham line drawing for slopes in the range **0 < m < 1.0** is given in the following procedure. Endpoint pixel positions for the line are passed to this procedure, and pixels are plotted from the left endpoint to the right endpoint.

Bresenham's Line-Drawing Algorithm for |m| < 1.0

1. **Input** the two line endpoints and store the left endpoint in (x_0, y_0).
2. **Set** the color for frame-buffer position (x_0, y_0); i.e., plot the first point.
3. **Calculate** the constants Δx, Δy, $2\Delta y$, and $2\Delta y - 2\Delta x$, and obtain the starting value for the decision parameter as

$$p_0 = 2\Delta y - \Delta x$$

4. At each x_k along the line, starting at $k = 0$, **perform** the following test:

   If $p_k < 0$, the next point to plot is $(x_k + 1, y_k)$ and

$$p_{k+1} = p_k + 2\Delta y$$

   Otherwise, the next point to plot is $(x_k + 1, y_k + 1)$ and

$$p_{k+1} = p_k + 2\Delta y - 2\Delta x$$

5. **Repeat** step 4 $\Delta x - 1$ more times.

To illustrate the algorithm, we digitize the line with endpoints **(20, 10)** and **(30, 18)**.

This line has a slope of 0.8, with

$$\Delta x = 30 - 20 = 10$$
$$\Delta y = 18 - 10 = 8$$
$$slope\,(m) = \frac{\Delta y}{\Delta x} = 0.8$$

The initial decision parameter has the value

$$p_0 = 2\Delta y - \Delta x = 6$$

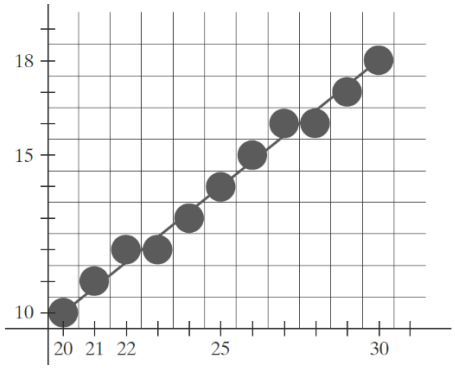
and the increments for calculating successive decision parameters are

$$2\Delta y = 16$$
$$2\Delta y - 2\Delta x = -4$$

We plot the initial point (x_0, y_0) = **(20, 10)**, and determine successive pixel positions along the line path from the decision parameter as follows:

k	p_k	(x_{k+1}, y_{k+1})
0	6	(21, 11)
1	2	(22, 12)
2	-2	(23, 12)
3	14	(24, 13)
4	10	(25, 14)
5	6	(26, 15)
6	2	(27, 16)
7	-2	(28, 16)
8	14	(29, 17)
9	10	(30, 18)

A plot of the pixels generated along this line path is shown in Figure below.



Write a program to implement the Bresenham's line algorithm.