## //bezierCurve.c

#include <stdio.h>

#include <graphics.h>

#include <math.h>

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

{

for (double 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, RED);

}

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

putpixel (x[i], y[i], RED);

}

int main() {

int gd = DETECT, gm = DETECT;

initgraph (&gd, &gm, "");

setbkcolor(WHITE);

/\* Four control points \*/

int x[4], y[4];

x[0] = 10, y[0] = 10;

x[1] = 100, y[1] = 150;

x[2] = 20, y[2] = 40;

x[3] = 400, y[3] = 100;

bezier (x, y);

delay(5000);

closegraph();

return 0;

}

## //****bresenhamCircle.c****

#include <graphics.h>

void plot8CirclePoints(int ox, int oy, int x, int y) {

putpixel(ox + x, oy + y, 1);

putpixel(ox + y, oy + x, 2);

putpixel(ox - x, ox + y, 3);

putpixel(ox - y, ox + x, 4);

putpixel(ox - x, oy - y, 5);

putpixel(ox - y, oy - x, 6);

putpixel(ox + x, oy - y, 7);

putpixel(ox + y, oy - x, 8);

}

void plotTriangle(int ox, int oy, int r) {

int x = r, y = 0, re = 0;

int xc = 1 - 2 \* r;

int yc = 1;

while(x >= y) {

plot8CirclePoints(ox, oy, x, y);

y++;

re += yc;

yc += 2;

if(2 \* re + xc > 0) {

x--;

re += xc;

xc += 2;

}

}

}

int main() {

int gd=DETECT, gm=DETECT;

initgraph(&gd,&gm,"");

setbkcolor(WHITE);

int ox = 250, oy = 250, r = 200;

plotTriangle(ox, oy, r);

delay(5000);

closegraph();

return 0;

}

## //****bresenhamLine.c****

#include <graphics.h>

void plotLine(int x0, int y0, int x1, int y1) {

int dx = x1 - x0;

int dy = y1 - y0;

int y = y0;

int eps = 0;

for(int x = x0; x <= x1; x++) {

putpixel(x, y, RED);

eps += dy;

if((eps << 1) >= dx) {

y++;

eps -= dx;

}

}

}

int main() {

int gd=DETECT, gm=DETECT;

initgraph(&gd,&gm,"");

setbkcolor(WHITE);

plotLine(3, 30, 500, 105);

delay(5000);

closegraph();

return 0;

}

## ****//cohenSutherlandLineClipping.c****

#include <graphics.h>

const int LEFT = 1, RIGHT = 2, BOTTOM = 4, TOP = 8;

double xmin, xmax, ymin, ymax;

int ComputeOutCode(double x, double y)

{

int code = 0;

if (x < xmin) // to the left of clip window

code |= LEFT;

else if (x > xmax) // to the right of clip window

code |= RIGHT;

if (y < ymin) // below the clip window

code |= BOTTOM;

else if (y > ymax) // above the clip window

code |= TOP;

return code;

}

void CohenSutherlandLineClipAndDraw(double x0, double y0, double x1, double y1)

{

int outcode0 = ComputeOutCode(x0, y0);

int outcode1 = ComputeOutCode(x1, y1);

int accept = 0;

while (1) {

if (!(outcode0 | outcode1)) {

accept = 1;

break;

} else if (outcode0 & outcode1) {

break;

} else {

double x, y;

int outcodeOut = outcode0 ? outcode0 : outcode1;

if (outcodeOut & TOP) {

x = x0 + (x1 - x0) \* (ymax - y0) / (y1 - y0);

y = ymax;

} else if (outcodeOut & BOTTOM) {

x = x0 + (x1 - x0) \* (ymin - y0) / (y1 - y0);

y = ymin;

} else if (outcodeOut & RIGHT) {

y = y0 + (y1 - y0) \* (xmax - x0) / (x1 - x0);

x = xmax;

} else if (outcodeOut & LEFT) {

y = y0 + (y1 - y0) \* (xmin - x0) / (x1 - x0);

x = xmin;

}

if (outcodeOut == outcode0) {

x0 = x;

y0 = y;

outcode0 = ComputeOutCode(x0, y0);

} else {

x1 = x;

y1 = y;

outcode1 = ComputeOutCode(x1, y1);

}

}

}

if (accept) {

setcolor(RED);

rectangle(xmin, ymin, xmax, ymax);

setcolor(BLUE);

line(x0, y0, x1, y1);

}

}

int main() {

int gd = DETECT, gm = DETECT;

initgraph(&gd, &gm, "");

setbkcolor(WHITE);

double x0, y0, x1, y1;

xmin = 100;

xmax = 500;

ymin = 10;

ymax = 300;

x0 = 50, y0 = 50, x1 = 600, y1 = 90;

CohenSutherlandLineClipAndDraw(x0, y0, x1, y1);

delay(5000);

closegraph();

return 0;

}

## ****//drawingObjects.c****

#include <graphics.h>

int main() {

int gd=DETECT, gm=DETECT;

initgraph(&gd,&gm,"");

setbkcolor(WHITE);

setcolor(BLACK);

line(50, 30, 400, 45);

circle(250, 100, 50);

rectangle(100, 100, 400, 400);

delay(5000);

closegraph();

return 0;

}