Set A:

//Bresenham’s line drawing algo

#include<graphics.h>

#include<conio.h>

#include<math.h>

int main(){

int gd=DETECT;

int gm;

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

int x1=150,y1=150,x2=370,y2=287;

int dx=abs(x2-x1), dy=abs(y2-y1);

int x=x1, y=y1;

putpixel(x,y,WHITE);

if(dx > dy){

int ds=2\*dy;

int dt=2\*(dy-dx);

int d=2\*dy-dx;

for(int i=0;i<dx;i++){

x++;

if(d<0){

d=d+ds;

}

else{

d=d+dt;

y++;

}

putpixel(x,y,WHITE);

printf("%d %d\n",x,y);

}

}

else{

int ds=2\*dx;

int dt=2\*(dx-dy);

int d=2\*dx-dy;

for(int i=0;i<dy;i++){

y++;

if(d<0){

d=d+ds;

}

else{

d=d+dt;

x++;

}

putpixel(x,y,WHITE);

printf("%d %d\n",x,y);

}

}

getch();

closegraph();

}

Set B:

//Mid point circle algo

#include<stdio.h>

#include<graphics.h>

void drawcircle(int x0, int y0, int radius)

{

int x = radius;

int y = 0;

int err = 0;

while (x >= y)

{

putpixel(x0 + x, y0 + y, 7);

putpixel(x0 + y, y0 + x, 7);

putpixel(x0 - y, y0 + x, 7);

putpixel(x0 - x, y0 + y, 7);

putpixel(x0 - x, y0 - y, 7);

putpixel(x0 - y, y0 - x, 7);

putpixel(x0 + y, y0 - x, 7);

putpixel(x0 + x, y0 - y, 7);

if (err <= 0)

{

y += 1;

err += 2\*y + 1;

}

if (err > 0)

{

x -= 1;

err -= 2\*x + 1;

}

}

}

int main()

{

int gdriver=DETECT, gmode, error, x, y, r;

initgraph(&gdriver, &gmode, "c:\\turboc3\\bgi");

printf("Enter radius of circle: ");

scanf("%d", &r);

printf("Enter co-ordinates of center(x and y): ");

scanf("%d%d", &x, &y);

drawcircle(x, y, r);

getch();

return 0;

}

Set C:

//Translation of circle

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

int main()

{ int gd,gm;

gd=DETECT;

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

int x,y,r,tx,ty;

printf("Enter the coordinates of circle:");

scanf("%d %d",&x,&y);

printf("\nEnter the radius of circle:");

scanf("%d",&r);

circle(x,y,r);

outtextxy(x+30,y,"Original Circle");

printf("\nEnter the translation units:");

scanf("%d %d",&tx,&ty);

circle(x+tx,y+ty,r);

outtextxy(x+tx+30,x+ty,"Translated Circle");

getch();

return 0;

closegraph();

}

Set D:

//Rotation of line

#include<graphics.h>

#include<stdio.h>

#include<conio.h>

#include<math.h>

int main()

{

int gd=DETECT,gm;

int pivot\_x,pivot\_y,x,y;

double degree,radian;

int rotated\_point\_x,rotated\_point\_y;

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

cleardevice();

printf("\t\t\*\*\*\*\* ROTATION \*\*\*\*\* \n");

printf("\n Enter an initial coordinates of the line = ");

scanf("%d %d",&pivot\_x,&pivot\_y);

printf("\n Enter a final coordinates of the line = ");

scanf("%d %d",&x,&y);

line(pivot\_x,pivot\_y,x,y);

printf("\n\n Now, Enter a degree = ");

scanf("%lf",&degree);

radian=degree\*0.01745;

rotated\_point\_x=(int)(pivot\_x +((x-pivot\_x)\*cos(radian)-(y-pivot\_y)\*sin(radian)));

rotated\_point\_y=(int)(pivot\_y +((x-pivot\_x)\*sin(radian)+(y-pivot\_y)\*cos(radian)));

setcolor(RED);

line(pivot\_x,pivot\_y,rotated\_point\_x,rotated\_point\_y);

getch();

closegraph();

}

Set E:

//Scaling of circle

#include<conio.h>

#include<graphics.h>

int main()

{ int gd,gm;

gd=DETECT;

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

int x,y,r;

float sx,sy;

printf("Program for scaling a circle-");

printf("\nEnter the coordinates of a circle:");

scanf("%d %d",&x,&y);

printf("\nEnter the radius of a circle:");

scanf("%d",&r);

circle(x,y,r);

outtextxy(x+20,y,"Original Circle");

printf("\nEnter the scaling units:");

scanf("%f %f",&sx,&sy);

circle(x\*sx,y\*sx,r);

outtextxy((x\*sx),(y\*sy),"Circle after Scaling");

getch();

return 0;

closegraph();

}

Set F:

//Reflection all points

#include <conio.h>

#include <graphics.h>

#include <stdio.h>

int main()

{

int gm, gd = DETECT, ax, x1 = 100;

int x2 = 100, x3 = 200, y1 = 100;

int y2 = 200, y3 = 100;

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

cleardevice();

// Draw the graph

line(getmaxx() / 2, 0, getmaxx() / 2,

getmaxy());

line(0, getmaxy() / 2, getmaxx(),

getmaxy() / 2);

// Object initially at 2nd quadrant

printf("Before Reflection Object"

" in 2nd Quadrant");

setcolor(14);

line(x1, y1, x2, y2);

line(x2, y2, x3, y3);

line(x3, y3, x1, y1);

getch();

// After reflection

printf("\nAfter Reflection");

// Reflection along origin i.e.,

// in 4th quadrant

setcolor(4);

line(getmaxx() - x1, getmaxy() - y1,

getmaxx() - x2, getmaxy() - y2);

line(getmaxx() - x2, getmaxy() - y2,

getmaxx() - x3, getmaxy() - y3);

line(getmaxx() - x3, getmaxy() - y3,

getmaxx() - x1, getmaxy() - y1);

// Reflection along x-axis i.e.,

// in 1st quadrant

setcolor(3);

line(getmaxx() - x1, y1,

getmaxx() - x2, y2);

line(getmaxx() - x2, y2,

getmaxx() - x3, y3);

line(getmaxx() - x3, y3,

getmaxx() - x1, y1);

// Reflection along y-axis i.e.,

// in 3rd quadrant

setcolor(2);

line(x1, getmaxy() - y1, x2,

getmaxy() - y2);

line(x2, getmaxy() - y2, x3,

getmaxy() - y3);

line(x3, getmaxy() - y3, x1,

getmaxy() - y1);

getch();

closegraph();

}

Set G:

//Shearing in 2D

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

int main()

{

int gd,gm;

gd=DETECT;

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

int shx,shy;

line(80,80,80,140);

line(80,140,140,140);

line(140,140,140,80);

line(140,80,80,80);

outtextxy(60,60,"Original Square");

printf("Program for Shearing in 2D-");

printf("\nEnter the shearing factors:");

scanf("%d %d",&shx,&shy);

line(80+80\*shx,80,80+140\*shx,140);

line(80+140\*shx,140,140+140\*shx,140);

line(140+140\*shx,140,140+80\*shx,80);

line(140+80\*shx,80,80+80\*shx,80);

outtextxy(230,60,"Square after shearing relative to x-axis");

line(80,80+80\*shy,80,140+80\*shy);

line(80,140+80\*shy,140,140+140\*shy);

line(140,140+140\*shy,140,80+140\*shy);

line(140,80+140\*shy,80,80+80\*shy);

outtextxy(60,230,"Square after shearing relative to y-axis");

getch();

return 0;

closegraph();

}

Set H:

//Rotate a coin

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

#include<dos.h>

void coin()

{

while(!kbhit())

{

for(int i=20;i>=0;i--)

{

fillellipse(230,100,i,20);

setfillstyle(5,3);

delay(50);

cleardevice();

}

for(int j=0;j<=20;j++)

{

fillellipse(230,100,j,20);

setfillstyle(5,3);

delay(50);

cleardevice();

}

}

}

int main()

{

int gd,gm;

detectgraph(&gd,&gm);

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

coin();

getch();

closegraph();

}

Set I:

//Rotate a coin on table

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

#include<dos.h>

void coin()

{

while(!kbhit())

{

for(int i=20;i>=0;i--)

{

rectangle(100,180,300,150);

fillellipse(230,100,i,20);

setfillstyle(5,3);

line(150,70,100,150);

line(150,70,330,70);

line(330,70,300,150);

line(301,180,330,100);

line(330,100,330,70);

line(100,180,100,300);

line(100,300,120,300);

line(120,300,120,180);

line(300,150,300,300);

line(300,300,280,300);

line(280,300,280,180);

line(160,180,160,250);

line(160,250,180,250);

line(180,250,180,180);

line(330,100,330,250);

line(330,250,310,250);

line(310,250,310,155);

delay(500);

cleardevice();

}

for(int j=0;j<=20;j++)

{

rectangle(100,180,300,150);

fillellipse(230,100,j,20);

setfillstyle(5,3);

line(150,70,100,150);

line(150,70,330,70);

line(330,70,300,150);

line(301,180,330,100);

line(330,100,330,70);

line(100,180,100,300);

line(100,300,120,300);

line(120,300,120,180);

line(300,150,300,190);

delay(500);

cleardevice();

}

}

}

int main()

{

int gd,gm;

detectgraph(&gd,&gm);

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

coin();

getch();

closegraph();

}

Set J:

//Flying Balloons

#include<stdio.h>

#include<graphics.h>

#include<conio.h>

#include<dos.h>

int main(){

int gd=DETECT;

int gm;

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

while(!kbhit()){

for(int i=610;i>-155;i--){

setcolor(RED);

setfillstyle(HATCH\_FILL,RED);

fillellipse(200,i,50,70);

setcolor(BLUE);

setfillstyle(LTSLASH\_FILL,BLUE);

fillellipse(350,i-100,50,70);

setcolor(GREEN);

setfillstyle(CLOSE\_DOT\_FILL,GREEN);

fillellipse(500,i-200,50,70);

setcolor(RED);

arc(200,i+90,90,270,20);

arc(200,i+130,270,450,20);

setcolor(BLUE);

arc(350,i+90-100,90,270,20);

arc(350,i+130-100,270,450,20);

setcolor(GREEN);

arc(500,i+90-200,90,270,20);

arc(500,i+130-200,270,450,20);

delay(10);

cleardevice();

}

}

getch();

closegraph();

}