Bresenham

#include<graphics.h>

#include<conio.h>

#include<stdio.h>

#include<dos.h>

void main(){

int x, y, x1, y1, x2, y2, p, dx, dy;

int gd=DETECT,gm;

initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");

printf("\nEnter the x coordinate of the first point: ");

scanf("%d",&x1);

printf("\nEnter the y coordinate of the first point: ");

scanf("%d",&y1);

printf("\nEnter the x coordinate of the second point: ");

scanf("%d",&x2);

printf("\nEnter the y co ordinate of the second point: ");

scanf("%d",&y2);

x=x1;

y=y1;

dx=x2-x1;

dy=y2-y1;

putpixel(x,y,2);

p=(2\*dy-dx);

while(x<=x2)

{

if(p<0)

{

x=x+1;

p=p+2\*dy;

}

else

{

x=x+1;

y=y+1;

p=p+(2\*dy)-(2\*dx);

}

putpixel(x,y,7);

}

getch();

closegraph();

}

Midpoint 3rd

#include<dos.h>

#include <stdio.h>

#include <graphics.h>

#include<conio.h>

void draw\_circle(int,int,int);

void symmetry(int,int,int,int);

void main()

{

int xc,yc,R;

int gd=DETECT,gm;

initgraph(&gd,&gm,"C:\\TurboC3\\BGI");

printf("Enter the centre of the circle:\n");

printf("Xc=");

scanf("%d",&xc);

printf("yc=");

scanf("%d",&yc);

printf("enter the radius of the circle:\n");

scanf("%d",&R);

draw\_circle(xc,yc,R);

getch();

closegraph();

}

void draw\_circle(int xc,int yc,int rad)

{

int x=0;

int y=rad;

int p=1-rad;

symmetry(x,y,xc,yc);

for(x=0;y>x;x++)

{

if(p<0)

p +=2\*x+3;

else

{

p+= 2\*(x-y)+5;

y--;

}

symmetry(x,y,xc,yc);

delay(10);

}

}

void symmetry(int x, int y,int xc, int yc)

{

putpixel(xc+x,yc-y,GREEN);

delay(50);

putpixel(xc+y,yc-x,GREEN);

delay(50);

putpixel(xc+y,yc+x,GREEN);

delay(50);

putpixel(xc+x,yc+y,GREEN);

delay(50);

putpixel(xc-x,yc+y,GREEN);

delay(50);

putpixel(xc-y,yc+x,GREEN);

delay(50);

putpixel(xc-y,yc-x,GREEN);

delay(50);

putpixel(xc-x,yc-y,GREEN);

delay(50);

}

Scanline 4th

#include <conio.h>

#include <graphics.h>

void main()

{

int n,i,j,k,gd,gm,dy,dx;

int x,y,temp;

int a[20][2],xi[20];

float slope[20];

clrscr();

printf("\n\n\t Enter the no. of edges of polygon: ");

scanf("%d",&n);

printf("\n\n\t Enter the coodinates of polygon :\n\n\n");

for(i=0;i<n;i++)

{

printf("\tX%d Y%d:",i,i);

scanf("%d %d",&a[i][0],&a[i][1]);

}

a[n][0]=a[0][0];

a[n][1]=a[0][1];

detectgraph(&gd,&gm);

initgraph(&gd,&gm,"C:\\TurboC3\\BGI");

for(i=0;i<n;i++)

{

line(a[i][0],a[i][1],a[i+1][0],a[i+1][1]);

}

getch();

for(i=0;i<n;i++)

{

dy=a[i+1][1]-a[i][1];

dx=a[i+1][0]-a[i][0];

if(dy==0) slope[i]=1.0;

if(dx==0) slope[i]=0.0;

if((dy!=0)&&(dx!=0))

{

slope[i]=(float) dx/dy;

}

}

for(y=0;y<480;y++)

{

k=0;

for(i=0;i<n;i++)

{

if( ((a[i][1]<=y)&&(a[i+1][1]>y))||((a[i][1]>y)&&(a[i+1][1]<=y)))

{

xi[k]=(int)(a[i][0]+slope[i]\*(y-a[i][1]));

k++;

}

}

for(j=0;j<k-1;j++)

for(i=0;i<k-1;i++)

{

if(xi[i]>xi[i+1])

{

temp=xi[i];

xi[i]=xi[i+1];

xi[i+1]=temp;

}

}

setcolor(3);

for(i=0;i<k;i+=2)

{

line(xi[i],y,xi[i+1]+1,y);

getch();

}

}

}

bezier

#include <stdio.h> #include <stdlib.h> #include <graphics.h> #include <math.h>

void bezier (int x[4], int y[4])

{

int gd = DETECT, gm; int i;

double t;

initgraph (&gd, &gm, "C:\\TurboC3\\BGI"); for (t = 0.0; t < 1.0; t += 0.0005)

{

double xt = pow (1-t, 3) \* x[0] + 3 \* t \* pow (1-t, 2) \* x[1] +3 \* pow (t, 2) \* (1-t) \* x[2] + pow (t, 3) \* x[3];

double yt = pow (1-t, 3) \* y[0] + 3 \* t \* pow (1-t, 2) \* y[1] +3 \* pow (t, 2) \* (1-t) \* y[2] + pow (t, 3) \* y[3];

putpixel (xt, yt, WHITE);

}

for (i=0; i<4; i++)

putpixel (x[i], y[i], YELLOW); getch();

closegraph(); return;

}

void main()

{

int x[4], y[4]; int i;

printf ("Enter the x- and y-coordinates of the four control points.\n");

for (i=0; i<4; i++)

scanf ("%d%d", &x[i], &y[i]); bezier (x, y);

}

Koch

#include<graphics.h>

#include<conio.h>

#include<math.h>

void koch(int x1, int y1, int x2, int y2, int it)

{

float angle = 60\*M\_PI/180;

int x3 = (2\*x1+x2)/3;

int y3 = (2\*y1+y2)/3;

int x4 = (x1+2\*x2)/3;

int y4 = (y1+2\*y2)/3;

int x = x3 + (x4-x3)\*cos(angle)+(y4-y3)\*sin(angle);

int y = y3 - (x4-x3)\*sin(angle)+(y4-y3)\*cos(angle);

if(it > 0)

{

koch(x1, y1, x3, y3, it-1);

koch(x3, y3, x, y, it-1);

koch(x, y, x4, y4, it-1);

koch(x4, y4, x2, y2, it-1);

}

else

{

line(x1, y1, x3, y3);

line(x3, y3, x, y);

line(x, y, x4, y4);

line(x4, y4, x2, y2);

}

}

int main(void)

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\TURBOC3\\BGI"); int x1 = 100, y1 = 100, x2 = 400, y2 = 400;

koch(x1, y1, x2, y2, 4); getch();

return 0;

}