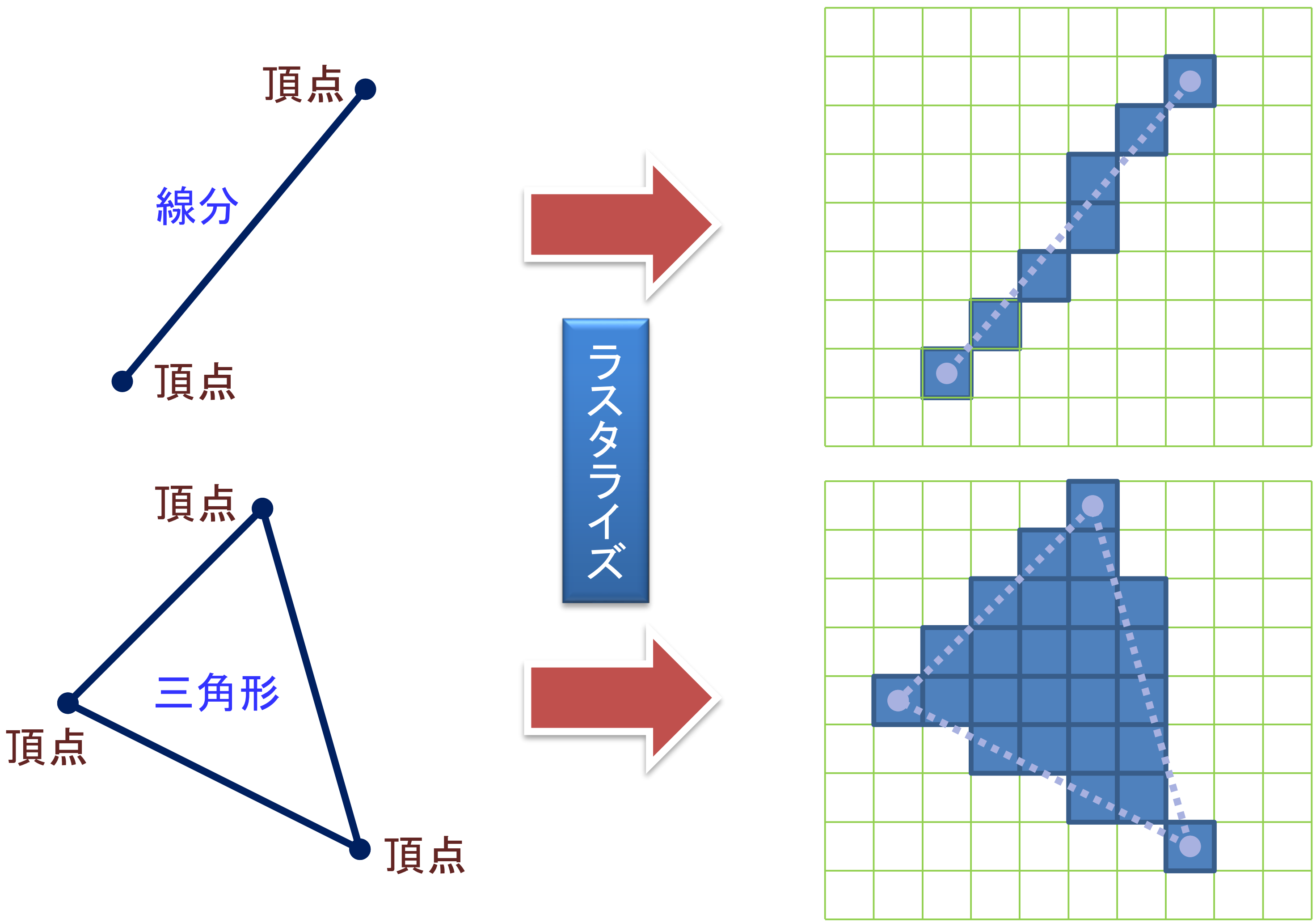
コンピュータグラフィックス

第3回：線を描く

デジタル画像の生成



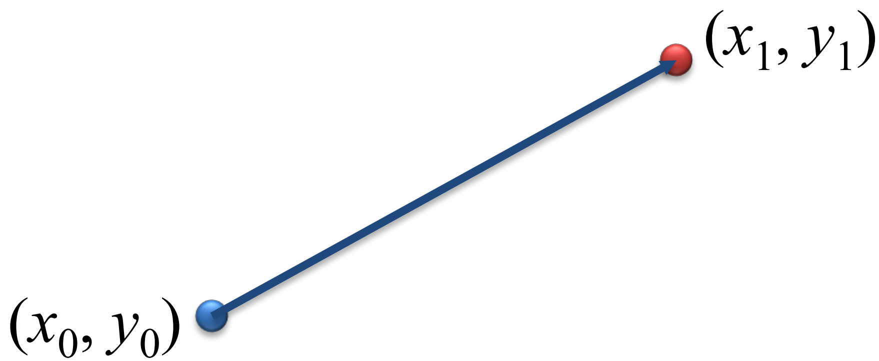
ベクトルデータ　　デジタル画像

走査変換（スキャンコンバージョン）

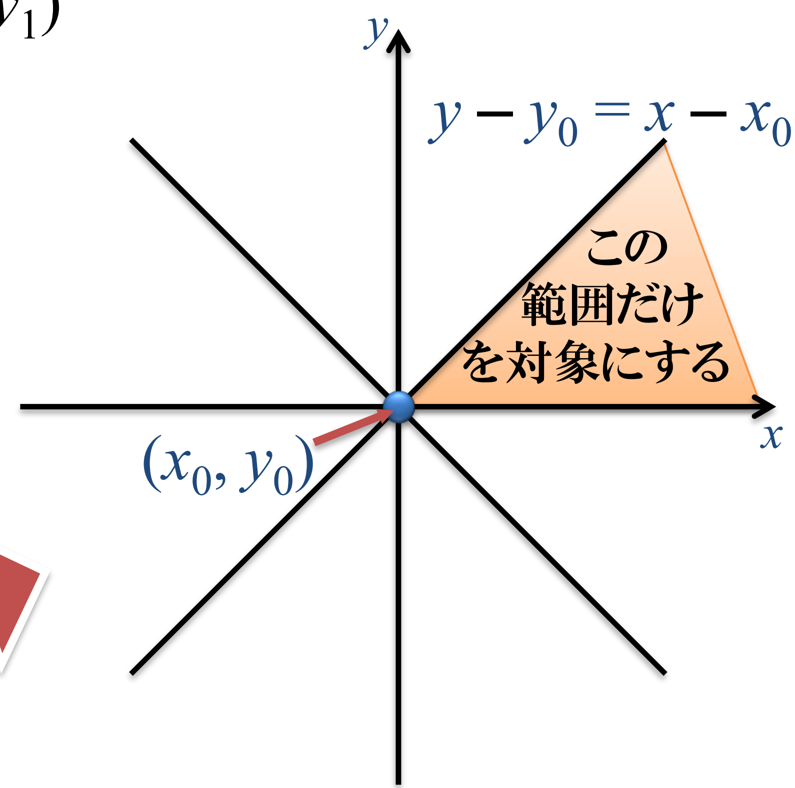
- ベクトルデータのデジタル画像化（ラスタライズ）
  - 線分を描く
  - 円を描く
  - 台形を塗りつぶす
  - 三角形を塗りつぶす
  - 任意の多角形を塗りつぶす

線分を描く

- **2点 (x_0, y_0), (x_1, y_1) を結ぶ線分の生成**

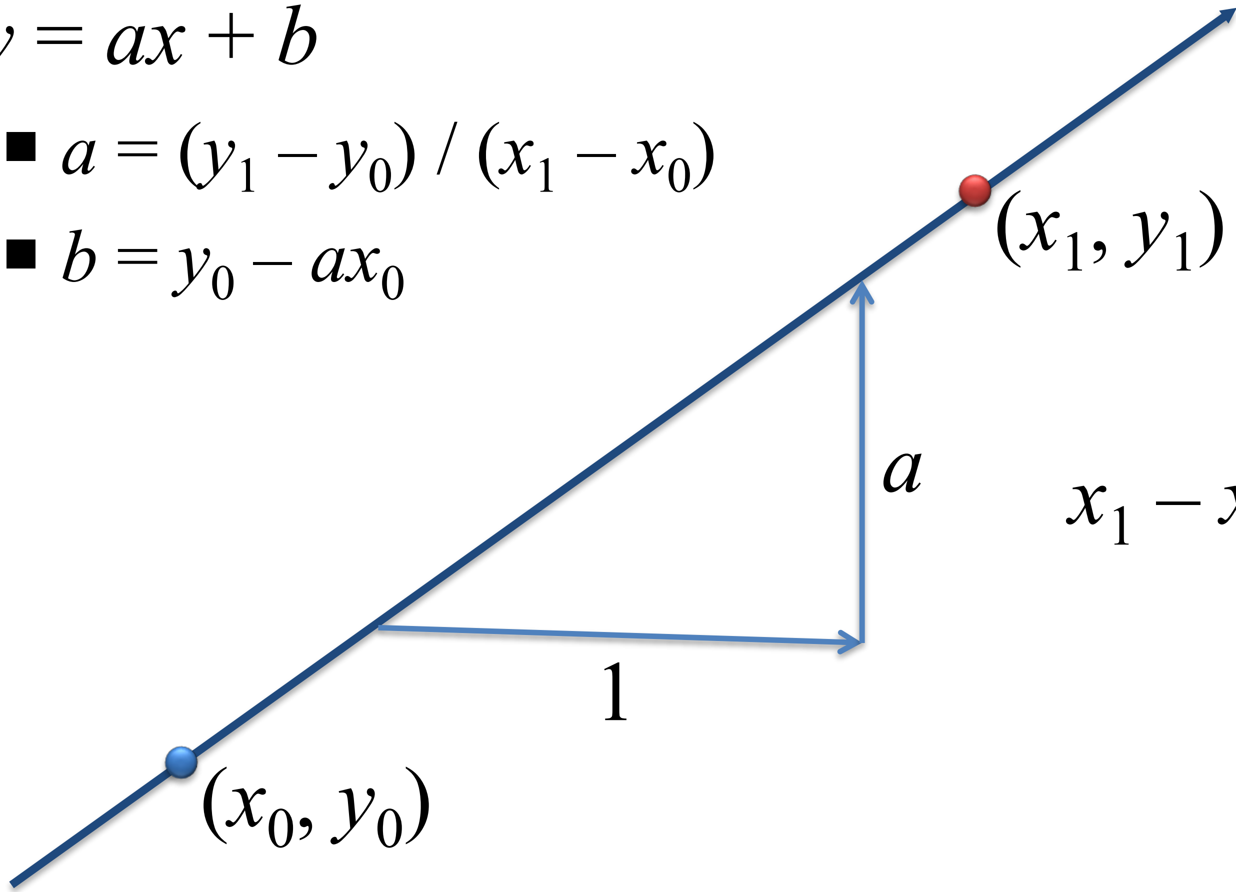


  - **条件**
    - $x_0 < x_1$
    - $y_0 < y_1$
    - $x_1 - x_0 > y_1 - y_0$



直線の方程式

- $y = ax + b$
  - $a = (y_1 - y_0) / (x_1 - x_0)$
  - $b = y_0 - ax_0$

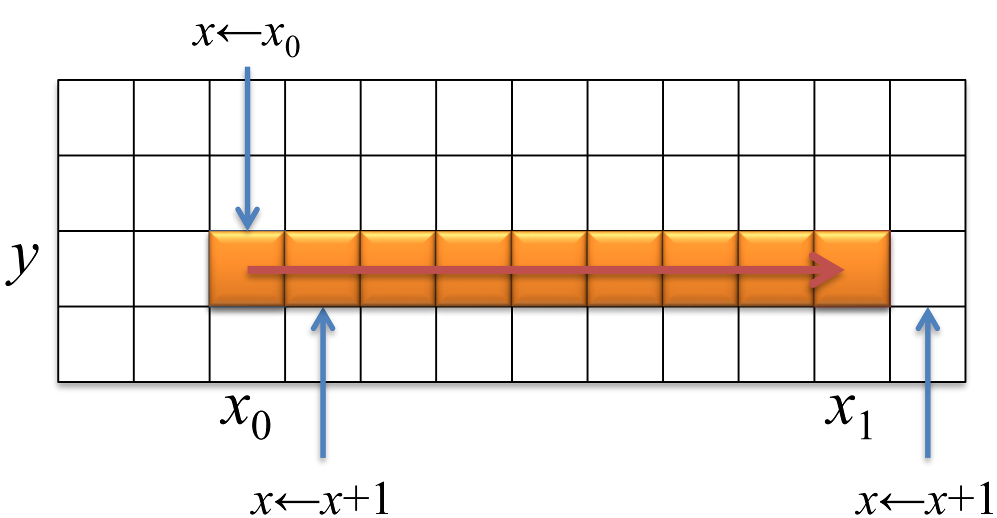


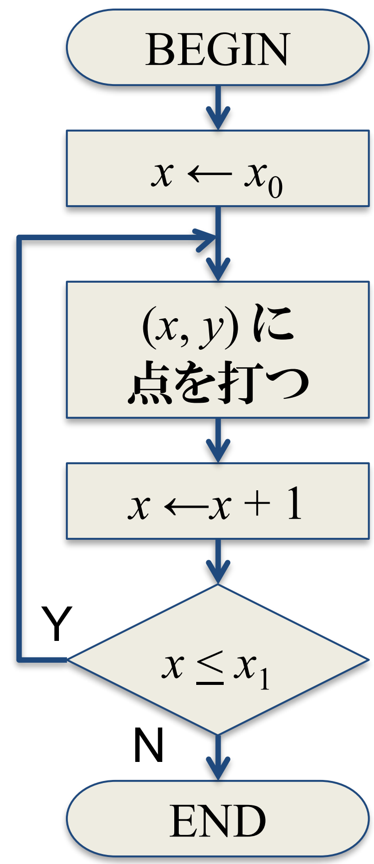
$$x_1 - x_0 > y_1 - y_0$$

⇓

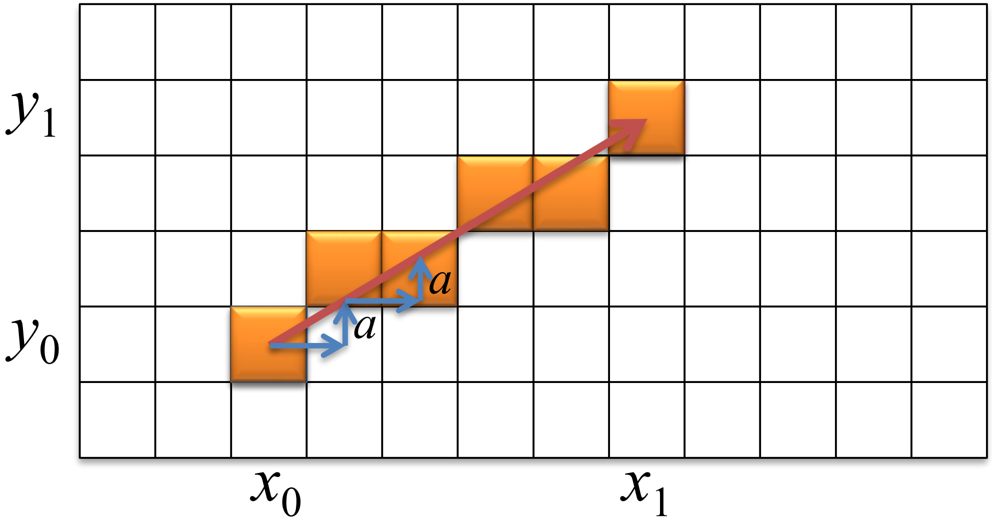
$$a < 1$$

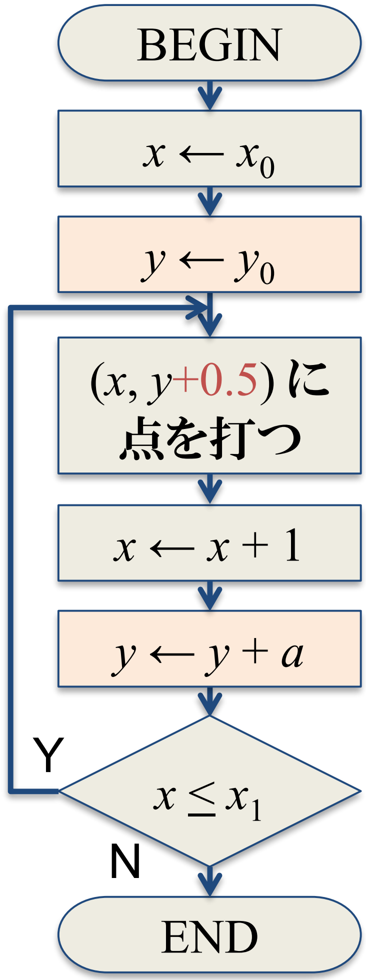
水平線を描く


$x \leftarrow x_0$
y
x_0
x_1
$x \leftarrow x+1$
$x \leftarrow x+1$

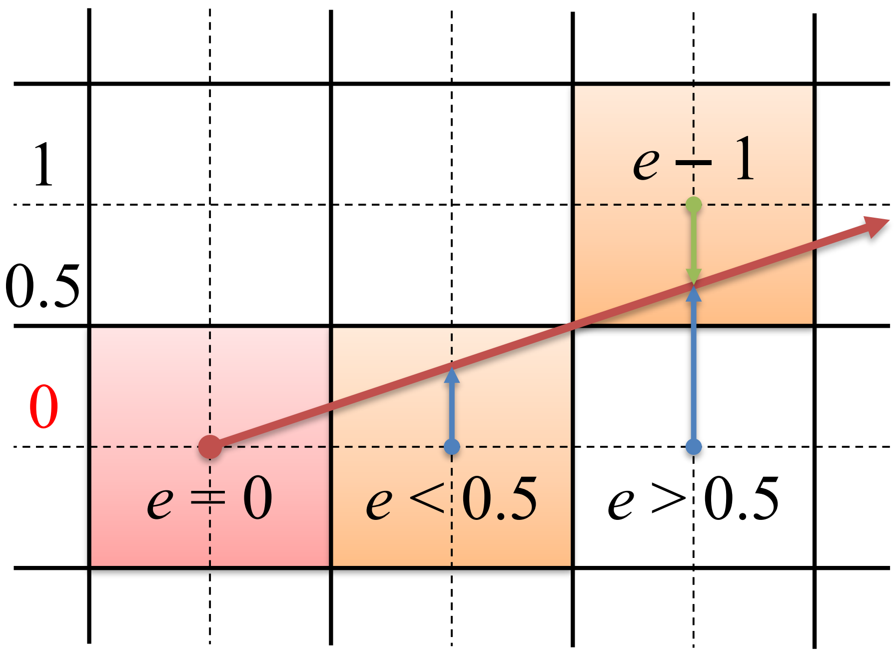

BEGIN
$x \leftarrow x_0$
(x, y) に
点を打つ
$x \leftarrow x + 1$
$x \leq x_1$
Y
N
END

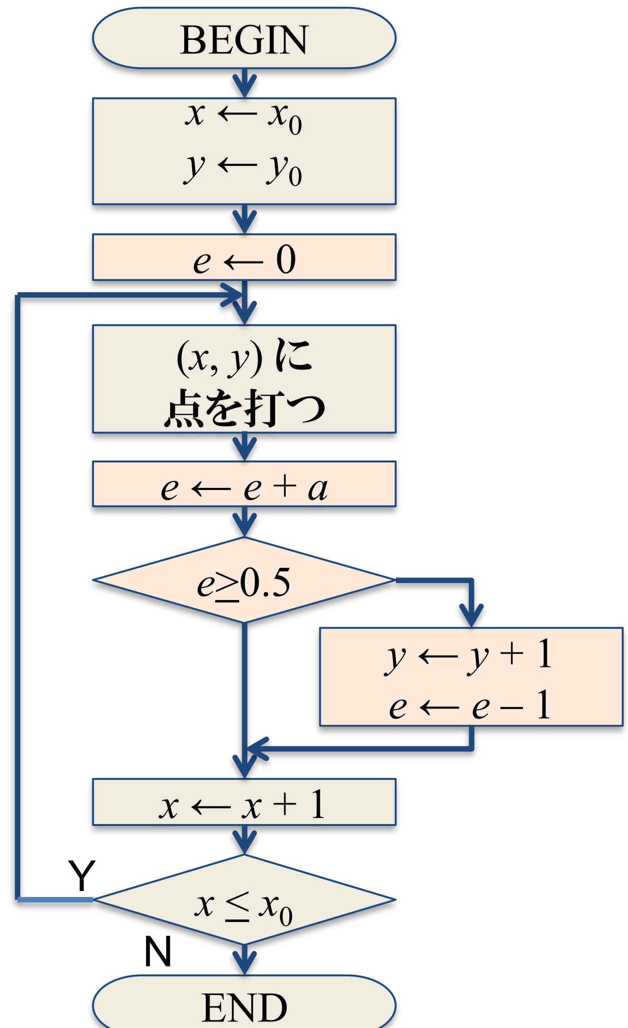
傾きを累積して斜線を描く



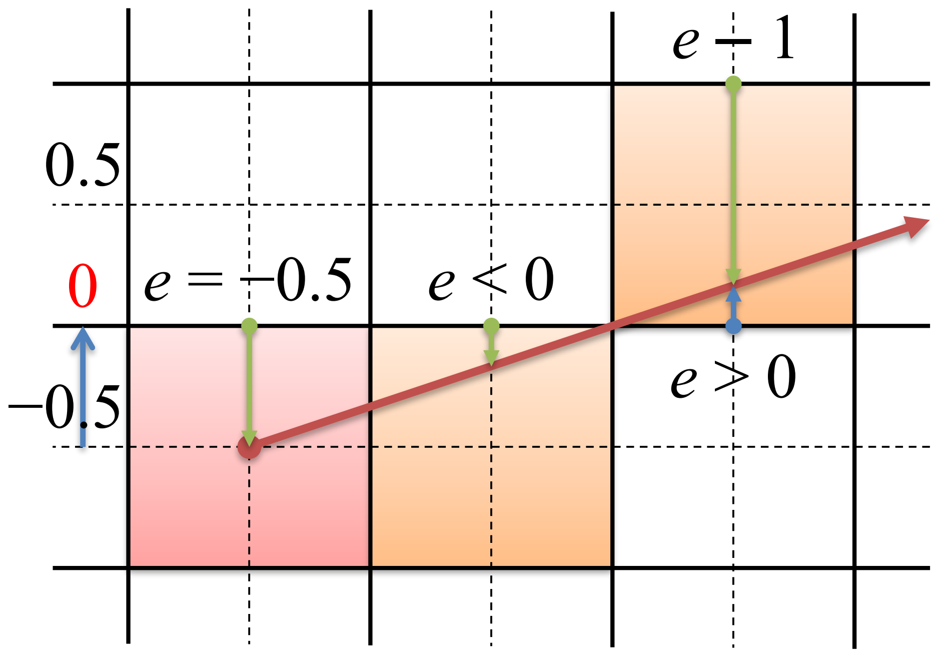


傾きの代わりに誤差を用いる

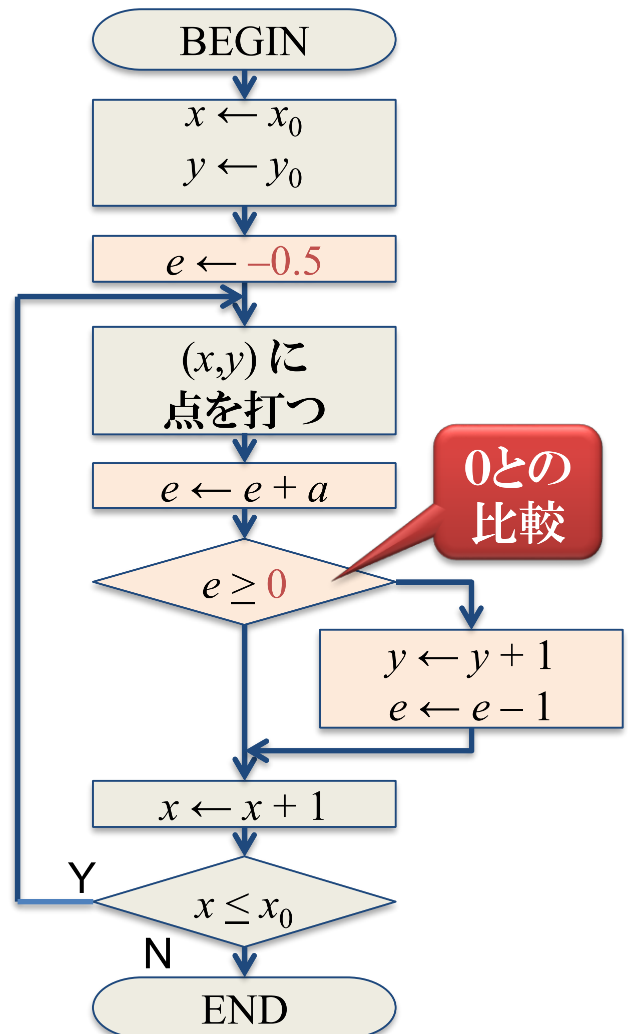




誤差の初期値をずらす



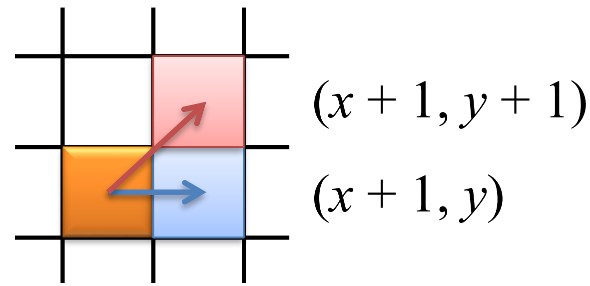
e − 1
0.5
0
e = −0.5
e < 0
−0.5
e > 0



BEGIN
x ← x0
y ← y0
e ← −0.5
(x,y) に
点を打つ
e ← e + a
0との
比較
e ≥ 0
y ← y + 1
e ← e − 1
x ← x + 1
Y
x ≤ x0
N
END

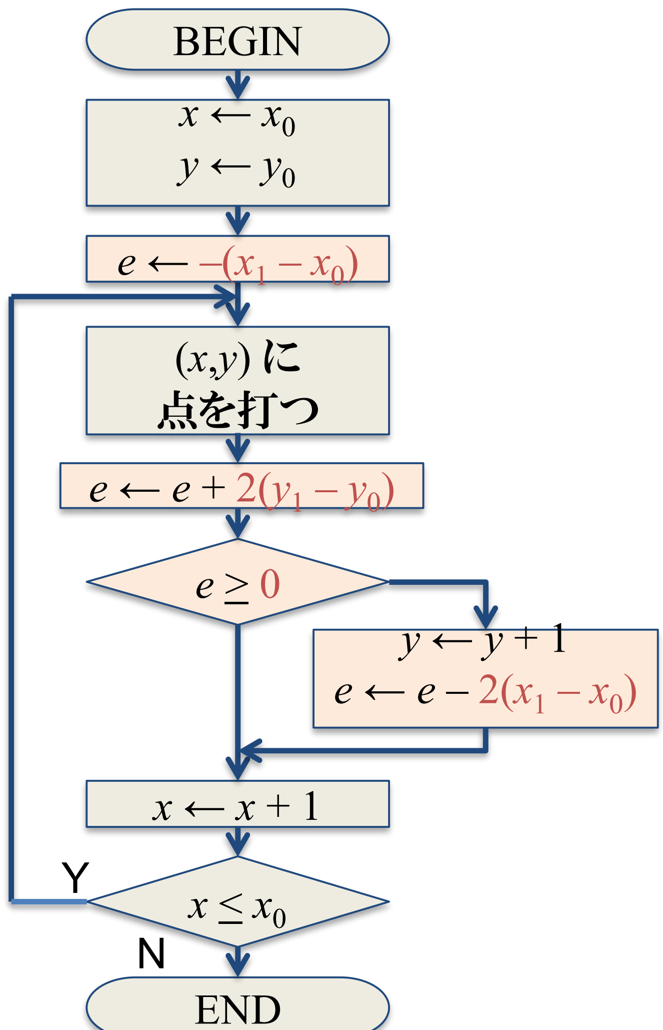
処理の整数化

- 次に打つべき点の位置の候補は次の2つ
  - $(x+1, y+1)$
  - $(x+1, y)$

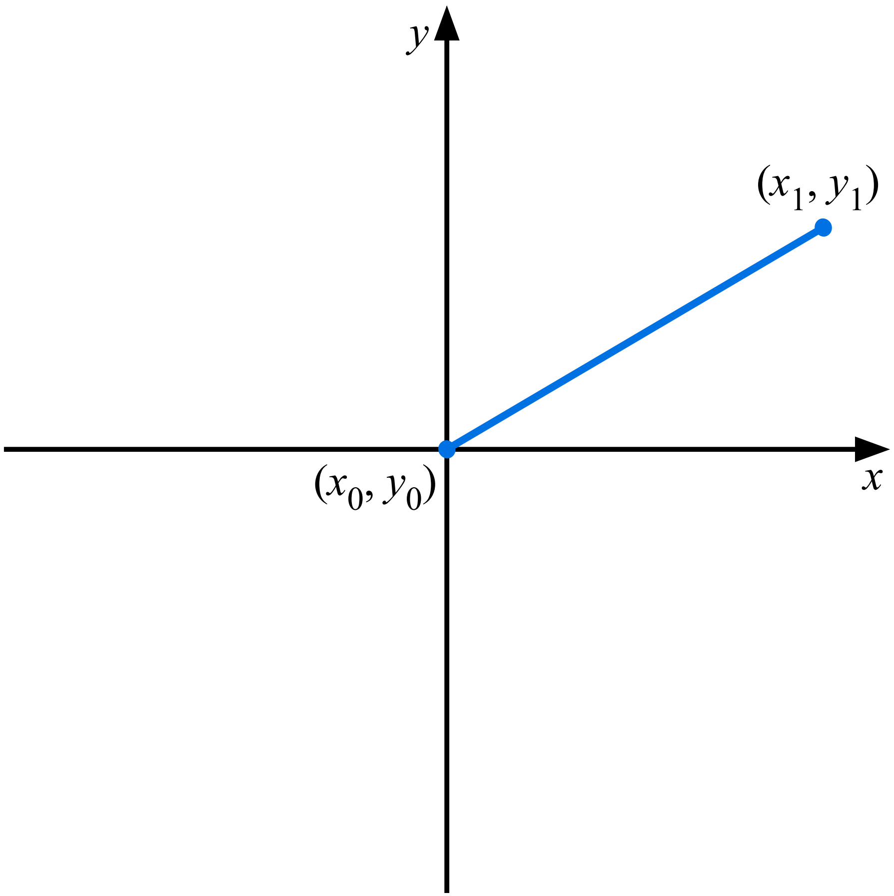


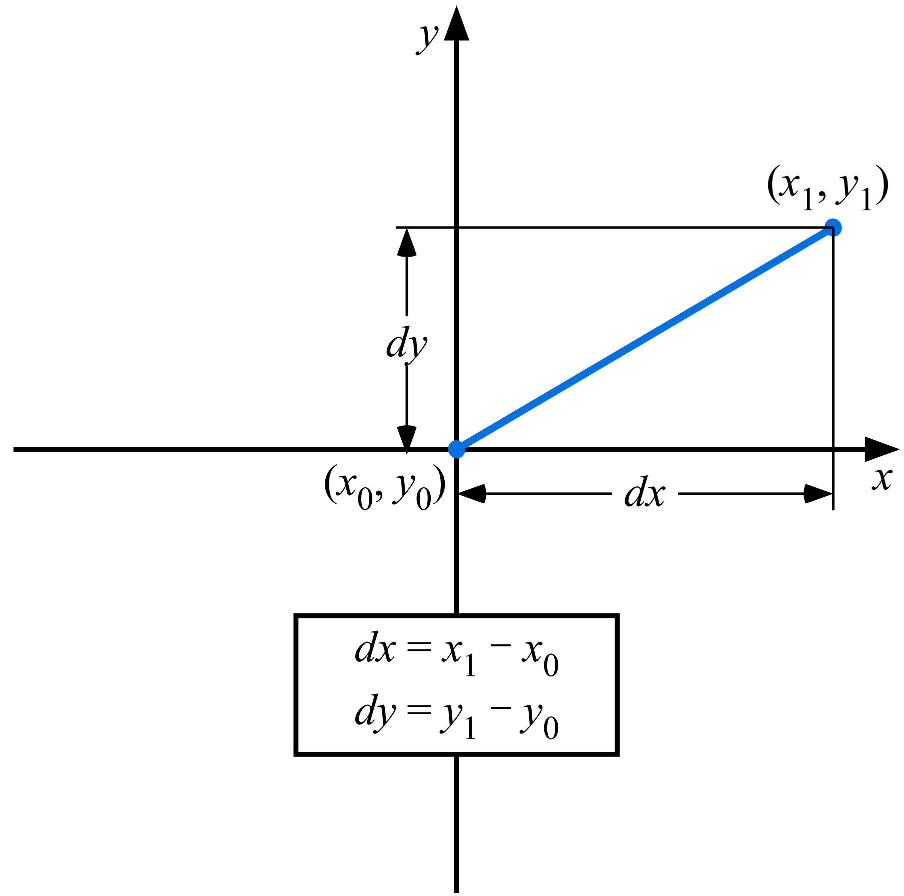
- 上のいずれかを選択する
  - e に対する加減算の後，符号判定により判断できる
  - e に関する処理に正の定数をかけても結果は同じ
- e に関する処理を $2(x_1 - x_0)$ 倍する

e に関する処理を $2(x_1 - x_0)$ 倍

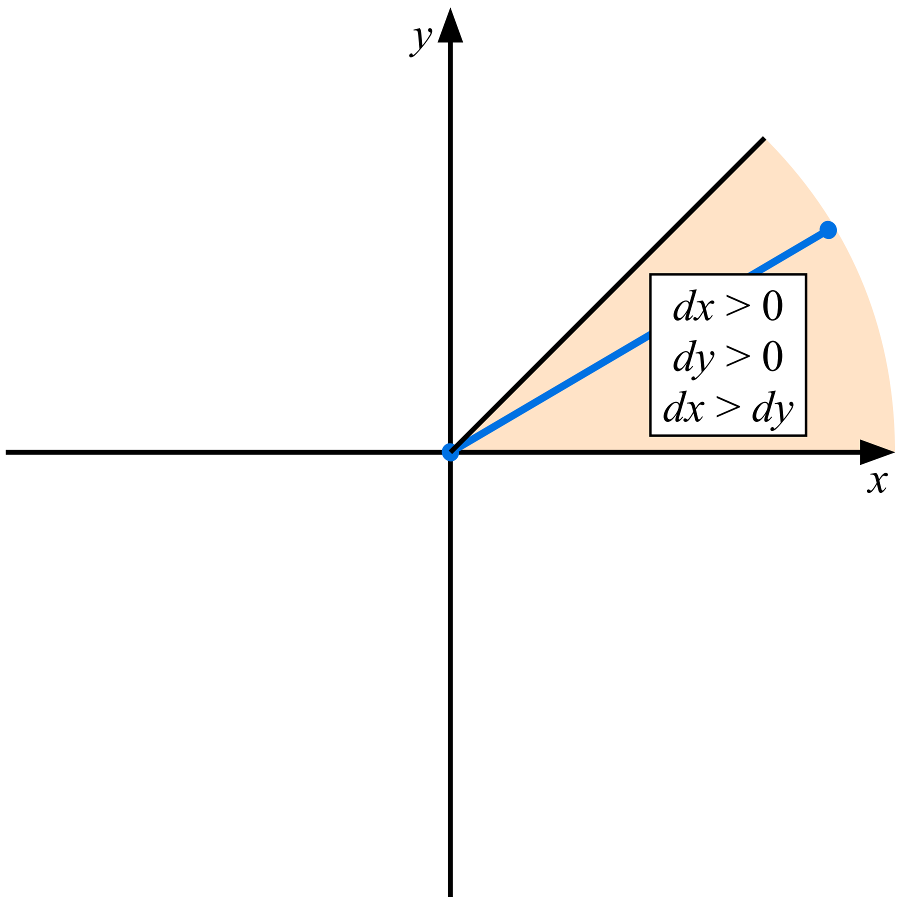


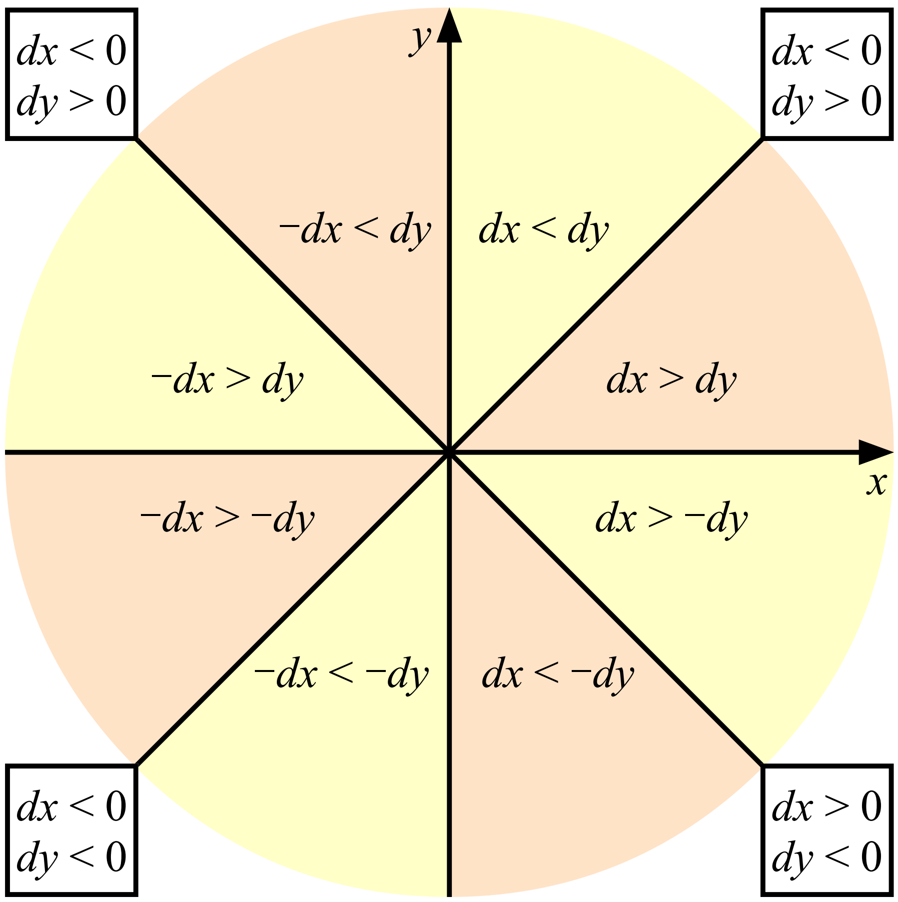
始点と終点の変位



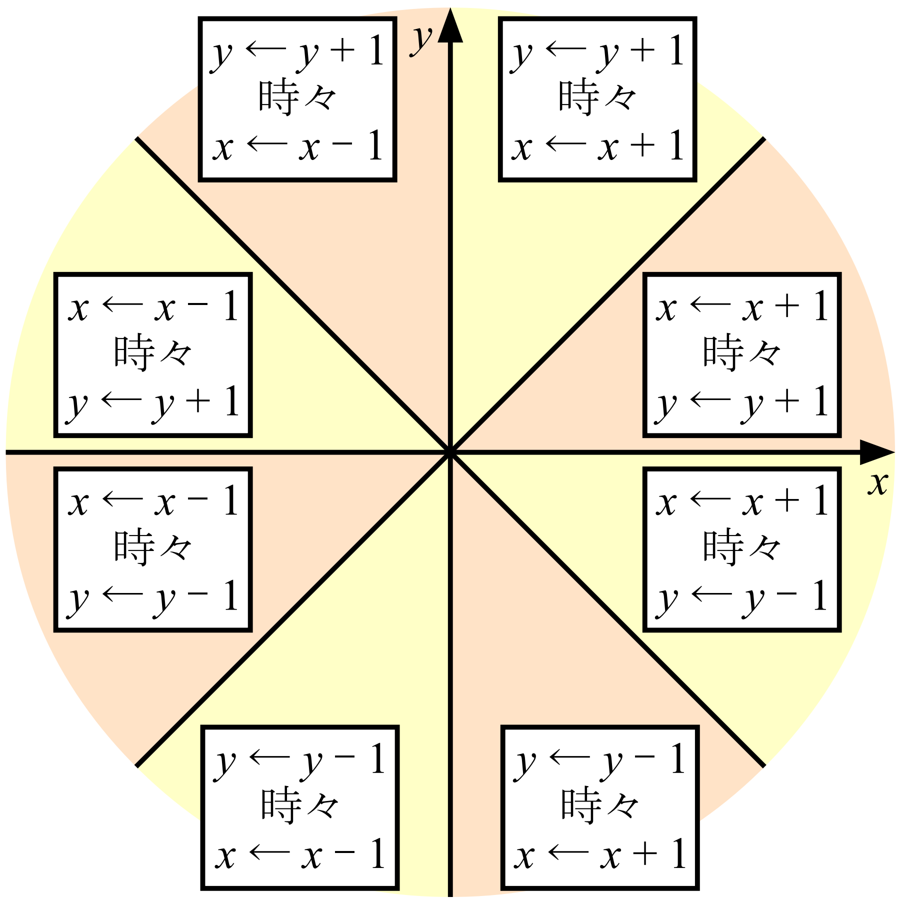


線分を生成する8分の1象限





象限ごとの処理

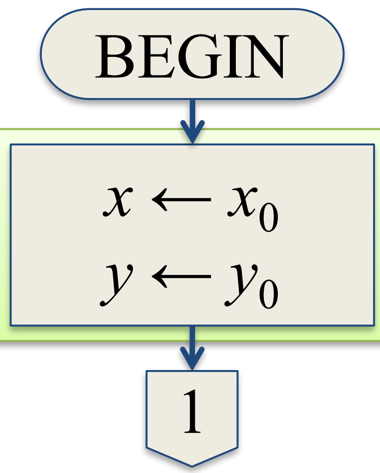


x の増分計算	
$dx \geqq 0$ なら:	$x \leftarrow x+1$
それ以外なら:	$x \leftarrow x-1$

y の増分計算	
$dy \geqq 0$ なら:	$y \leftarrow y+1$
それ以外なら:	$y \leftarrow y-1$

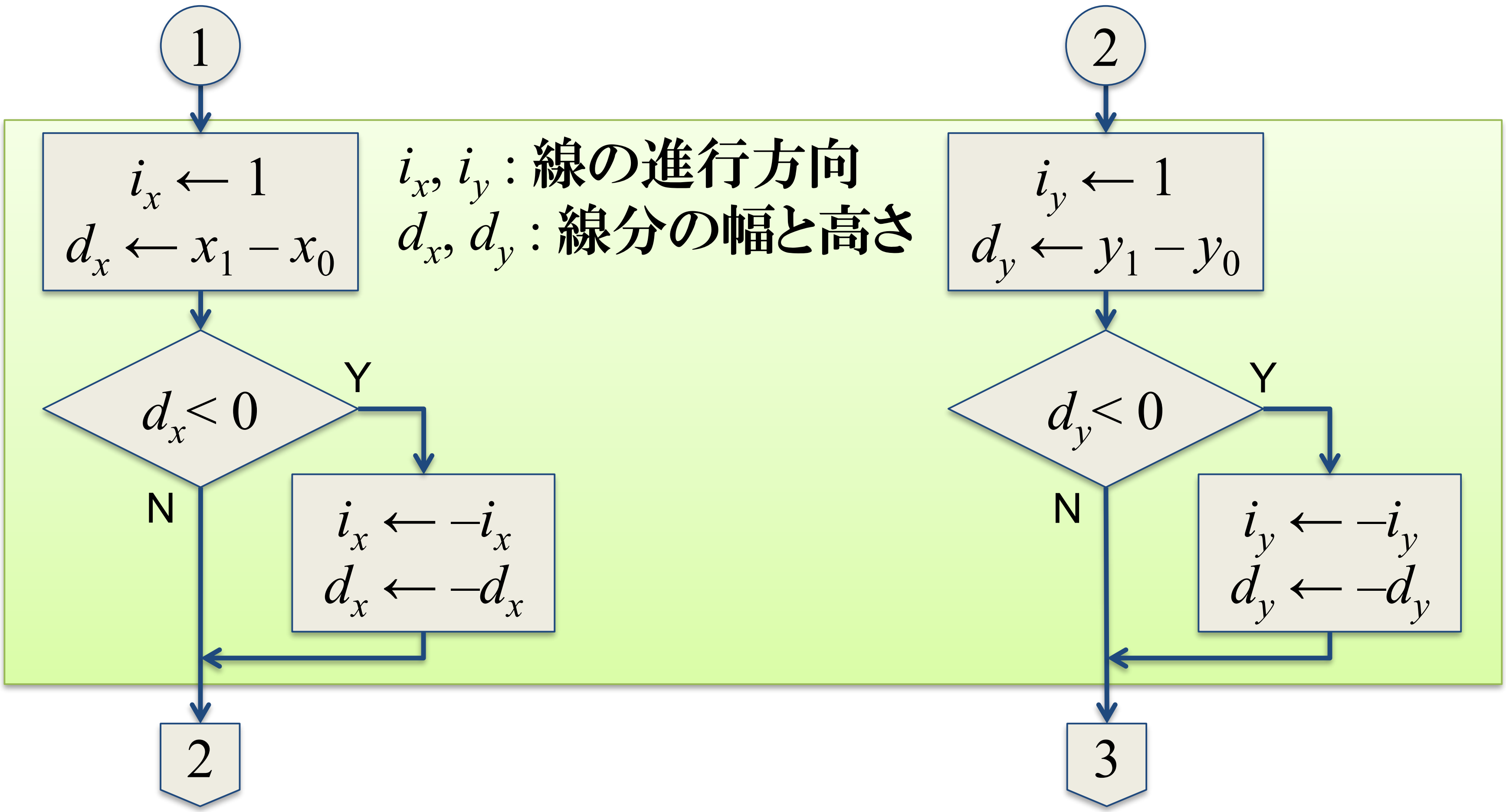
増分方向の決定	
$\lvert dx \rvert \geqq \lvert dy \rvert$ なら 右を $\lvert dx \rvert+1$回 繰り返し	・x の増分計算 ・y 方向の誤差判定に基づく y の増分計算
それ以外なら 右を $\lvert dy \rvert+1$回 繰り返し	・y の増分計算 ・x 方向の誤差判定に基づく x の増分計算

Bresenham のアルゴリズム (1)

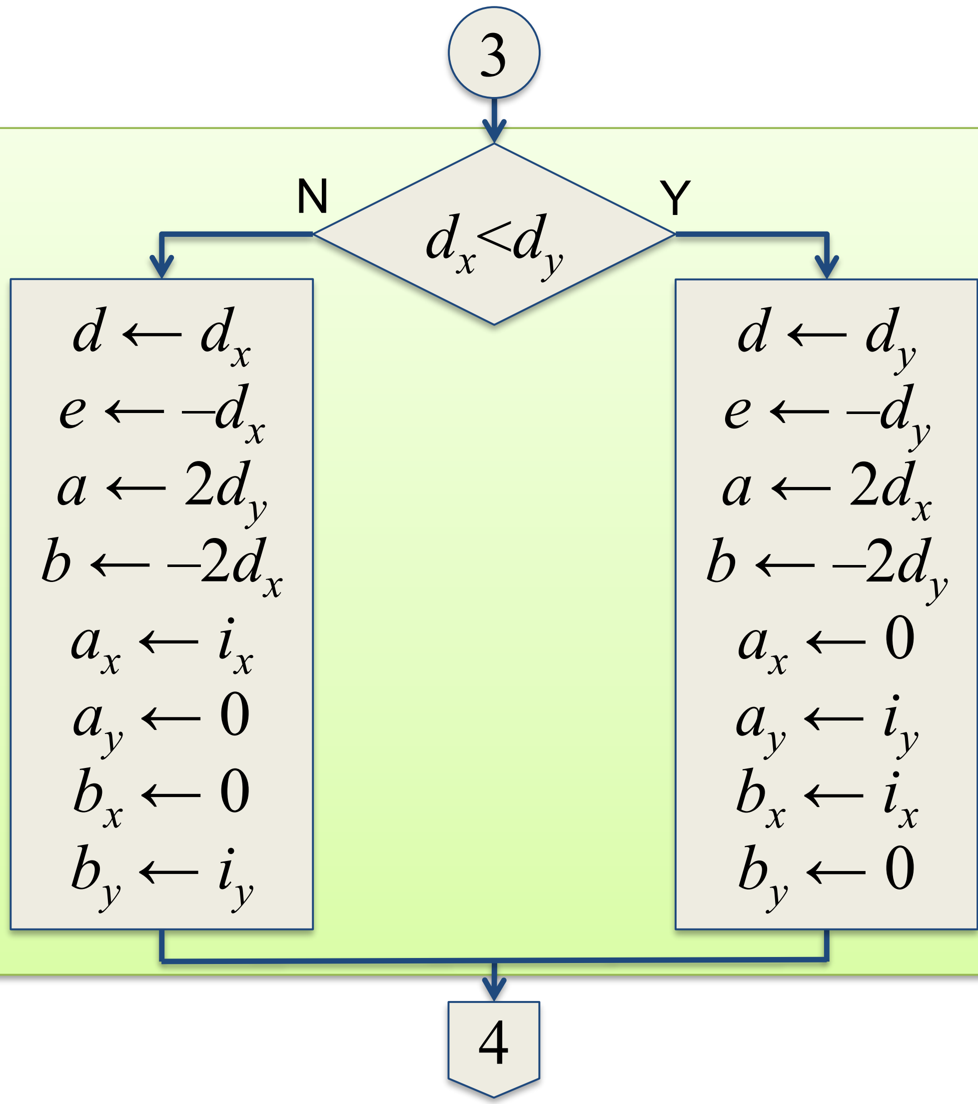


点の位置 (x, y) に始点の位置 (x_0, y_0) を設定

Bresenham のアルゴリズム (2)

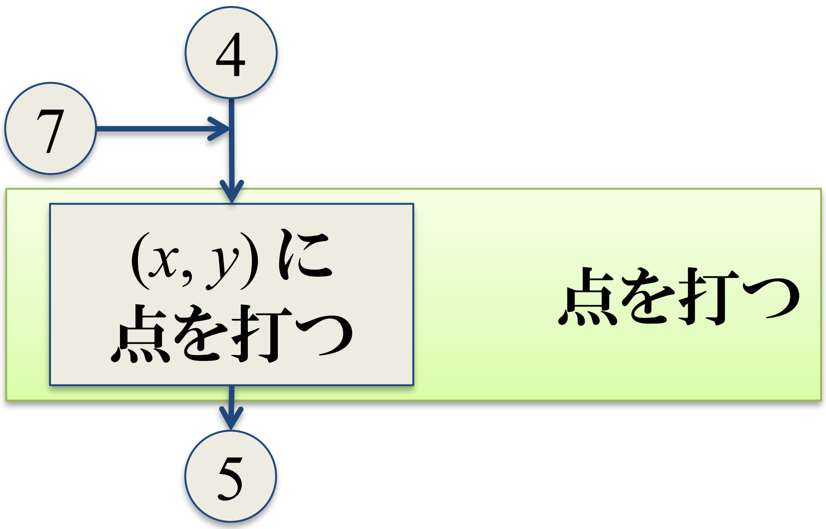


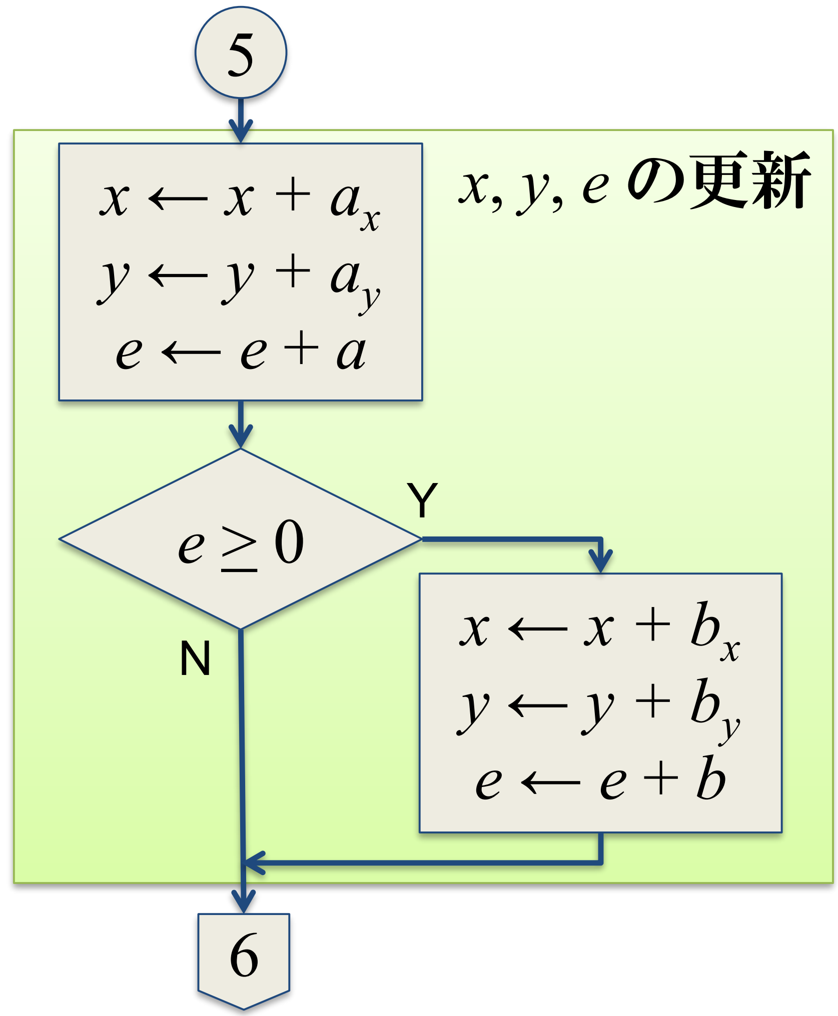
Bresenham のアルゴリズム (3)



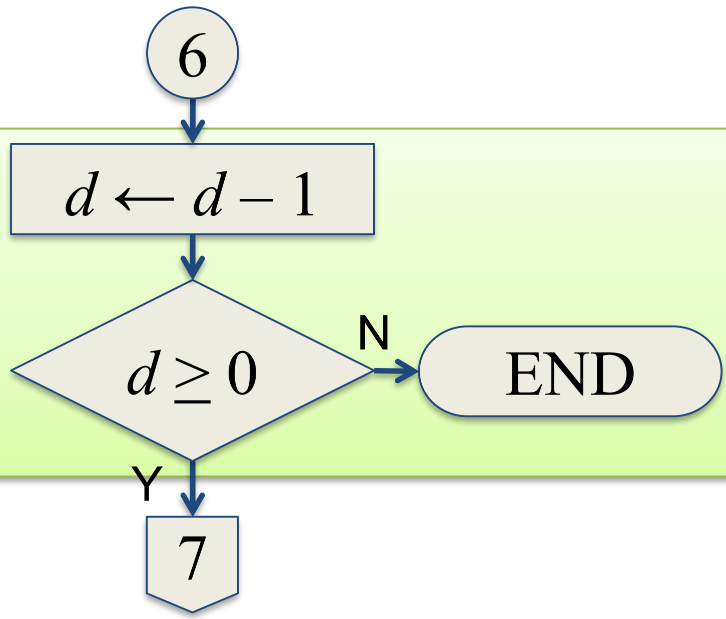
d : 点を打つ数
e : 誤差
a : 誤差の増分
b : 誤差の補正量
a_x, a_y : 毎回の増分
b_x, b_y : 補正時のの増分

Bresenham のアルゴリズム (4)



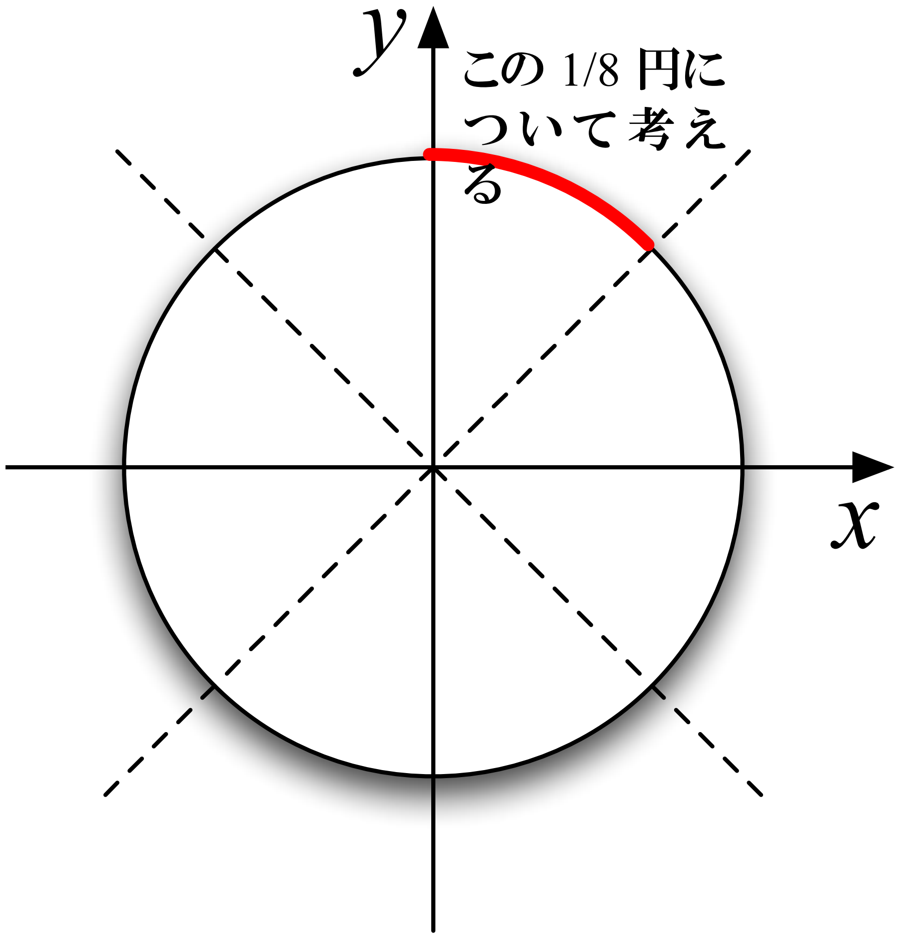


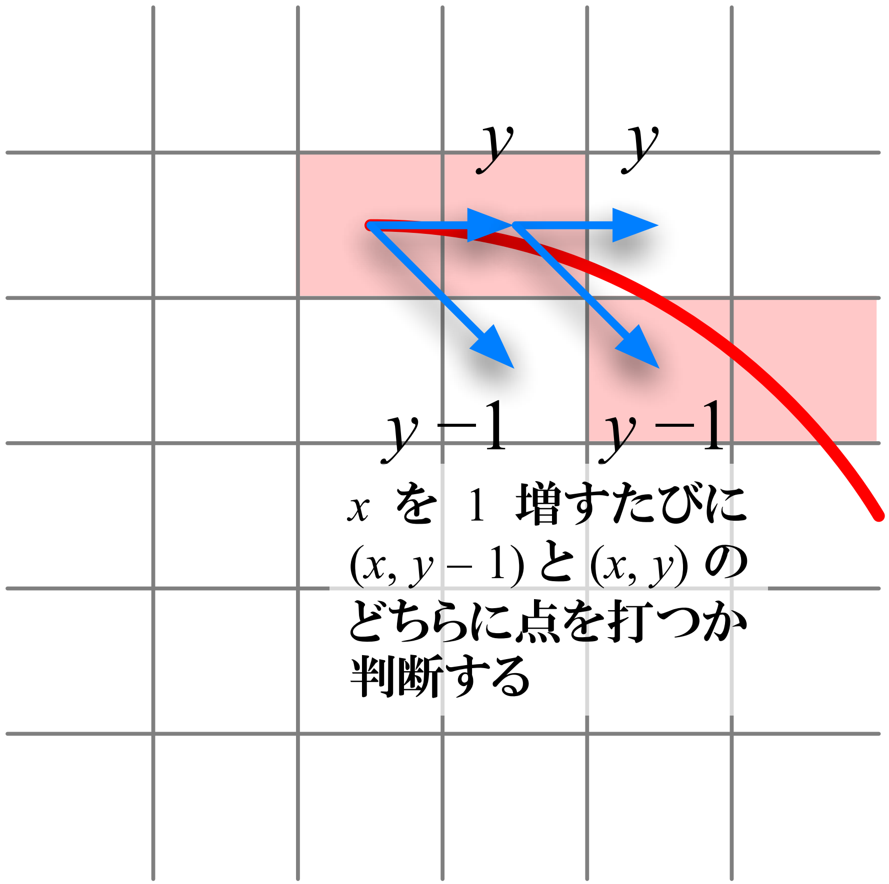
Bresenham のアルゴリズム (5)



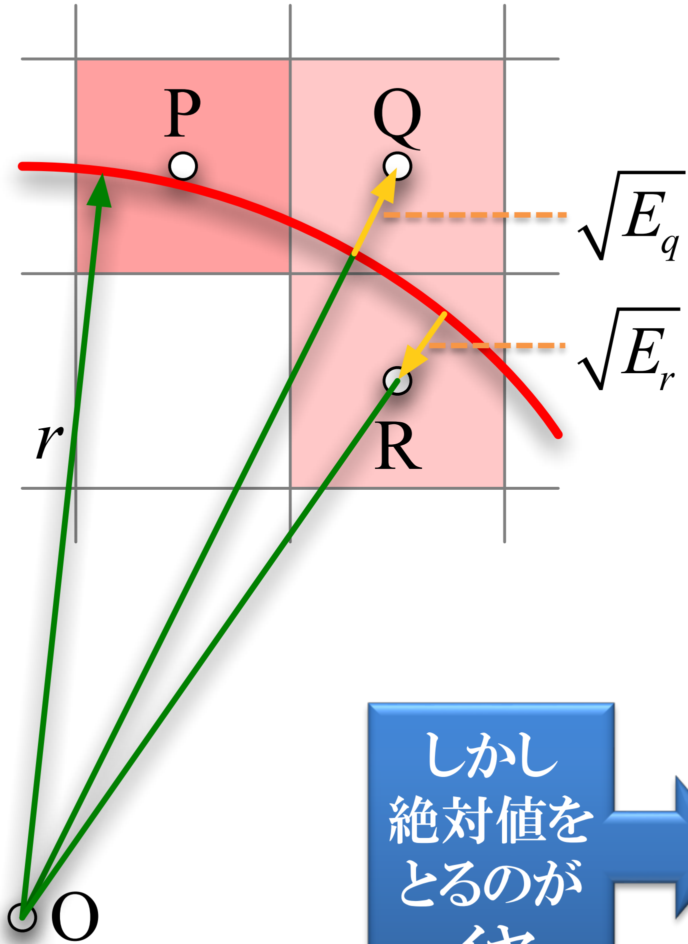
点を全部打ったら終わり

円を描く





真の円との二乗誤差による判定



$$\overrightarrow{\mathrm{OP}} = (x_i, y_i)$$

$$\overrightarrow{\mathrm{OQ}} = (x_{i+1}, y_{i+1}) = (x_i + 1, y_i)$$

$$\overrightarrow{\mathrm{OR}} = (x_{i+1}, y_{i+1}) = (x_i + 1, y_i - 1)$$

真の円の半径 r に対する二乗誤差

$$E_q = \left|\overrightarrow{\mathrm{OQ}}\right|^2 - r^2$$

$$E_r = \left|\overrightarrow{\mathrm{OR}}\right|^2 - r^2$$

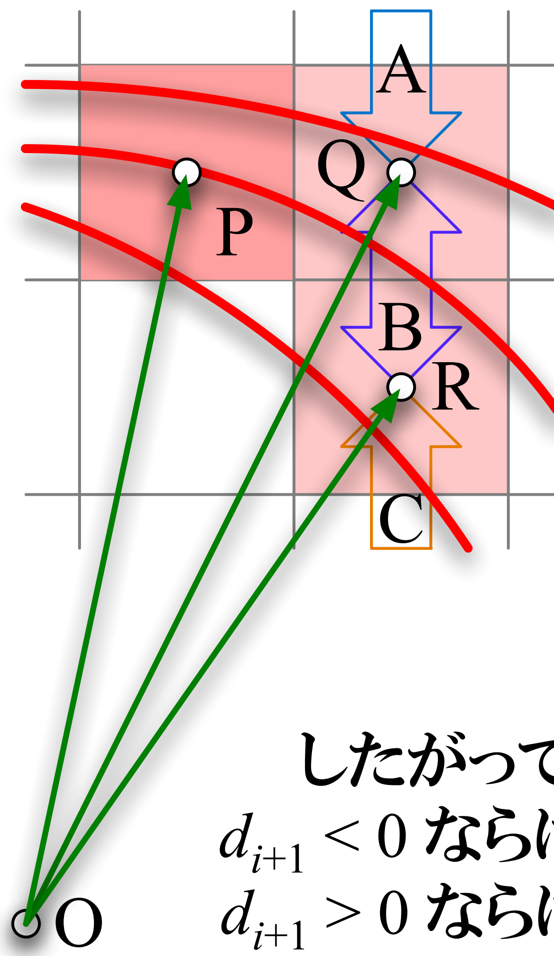
このとき

$$\left.\begin{array}{l} |E_q| < |E_r| \Rightarrow \mathrm{Q} \\ |E_q| > |E_r| \Rightarrow \mathrm{R} \end{array}\right\}$$ を選択する

しかし
絶対値を
とるのが
イヤ

絶対値を使わずに判別する

- $d_{i+1} = E_q + E_r$ とおく
- 真の線が
  - Aを通るコース(Qを選ぶ)
    - $E_q < 0, E_r < 0 \Rightarrow d_{i+1} < 0$
  - Cを通るコース(Rを選ぶ)
    - $E_q > 0, E_r > 0 \Rightarrow d_{i+1} > 0$
  - Bを通るコース
    - $E_q > 0, E_r < 0$
    - Qを選ぶべきコースなら
      - $|E_q| < |E_r| \Rightarrow d_{i+1} < 0$
    - Rを選ぶべきコースなら
      - $|E_q| > |E_r| \Rightarrow d_{i+1} > 0$



したがって
$d_{i+1} < 0$ ならば Q
$d_{i+1} > 0$ ならば R

d_{i+1} を漸化式で表す

$$d_{i+1} = E_q + E_r = \left\{ (x_i + 1)^2 + y_i^2 - r^2 \right\} + \left\{ (x_i + 1)^2 + (y_i - 1)^2 - r^2 \right\}$$

$$d_i = \left\{ (x_{i-1} + 1)^2 + y_{i-1}^2 - r^2 \right\} + \left\{ (x_{i-1} + 1)^2 + (y_{i-1} - 1)^2 - r^2 \right\}$$

$$\begin{aligned} d_{i+1} - d_i = &\left\{ (x_i + 1)^2 + y_i^2 - r^2 \right\} + \left\{ (x_i + 1)^2 + (y_i - 1)^2 - r^2 \right\} \\ &- \left\{ (x_{i-1} + 1)^2 + y_{i-1}^2 - r^2 \right\} + \left\{ (x_{i-1} + 1)^2 + (y_{i-1} - 1)^2 - r^2 \right\} \end{aligned}$$

$d_i < 0$ のとき

- この場合は Q を選択

$$\begin{cases} x_i = x_{i-1} + 1 \\ y_i = y_{i-1} \end{cases}$$

- これを代入すれば

$$d_{i+1} - d_i = 4x_{i-1} + 6$$

$$d_{i+1} = d_i + 4x_{i-1} + 6$$

$d_i \geqq 0$ のとき

- この場合は R を選択

$$\begin{cases} x_i = x_{i-1} + 1 \\ y_i = y_{i-1} - 1 \end{cases}$$

- これを代入すれば

$$d_{i+1} - d_i = 4(x_{i-1} - y_{i-1}) + 10$$

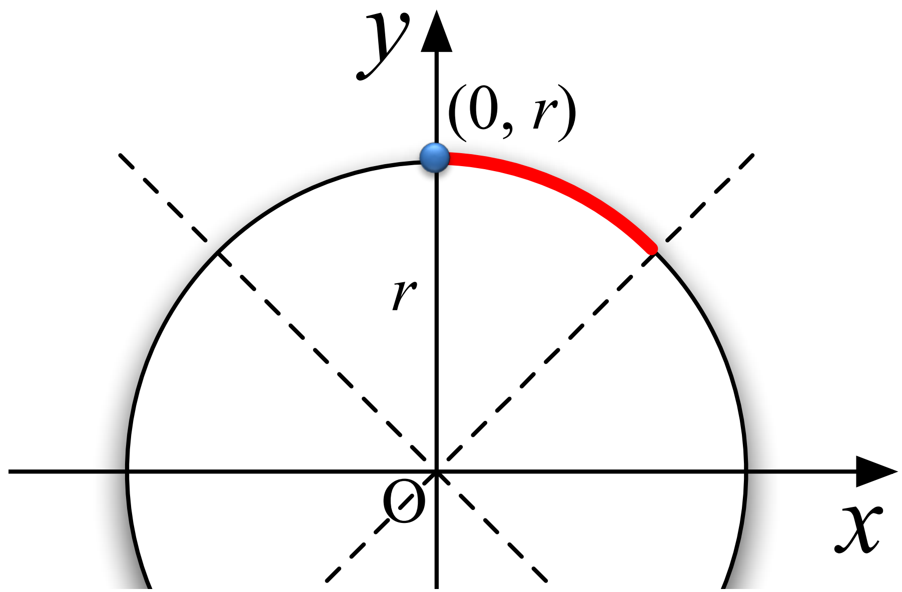
$$d_{i+1} = d_i + 4(x_{i-1} - y_{i-1}) + 10$$

起点(真上)では

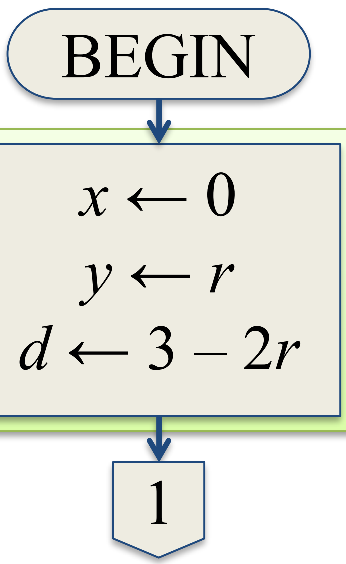
- $x_0 = 0, y_0 = r, d_0 = 0$

$$d_1 = \left\{ (x_0 + 1)^2 + y_0 - r^2 \right\} + \left\{ (x_0 + 1)^2 + (y_0 - 1) - r^2 \right\}$$

$$= 3 - 2r$$



Michener のアルゴリズム (1)



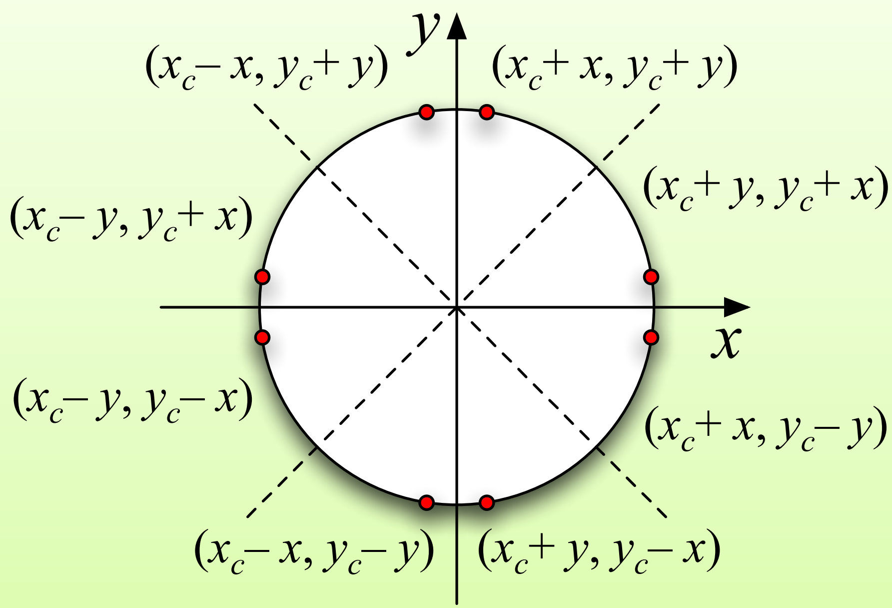
点の位置 (x, y) に真上の位置 $(0, r)$ を設定
d に $3 - 2r$ を設定

Michener のアルゴリズム (2)

1

4

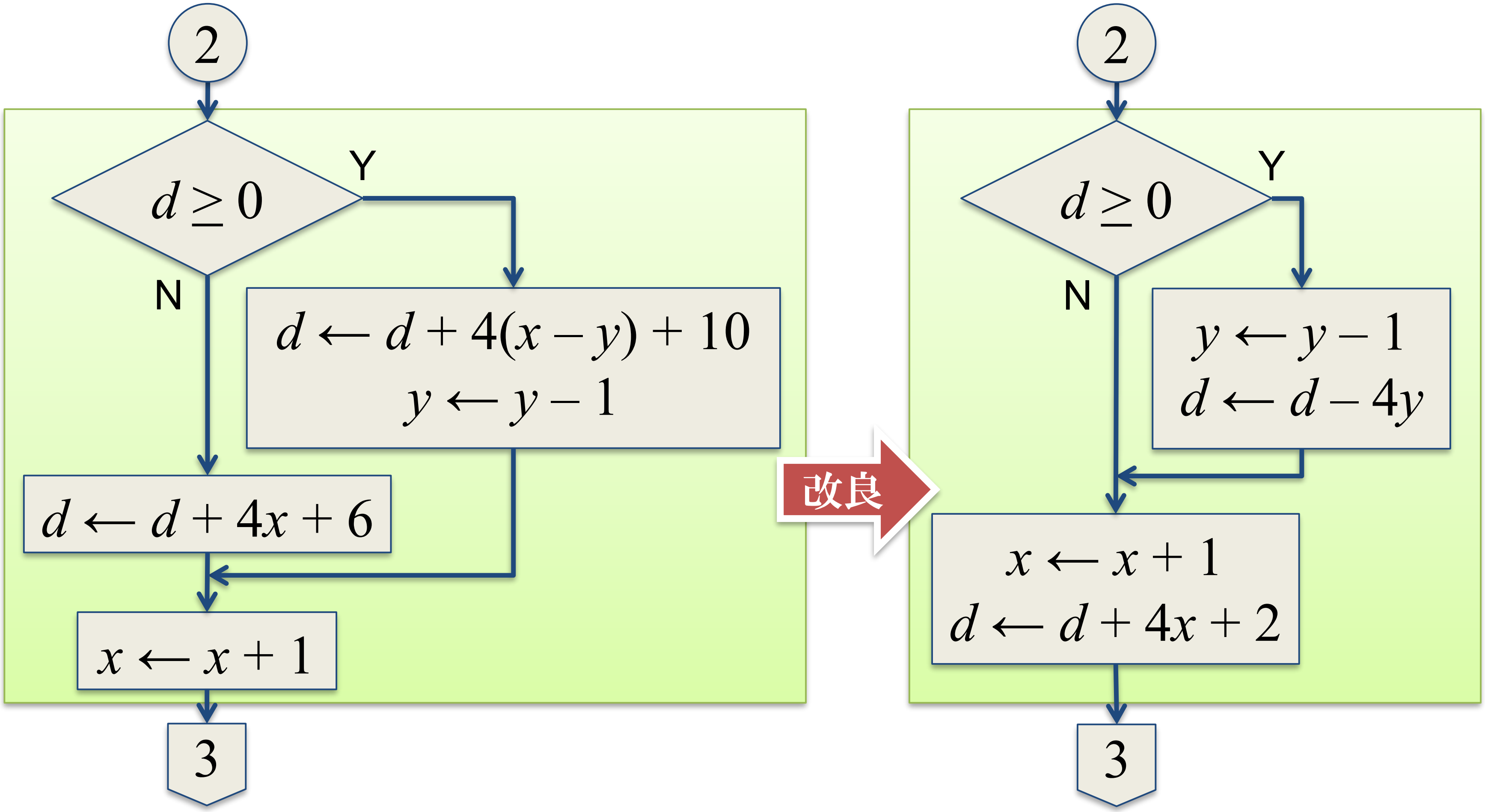
$(x_c + x, y_c + y)$, $(x_c - x, y_c - y)$
$(x_c + y, y_c + x)$, $(x_c - y, y_c - x)$
$(x_c + x, y_c - y)$, $(x_c - x, y_c + y)$
$(x_c - y, y_c + x)$, $(x_c + y, y_c - x)$
に点を打つ



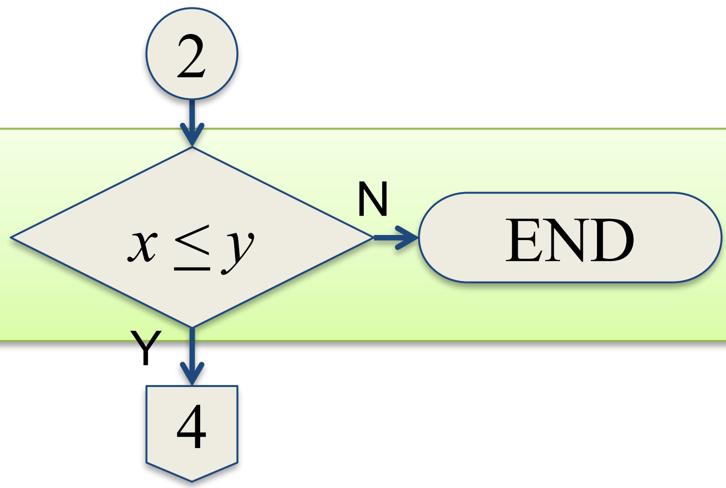
同時に8点描く

2

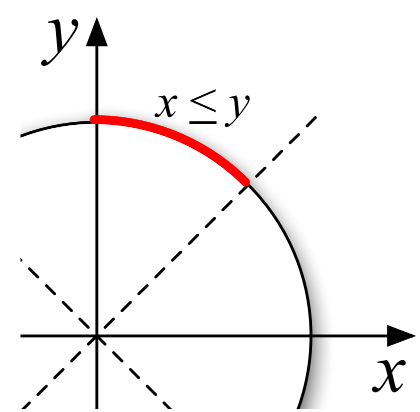
Michener のアルゴリズム (3)



Michener のアルゴリズム (4)



1/8円描いたら終わり



おわり