thuật toán vẽ đường tròn

Bởi:
lâm thị ngọc châu

Thuật toán vẽ đường tròn

Trong hệ tọa độ Descartes, phương trình đường tròn bán kính R có dạng:

Với tâm O(0,0) : $x^2 + y^2 = R^2$

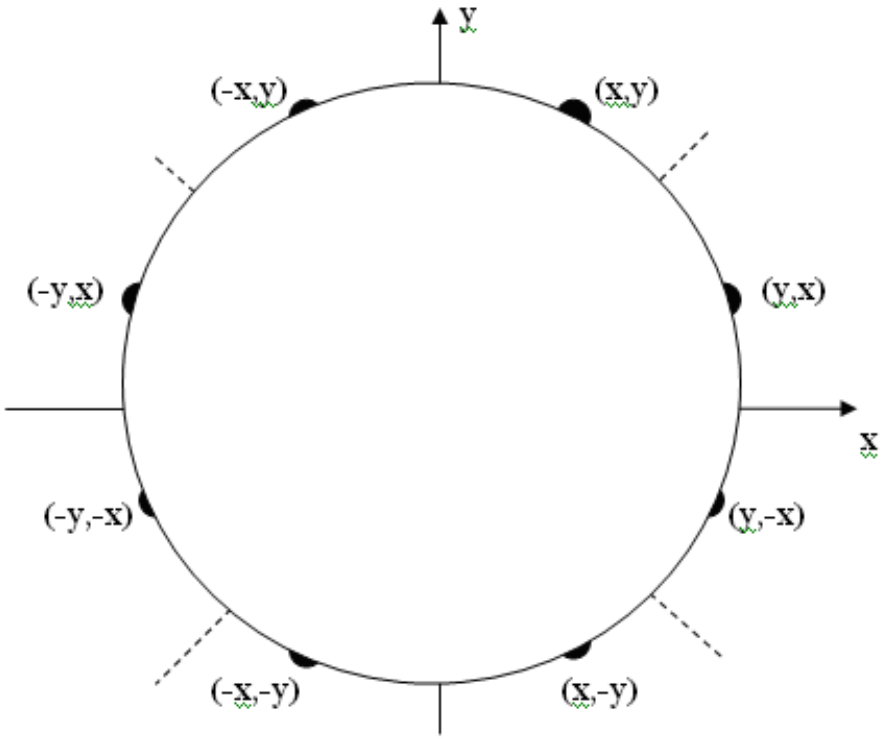
Với tâm C(x_c,y_c): $(x - x_c)^2 + (y - y_c)^2 = R^2$

Trong hệ tọa độ cực :

$$\begin{cases} x = x_c + R.\cos\text{-}0\text{-} \\ y = y_c + Y.\sin\text{-}0\text{-} \end{cases}$$

với -0- thuộc [0, 2pi].

Do tính đối xứng của đường tròn C (xem hình 1.7) nên ta chỉ cần vẽ 1/8 cung tròn, sau đó lấy đối xứng qua 2 trục tọa độ và 2 đường phân giác thì ta vẽ được cả đường tròn.



Hình 1.7 : Đường tròn với các điểm đối xứng.

Thuật toán đơn giản

Cho x = 0, 1, 2, ..., int($\frac{R\sqrt{2}}{2}$) v?i R>1.

- Tại mỗi giá trị x, tính

int($y = \sqrt{R^2 - x^2}$).

- Vẽ điểm (x,y) cùng 7 điểm đối xứng của nó.

Cài đặt minh họa thuật toán đơn giản.

Procedure Circle (x_c, y_c, R : integer) ;

Var x, y : integer ;

Procedure DOIXUNG ;

Begin

putpixel (x_c + x , y_c +y, color) ;

putpixel (x_c - x , y_c + y, color) ;

putpixel (x_c + x , y_c - y, color) ;

putpixel (x_c - x , y_c- y, color) ;

putpixel (x_c + y , y_c + x, color) ;

putpixel (x_c - y , y_c + x, color) ;

putpixel (x_c + y , y_c - x, color) ;

putpixel (x_c - y , y_c - x, color) ;

End ;

Begin

For x : = 0 to round(R*Sqrt(2)/2) do

Begin

y : = round(Sqrt(R*R - x*x)) ;

DOIXUNG;

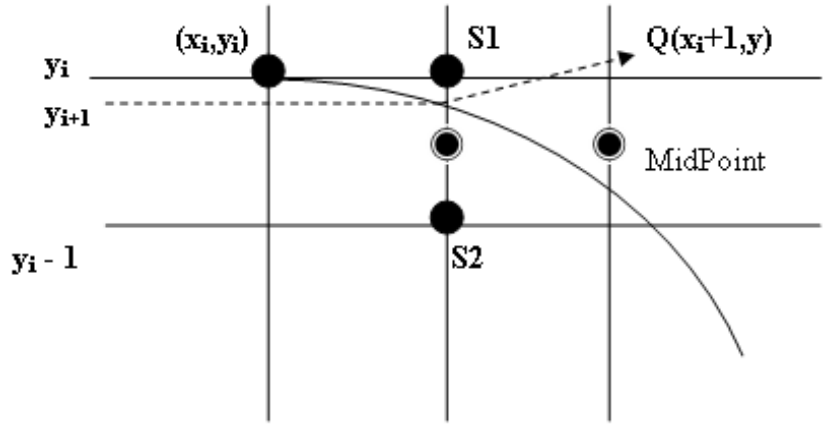
End ;

End ;

Thuật toán xét điểm giữa (MidPoint)

Do tính đối xứng của đường tròn nên ta chỉ cần vẽ 1/8 cung tròn, sau đó lấy đối xứng là vẽ được cả đường tròn. Thuật toán MidPoint đưa ra cách chọn y_{i+1} là y_i hay y_i-1 bằng cách so sánh điểm thực Q(x_{i+1},y) với điểm giữa MidPoind là trung điểm của S1 và S2. Chọn điểm bắt đầu để vẽ là (0,R). Giả sử (x_i, y_i) là điểm nguyên đã tìm được ở bước thứ i (xem hình 1.8), thì điểm (x_{i+1}, y_{i+1}) ở bước i+1 là sự lựa chọn giữa S1 và S2.

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



Hình 1.8 : Đường tròn với điểm $Q(x_i+1, y)$ và điểm MidPoint.

Đặt $F(x,y) = x^2 + y^2 - R^2$, ta có :

. $F(x,y) < 0$, nếu điểm (x,y) nằm trong đường tròn.

. $F(x,y) = 0$, nếu điểm (x,y) nằm trên đường tròn.

. $F(x,y) > 0$, nếu điểm (x,y) nằm ngoài đường tròn.

Xét $P_i = F(MidPoint) = F(x_i +1, y_i - 1/2)$. Ta có :

- Nếu $P_i < 0$: điểm MidPoint nằm trong đường tròn. Khi đó, điểm thực Q gần với điểm S1 hơn nên ta chọn $y_{i+1} = y_i$.

- Nếu $P_i >= 0$: điểm MidPoint nằm ngòai đường tròn. Khi đó, điểm thực Q gần với điểm S2 hơn nên ta chọn $y_{i+1} = y_i - 1$.

Mặt khác :

$P_{i+1} - P_i = F(x_{i+1} +1, y_{i+1} - 1/2) - F(x_i + 1, y_i - 1/2)$

$= [(x_{i+1} +1)^2 + (y_{i+1} - 1/2)^2 - R^2] - [(x_i +1)^2 + (y_i - 1/2)^2 - R^2]$

$= 2x_i + 3 + ((y_{i+1})^2 + (y_i)^2) - (y_{i+1} - y_i)$

Vậy :

- Nếu $P_i < 0$: chọn $y_{i+1} = y_i$. Khi đó $P_{i+1} = P_i + 2x_i +3$

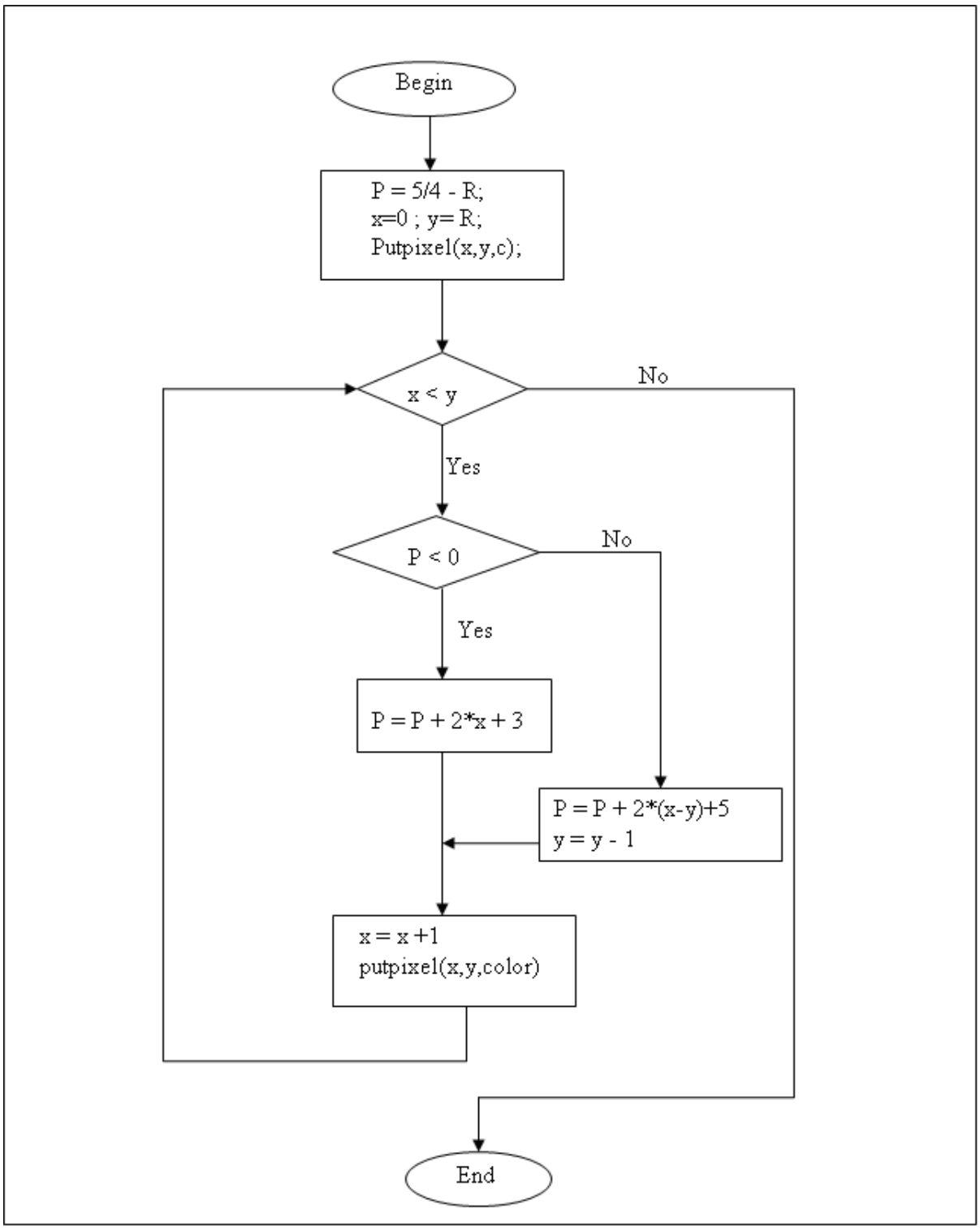
- Nếu $P_i >= 0$: chọn $y_{i+1} = y_i - 1$. Khi đó $P_{i+1} = P_i + 2x_i - 2y_i +5$.

- P_i ứng với điểm ban đầu (x_0 , y_0) = (0,R) là:

$$P_0 = F(x_0 + 1, y_0 - 1/2) = F(1, R - 1/2) = \frac{5}{4} - R$$

Lưu đồ thuật toán MidPoint vẽ đường tròn

thuật toán vẽ đường tròn



Minh họa thuật toán MidPoint:

Procedure DTR(xc, yc, r, mau : integer);

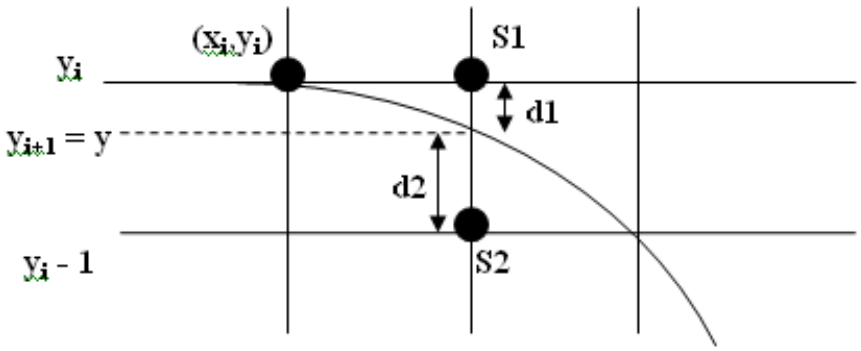
var x, y, p : integer ;

thuật toán vẽ đường tròn

```
begin
x:=0 ; y:=r;
p:=1 - r;
while ( y > x) do
begin
doi_xung;
if (p<0) then p:=p+2*x+3
else begin
p:=p+2*(x-y)+5 ;
y:=y-1;
end;
x:=x+1;
end; {while}
end;
```

Vẽ đường tròn bằng thuật toán Bresenham

Tương tự thuật toán vẽ đường thẳng Bresenham, các vị trí ứng với các tọa độ nguyên nằm trên đường tròn có thể tính được bằng cách xác định một trong hai pixel gần nhất với đường tròn thực hơn trong mỗi bước (xem hình 1.9).



Hình 1.9 : Đường tròn với khoảng cách d1 và d2.

Ta có :

$d1 = (y_i)^2 - y^2 = (y_i)^2 - (R^2 - (x_i + 1)^2)$

$d2 = y^2 - (y_i - 1)^2 = (R^2 - (x_i + 1)^2) - (y_i - 1)^2$

$P_i = d1 - d2$

Tính $P_{i+1} - P_i$

$\Rightarrow P_{i+1} = P_i + 4x_i + 6 + 2((y_{i+1})^2 - (y_i)^2) - 2(y_{i+1} - y_i)$

- Nếu $P_i < 0$: chọn $y_{i+1} = y_i$. Khi đó $P_{i+1} = P_i + 4x_i + 6$

- Nếu $P_i >= 0$: chọn $y_{i+1} = y_i - 1$. Khi đó $P_{i+1} = P_i + 4(x_i - y_i) + 10$.

- P_0 ứng với điểm ban đầu $(x_0, y_0) = (0,R)$ là: $P_0 = 3 - 2R$.

Minh họa thuật toán vẽ đường tròn bằng Bresenham

```
Procedure DTR_BRES(xc,yc,r,mau : integer);
var x,y,p:integer;
begin
x:=0 ; y:=r;
p:= 3 – 2*r ;
while ( x<y ) do
begin
doi_xung;
if (p<0) then p:= p + 4*x + 6
else begin
p:= p + 4*(x-y) + 10 ;
```

y:=y-1;

end;

x:=x+1;

end;{while}

end;

Thuật toán vẽ Ellipse

Tương tự thuật toán vẽ đường tròn, sử dụng thuật toán Bresenham để vẽ, ta chỉ cần vẽ 1/4 ellipse, sau đó lấy đối xứng qua các trục tọa độ sẽ vẽ được toàn bộ ellipse.

Xét ellipse có tâm O, các bán kính là a và b, phương trình là :

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$$

Chọn tọa độ pixel đầu tiên cần hiển thị là $(x_i, y_i) = (0,b)$. Cần xác định pixel tiếp theo là (x_{i+1}, y_{i+1}). Ta có :

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

$d1 = (y_i)^2 - y^2$

$d2 = y^2 - (y_i - 1)^2$

$P_i = d1 - d2$

Tính $P_{i+1} - P_i$

$$\Rightarrow \quad P_{i+1} = P_i + 2((y_{i+1})^2 - (y_i)^2) - 2(y_{i+1} - y_i) + \frac{2b^2}{a^2}(2x_i + 3)$$

- Nếu $P_i < 0$: chọn $y_{i+1} = y_i$. Khi đó - Nếu $P_i >= 0$: chọn $y_{i+1} = y_i - 1$. Khi đó

- P_i ứng với điểm ban đầu (x_0 , y_0) = (0,b) là: P_0 =

$$\frac{2b^2}{a^2}$$

- 2b + 1

$$P_{i+1} = P_i + \frac{2b^2}{a^2}(2x_i + 3) + 4(1 - y_i)$$

Minh họa thuật toán vẽ Ellipse

Procedure Ellipse(xc,yc,a,b : integer);

var x,y : integer;

z1, z2, P : real;

procedure dx;

begin

putpixel (x_c + x , y_c +y, color) ;

putpixel (x_c - x , y_c + y, color) ;

putpixel (x_c + x , y_c - y, color) ;

putpixel (x_c - x , y_c- y, color) ;

end;

begin

x:=0 y:=b;

z1:= (b*b)/(a*a);

z2:= 1/ z1;

P:= 2*z1 - 2*b +1;

while (z1* (x/y) ≤ 1) do

thuật toán vẽ đường tròn

```
begin
dx;
if P < 0 then P:= P + 2*z1*(2*x+3)
else begin
P:= P + 2*z1*(2*x+3) + 4*(1-y);
y:= y -1;
end;
x:= x+1;
end;
x:=a ; y:= 0;
P:= 2*z2 - 2*a +1;
while (z2* (y/x) < 1) do
begin
dx;
if P < 0 then P:= P + 2*z2*(2*y+3)
else begin
P:= P + 2*z2*(2*y+3) + 4*(1-x);
x:= x -1;
end;
y:= y +1;
end;
end;
```

Vẽ đường conics và một số đường cong khác

Phương trình tổng quát của các đường conics có dạng :

$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$

Giá trị của các hằng số A, B, C, D, E, F sẽ quyết định dạng của đường conics, cụ thể là nếu:

$B^2 - 4AC < 0$: dạng đường tròn (nếu A=C và B=0) hay ellipse.

$B^2 - 4AC = 0$: dạng parabol.

$B^2 - 4AC > 0$: dạng hyperbol.

Áp dụng ý tưởng của thuật toán Midpoint để vẽ các đường conics và một số đường cong khác theo các bước theo các bước tuần tự sau:

- Bước 1: Dựa vào dáng điệu và phương trình đường cong, để xem thử có thể rút gọn phần đường cong cần vẽ hay không.

- Bước 2: Tính đạo hàm, từ đó phân thành các vùng vẽ.

. Nếu $0 \le f'(x) \le 1$: $x_{i+1} = x_i + 1$; $y_{i+1} = y_i$ (hoặc = $y_i + 1$)

. Nếu $-1 \le f'(x) \le 0$: $x_{i+1} = x_i + 1$; $y_{i+1} = y_i$ (hoặc = $y_i - 1$)

. Nếu $f'(x) > 1$: $y_{i+1} = y_i + 1$; $x_{i+1} = x_i$ (hoặc = $x_i + 1$)

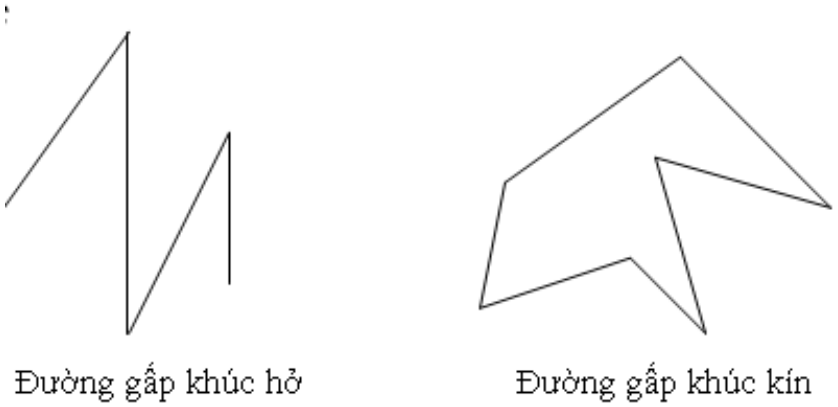
. Nếu $f'(x) < -1$: $y_{i+1} = y_i + 1$; $x_{i+1} = x_i$ (hoặc = $x_i + 1$)

- Bước 3 : Tính Pi cho từng trường hợp để quyết định f '(x) dựa trên dấu của Pi. Pi thường là hàm được xây dựng từ phương trình đường cong. Cho Pi=0 nếu (x_i , y_i) thuộc về đường cong. Việc chọn Pi cần chú ý sao cho các thao tác tínn Pi sau này hạn chế phép toán trên số thực.

- Bước 4 : Tìm mối liên quan của P_{i+1} và P_i bằng cách xét hiệu $P_{i+1} - P_i$

- Bước 5 : Tính P_0 và hoàn chỉnh thuật toán.

Vẽ đa giác



Hình 1.10 : Hai dạng của đường gấp khúc.

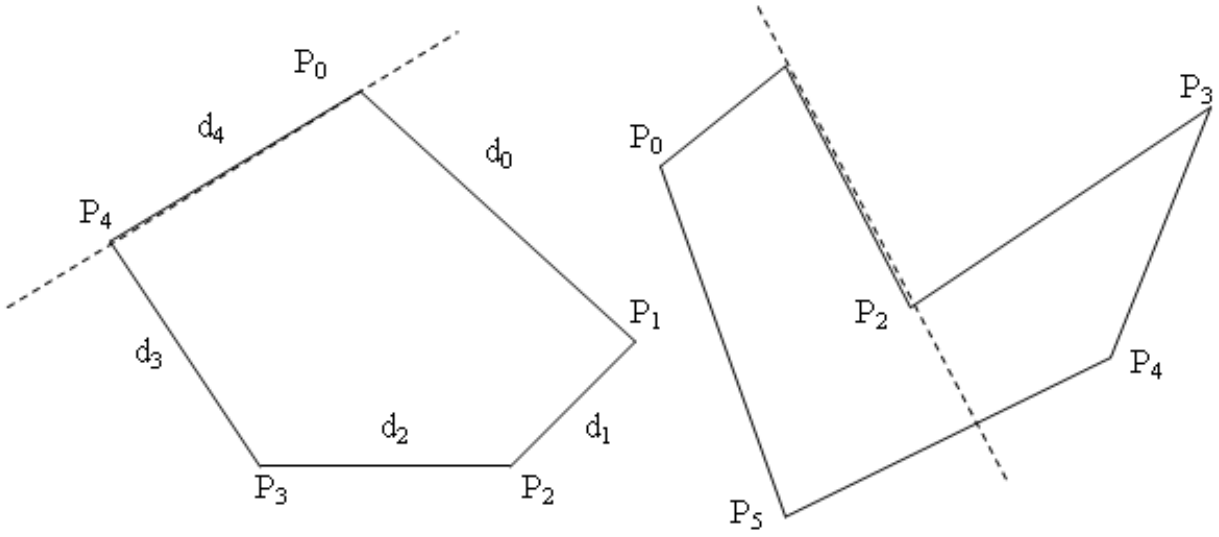
Định nghĩa đa giác (Polygone): Đa giác là một đường gấp khúc kín có đỉnh đầu và đỉnh cuối trùng nhau (xem hình 1.10)

Xây dựng cấu trúc dữ liệu để vẽ đa giác

```
Type

d_dinh = record

x,y: longint;

end;

dinh = array[0..10] of d_dinh;

var

d: dinh;
```

Với cách xây dựng cấu trúc dữ liệu như thế này thì chúng ta chỉ cần nhập vào tọa độ các đỉnh và sau đó gọi thủ tục vẽ đường thẳng lần lượt qua 2 đỉnh như (0, 1), (1,2), ..., (n-1, n), trong đó đỉnh n trùng với đỉnh 0 thì ta sẽ vẽ được toàn bộ đa giác.

Đa giác được gọi là lồi nếu bất kỳ đường thẳng nào đi qua một cạnh của đa giác thì toàn bộ đa giác nằm về một phía của đường thẳng đó. Ngược lại, nếu tồn tại ít nhất một cạnh của đa giác chia đa giác làm 2 phần thì gọi là **đa giác lõm** (xem hình 1.11).



Hình 1.11 : Đa giác lồi và đa giác lõm

Thuật toán kiểm tra một đa giác là lồi hay lõm

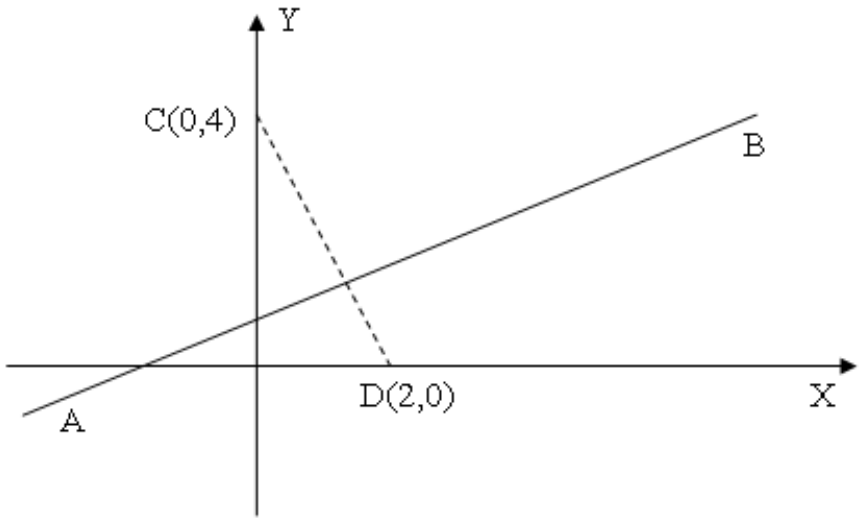
Thuật toán 1: Lần lượt thiết lập phương trình đường thẳng đi qua các cạnh của đa giác. Ứng với từng phương trình đường thẳng, xét xem các đỉnh còn lại có nằm về một phía đối với đường thẳng đó hay không ? Nếu đúng thì kết luận đa giác lồi, ngược lại là đa giác lõm.

Nhận xét : Phương trình đường thẳng y = ax + b chia mặt phẳng ra làm 2 phần. Các điểm nằm C(xc,yc) trên đường thẳng sẽ có yc > axc + b và các điểm D(xd,yd) nằm phía dưới đường thẳng sẽ có yd < axd + b.

Ví dụ : Cho đường thẳng AB có phương trình

$$y = \frac{1}{2}x + 1$$

và hai điểm C, D có tọa độ là C(0,4), D(2,0) (xem hình 1.12).



Hình 1.12 : Đường thẳng AB và 2 điểm C, D.

Ta có :

$$Y_c = 4 > ax_c + b = \frac{1}{2}.0 + 1$$

$$Y_d = 0 < ax_d + b = \frac{1}{2}.2 + 1$$

và Vậy hai điểm C, D nằm về hai phía đối với đường thẳng AB.

Thuật toán 2 :

Nhận xét :

Trong mặt phẳng Oxy, cho 2 véc tơ

$$\vec{a}$$

và

$$\vec{b}$$

, Tích vô hướng của 2 véc tơ là :

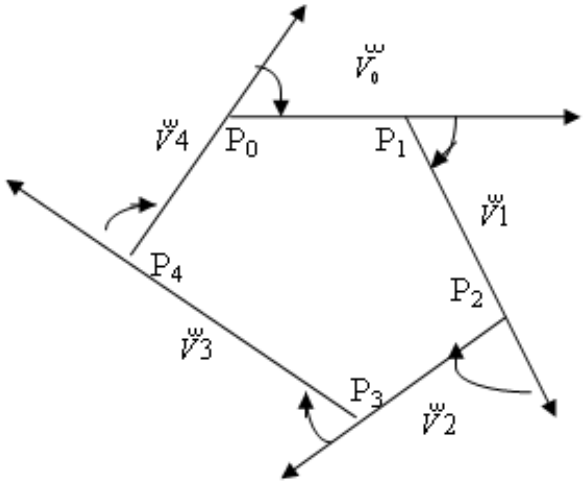
$$T(\vec{a}, \vec{b}) = \begin{vmatrix} a_x & a_y \\ b_x & b_y \end{vmatrix} = a_x * b_y - a_y * b_x$$

Khi đó :

$\vec{a}$ qu?o trái sang $\vec{b}$ n?u T ?0

$\vec{a}$ qu?o ph?i sang $\vec{b}$ n?u T < 0

Một đa giác là lồi khi đi dọc theo biên của nó thì chỉ đi theo một hướng mà thôi. Nghĩa là chỉ quẹo phải hay quẹo trái. Ngược lại là đa giác lõm (xem hình 1.13).



Hình 1.13 : Đa giác lồi có 5 đỉnh.

Xét đa giác gồm các đỉnh P_0, P_1 ,.... P_n , ($P_0 = P_n$) , $n \geq 3$ (xem hình 1.13).

Tính $V_i = P_{i+1} - P_i$, mọi i = 0, 1, ..., n-1.

Tính $T_i = T(V_i , V_{i+1})$

Nếu với mọi T_i đều cùng dấu thì kết luận đa giác lồi.

Ngược lại, là đa giác lõm.

Tổng kết chương 1

- Chương 1 đã trình bày khái niệm về một hệ độ họa, sự hiển thị của điểm trên màn hình với tọa độ phải là số nguyên.

- Phân biệt thế nào là hệ tọa độ thế giới thực, hệ tọa độ thiết bị và hệ tọa độ chuẩn.

- Cần lưu ý về hệ số góc của đường thẳng. Bởi vì, với hệ số góc khác nhau thì giải thuật có thay đổi. Nhất là trong giải thuật Bresenham.

- Chú ý hơn trong cách xây dựng cấu trúc dữ liệu để lưu tọa độ của các đỉnh đa giác.

- So sánh các trường hợp sử dụng công thức của các đường cong (có tham số và không có tham số).

Bài tập chương 1

Viết chương trình vẽ bầu trời có 10.000 điểm sao, mỗi điểm sao xuất hiện với một màu ngẫu nhiên. Những điểm sao này hiện lên rồi từ từ tắt cũng rất ngẫu nhiên.

Viết chương trình thực hiện 2 thao tác sau :

- Khởi tạo chế độ đồ họa, đặt màu nền, đặt màu chữ, định dạng chữ (settextstyle(f,d,s)), xuất một chuỗi ký tự ra màn hình. Đổi font, hướng, kích thước.

- Xuất một chuỗi ra màn hình, chuỗi này có tô bóng.

(lưu ý rằng nội dung chuỗi ký tự, màu tô, màu bóng là được nhập từ bàn phím).

Viết chương trình vẽ đoạn thẳng AB với màu color theo giải thuật DDA. Biết rằng tọa độ A,B, color được nhập từ bàn phím. Trang trí màu nền, ghi chú các tọa độ A, B ở hai đầu đoạn thẳng.

Tương tự như bài tập 3 nhưng sử dụng giải thuật Bresenham. Lưu ý các trường hợp đặc biệt của hệ số góc.

Tổng hợp bài tập 4, viết chương trình vẽ đường thẳng bằng giải thuật Bresenham cho tất cả các trường hợp của hệ số góc. Lưu ý xét trường hợp đặc biệt khi đường thẳng song song với trục tung hay với trục hoành.

Viết chương trình nhập tọa độ 3 điểm A, B, C từ bàn phím. Tìm tọa độ điểm D thuộc AB sao cho CD vuộng góc AB. Vẽ đoạn thẳng AB và CD.

Viết chương trình xét vị trí tương đối của 2 đoạn thẳng AB và CD. Biết rằng trong màn hình đồ họa đoạn thẳng AB và CD được gọi là cắt nhau khi hai điểm A, B ở về hai phía của CD và ngược lại.

Viết chương trình vẽ đường tròn theo giải thuật đơn giản (đối xứng).

Viết chương trình vẽ đường tròn theo giải thuật Bresenham.

Viết chương trình vẽ đường tròn theo giải thuật MidPoint.

Viết chương trình vẽ một đường tròn tâm O bán kính R. Vẽ các đường tròn đồng tâm với O, có bán kính chạy từ 1 đến R. Sau đó xoá các đường tròn đồng tâm này và vẽ các đường tròn đồng tâm khác đi từ R đến 1.

iết chương trình vẽ một đường tròn tâm O bán kính R. Hãy vẽ một đoạn thẳng từ tâm O độ dài R. Hãy quay đoạn thẳng này quanh đường tròn.

Viết chương trình vẽ Elippse.

Viết chương trình vẽ Elippse có bán kính lớn là a, bán kính nhỏ là b và một đường tròn nội tiếp Elippse. Tô đường tròn bằng các đường tròn đồng tâm. Sau đó tô elippse bằng các elippse đồng tâm có bán kính lớn chạy từ b đến a, bán kính nhỏ là b.

Viết chương trình vẽ một hình chữ nhật, một hình vuông và một hình bình hành. Yêu cầu chú thích tọa độ các đỉnh.

Viết chương trình vẽ một tam giác. Tọa độ các đỉnh được nhập từ bàn phím, mỗi cạnh có một màu khác nhau.

Viết chương trình vẽ một đa giác có n đỉnh.

Viết chương trình xét tính lồi lõm của một đa giác bằng cách thiết lập phương trình đường thẳng đi qua các cạnh của đa giác.

Viết chương trình xét tính lồi lõm của một đa giác bằng cách thiết lập các véc tơ chỉ phương của các cạnh.