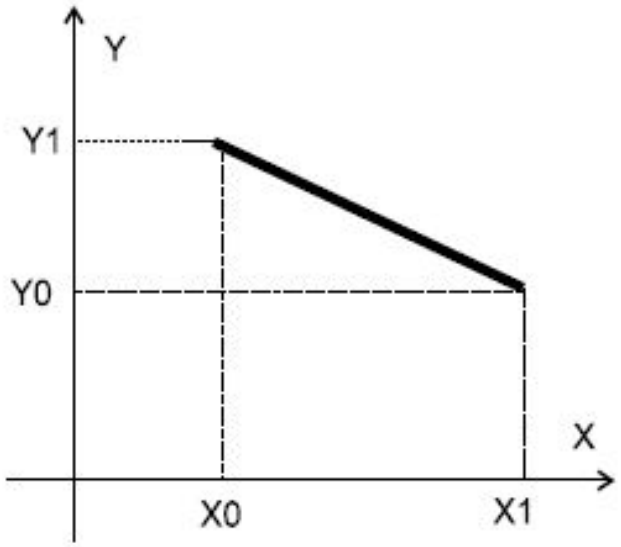
Треугольники

И не только они

Отрезки прямой линии



$$y=a\cdot x+b;$$

$$a = (y_1 - y_0)/(x_1 - x_0);$$

$$b = y_0 - a\cdot x_0.$$

Основная идея алгоритма Брезенхема для рисования прямой

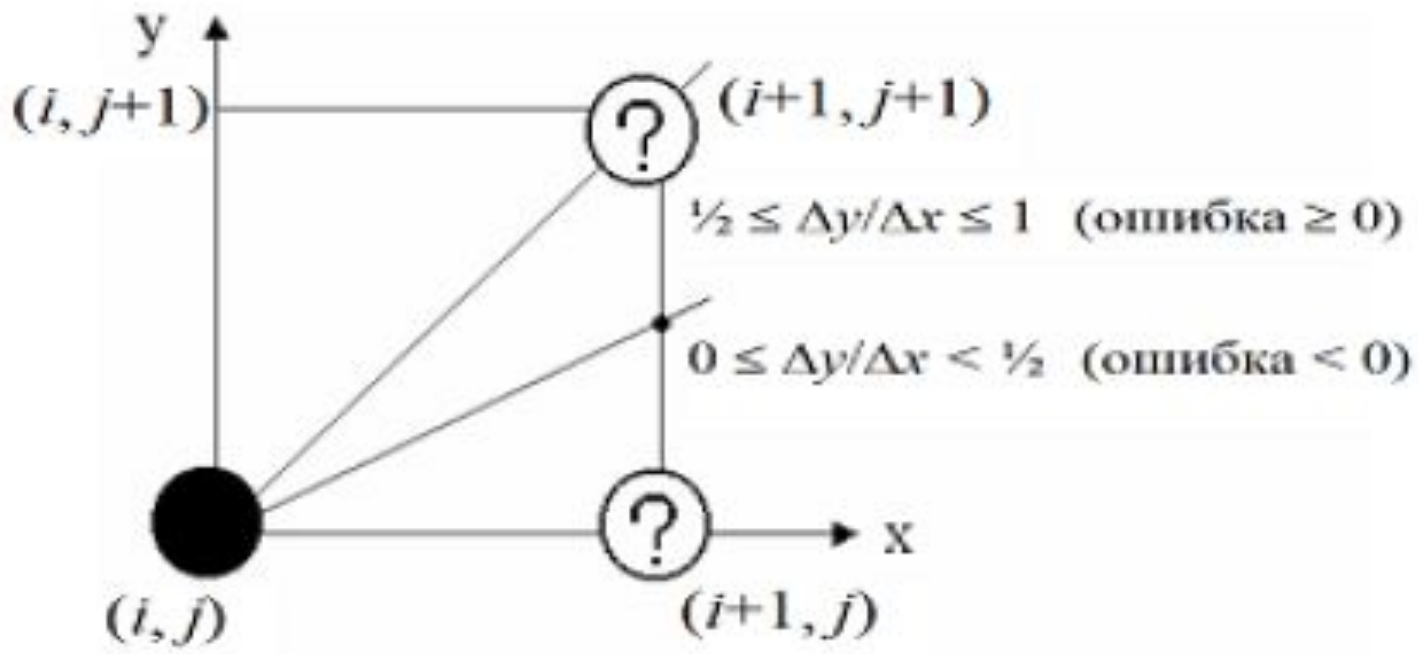
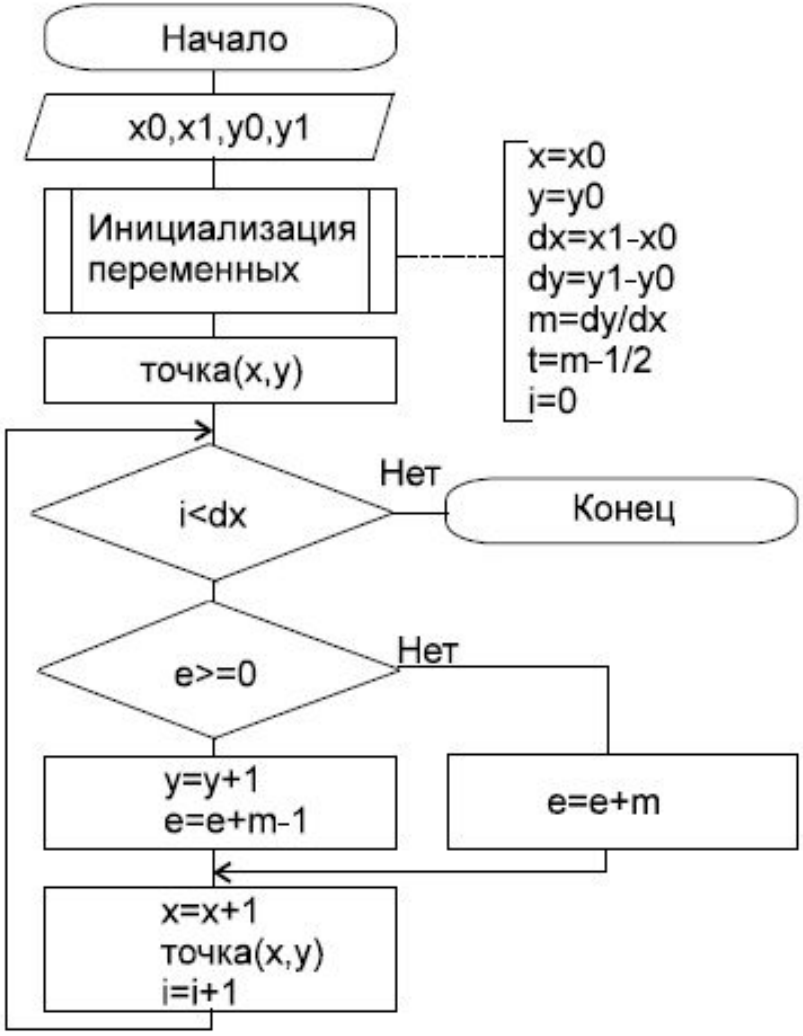
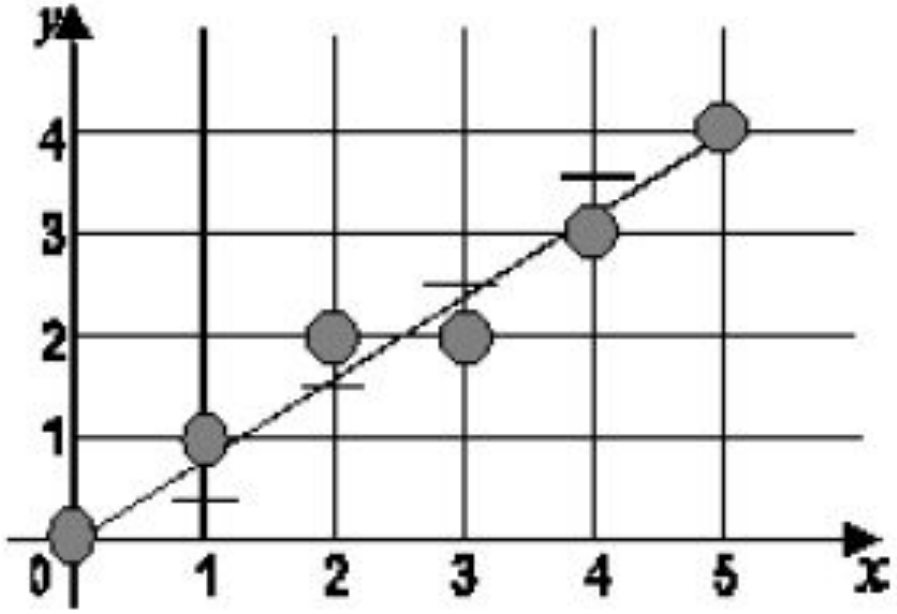


Схема алгоритма Брезенхема рисования прямой



Результат работы алгоритма Брезенхема и его трассировка



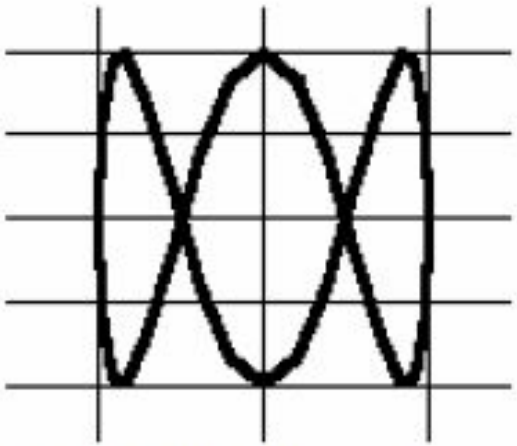
i	Точка	e	x	y
0	(0, 0)	3/10	0	0
1	(1, 1)	1/10	1	1
2	(2, 2)	-1/10	2	2
3	(3, 2)	7/10	3	2
4	(4, 3)	5/10	4	3
5	(5, 4)	3/10	5	4

Окружность

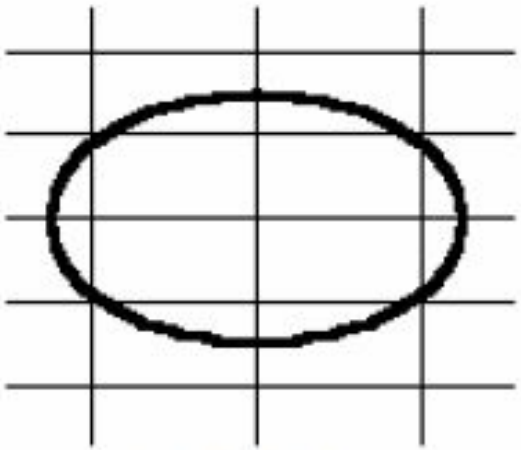
```
For X:=X0-R to X<X0+R do begin
        Точка(X, Y0+sqrt(R*R-sqr(X-X0)));
        Точка(X, Y0-sqrt(R*R- sqr(X-X0)));
End;
```

Параметрически заданные кривые

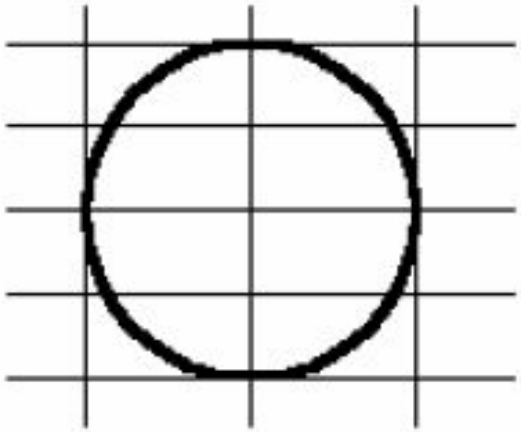
$$x = x_0 + R_1 \cdot \cos(\omega_1 \cdot t);$$
$$y = y_0 + R_2 \cdot \sin(\omega_2 \cdot t);$$



а) фигура Лиссажу



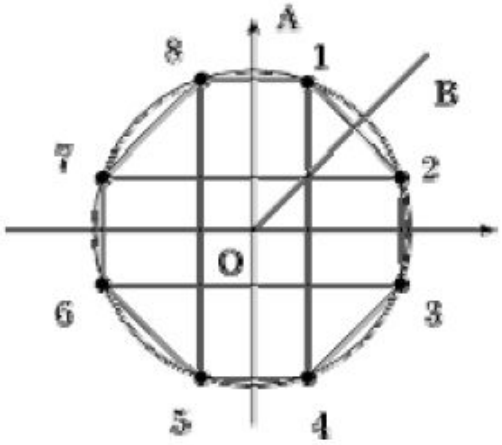
б) эллипс



в) окружность

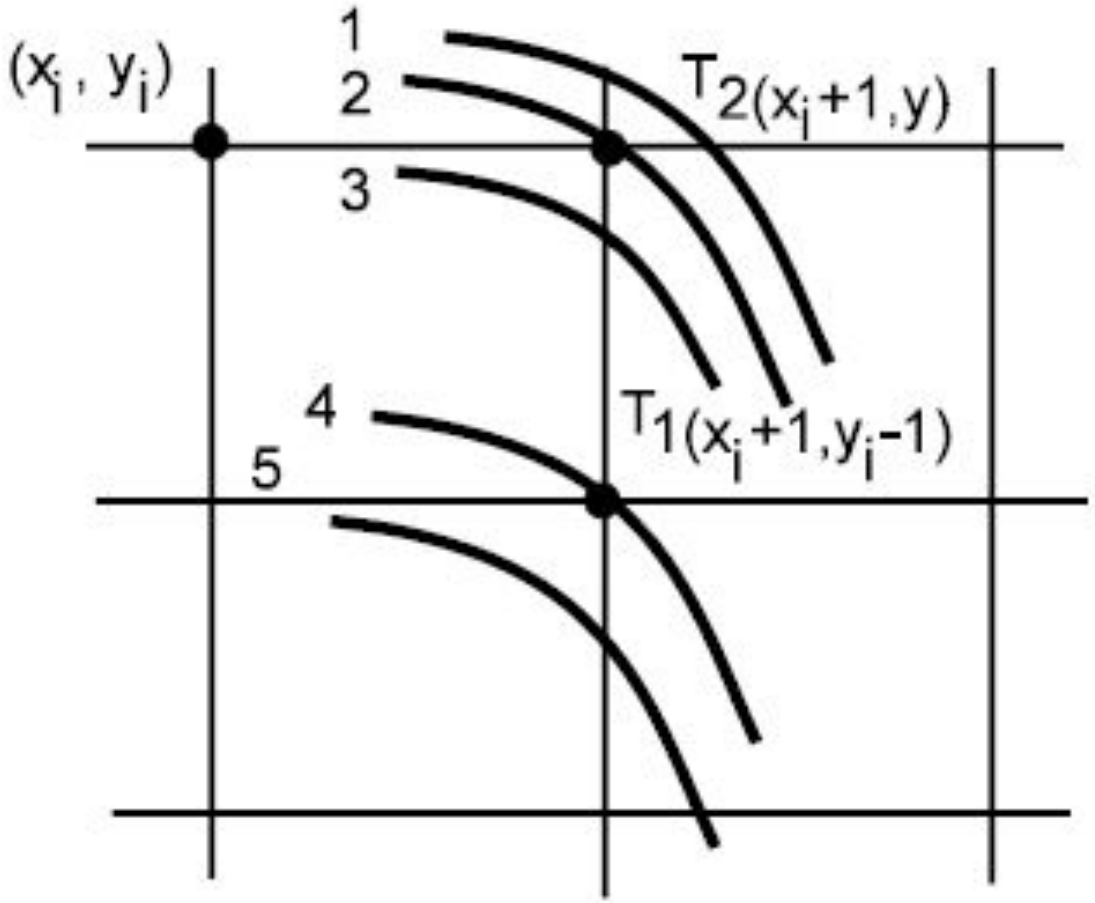
```
procedure my_circle(x,y,r:integer);
 var alfa: integer;
 begin
  moveto(x+r,y);
  for alfa:=1 to 360 do
  { угол измеряется в градусах }
  lineto(round(x+r*cos(alfa*pi/360)),
         round(y+r*sin(alfa*pi/360)))
  { здесь используется радианная мера углов }
 end;
```

Симметричное отражение точки окружности




```
Procedure Draw8Pixels;
begin
  Точка(x+x0, y+y0, color);
  Точка(x+x0, -y+y0, color);
  Точка(-x+x0, y+y0, color);
  Точка(-x+x0, -y+y0, color);
  Точка(y+x0, x+y0, color);
  Точка(y+x0, -x+y0, color);
  Точка(-y+x0, x+y0, color);
  Точка(-y+x0, -x+y0, color);
end;
```

Возможное расположение окружности относительно пикселов экрана

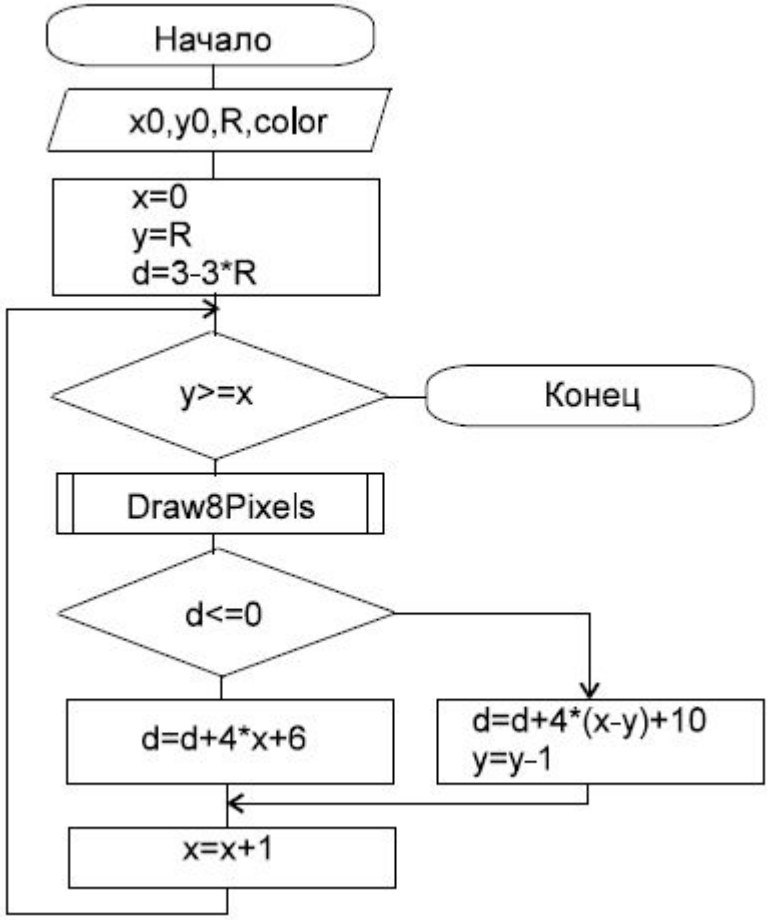


$$R_1^2 = (x_i+1)^2+(y_i-1)^2, \qquad \Delta^i_{T1} = R_1^2 - R^2 = (x_i+1)^2+(y_i-1)^2 - R^2,$$
$$R_2^2 = (x_i+1)^2+(y_i)^2. \qquad \Delta^i_{T2} = R_2^2 - R^2 = (x_i+1)^2+(y_i)^2 - R^2$$

$$\Delta^i = \Delta^i_{T1}+\Delta^i_{T2}.$$

если $\Delta^i > 0$, выберем точку T_1;
если $\Delta^i \le 0$, выберем точку T_2.

Схема алгоритма Брезенхема рисования окружности



Кривые и поверхности Безье

$$x = P_x(t) = \sum_{i=0}^{m} C_m^i t^i (1-t)^{m-i} x_i ,$$

$$y = P_y(t) = \sum_{i=0}^{m} C_m^i t^i (1-t)^{m-i} y_i ,$$

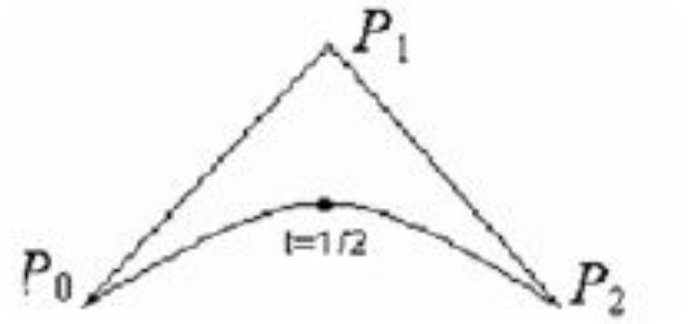
$$C_m^i = \frac{m!}{i!(m-i)!}$$

Кривая Безье 1-го порядка

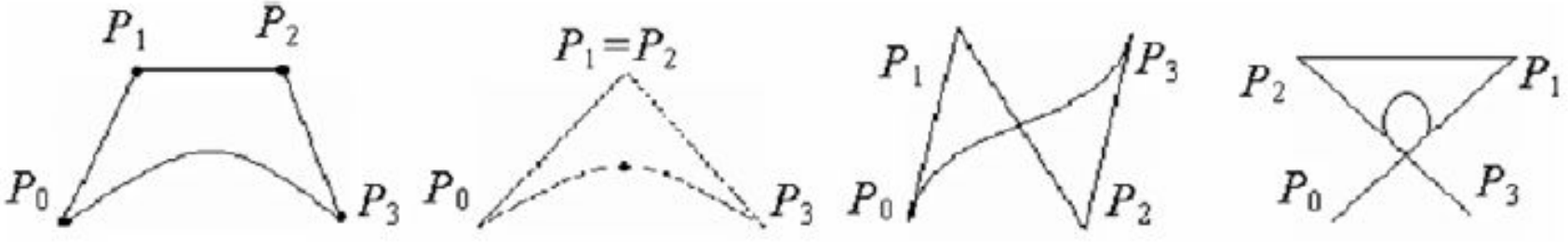
$$P(t) = (1-t)\cdot P_0 + t\cdot P_1 \, .$$

Кривая Безье 2-го порядка

$$P(t) = (1\text{-}t)^2 \cdot P_0 + 2t\,(1\text{-}t) \cdot P_1 + t^2 \cdot P_2 \,.$$

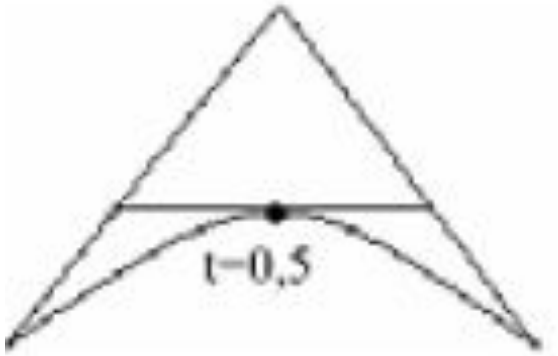


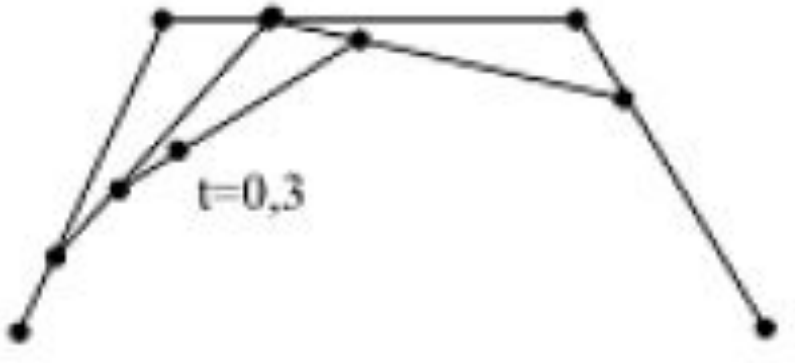
Кривая Безье 3-го порядка



$$P(t) = (1-t)^3 \cdot P_0 + 3t(1-t)^2 \cdot P_1 + 3t^2(1-t) \cdot P_2 + t^3 \cdot P_3.$$

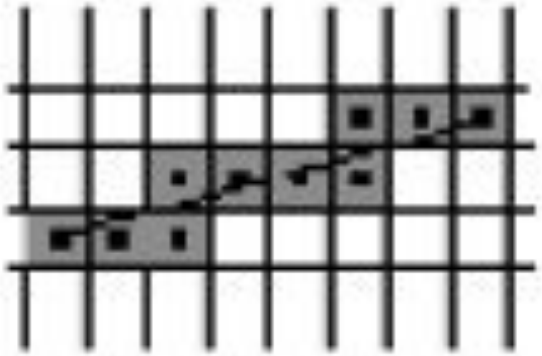
Геометрический способ построения кривых Безье



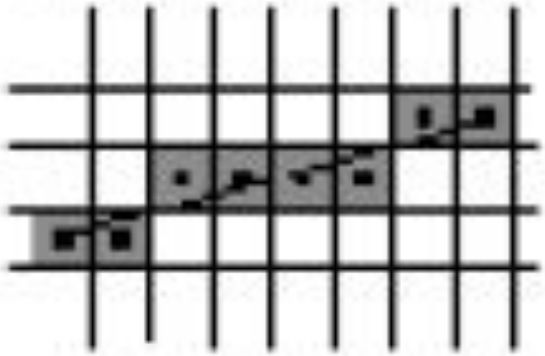


Смежность

4-связное представление



8-связное представление

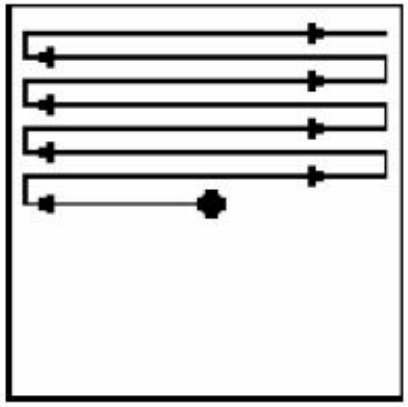


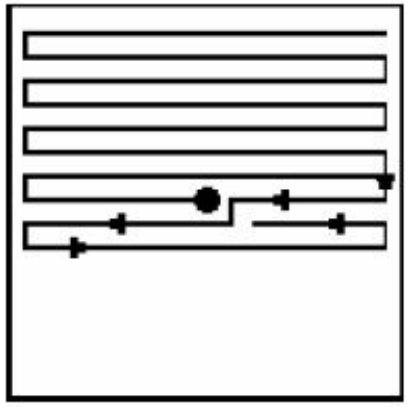
Недостатки закрашивания

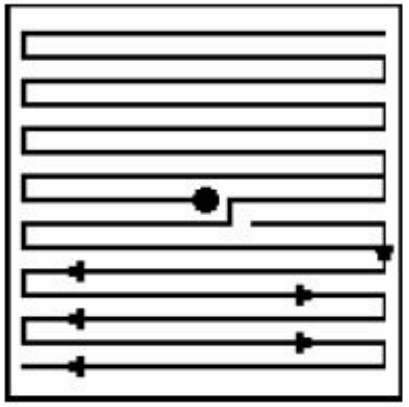
Простейший метод перекраски 4-х связной области

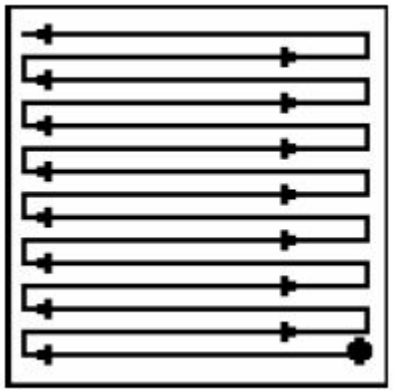
```
void Flood_Fill_4(int x, int y, int oldcolor, int newcolor)
{
    if ( GetPixel (x, y)==oldcolor )
    {
            SetPixel (x, y, newcolor);

            Flood_Fill_4(x-1, y, oldcolor, newcolor);
            Flood_Fill_4(x, y-1, oldcolor, newcolor);
            Flood_Fill_4(x+1, y, oldcolor, newcolor);
            Flood_Fill_4(x, y+1, oldcolor, newcolor);
    }

}
```

Алгоритм с затравкой

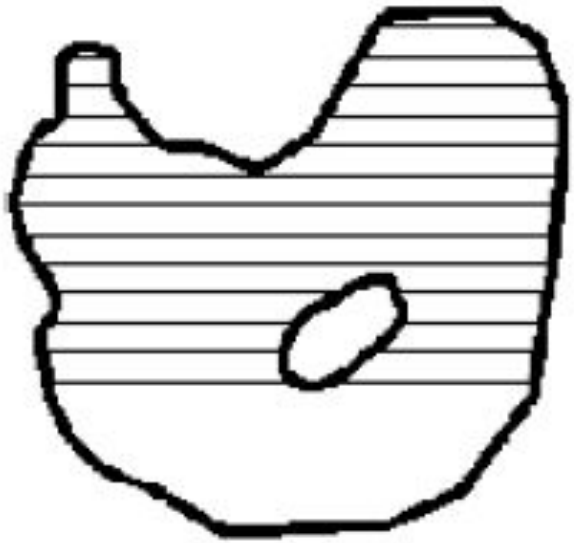








Работа модифицированного рекурсивного алгоритма



Заполнение полигонов

1) Найти *miny* и *maxy* среди всех вершин P_i.

2) Выполнить цикл по y от *min* до *max*.

{

3) Нахождение точек пересечения всех отрезков контура с горизонталью y. Координаты x_j точек пересечения записать в массив.

4) Сортировка массива $\{x_j\}$ по возрастанию.

5) Вывод горизонтальных отрезков с координатами:

(x_0,y)-(x_1,y)

(x_2,y)-(x_3,y)

...

(x_{2k},y)-(x_{2k+1},y)

}