Pract 1b- Draw co ordinate axis at the center.

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int gd=DETECT,gm;

int midx,midy;

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

cleardevice();

midx=getmaxx()/2;

midy=getmaxy()/2;

line(1,midy,640,midy);

line(midx,1,midx,480);

outtextxy(50,300,"VIRAJ");

getch();

}

Pract 2a- Draw circle, rectangle,elipse,half-elipse.

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int gd=DETECT,gm;

int midx,midy;

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

cleardevice();

midx=getmaxx()/2;

midy=getmaxy()/2;

line(1,midy,640,midy);

line(midx,1,midx,480);

setcolor(WHITE);

circle(midx+(-150),midy-(120),40);

printf("\t\tIt's a circle");

setcolor(GREEN);

rectangle(midx+(100),midy-(100),midx+(200),midy-(150));

printf("\t\t\t\tThis is rectangle\n\n\n\n\n");

setcolor(BLUE);

ellipse(midx+(-150),midy-(-100),0,360,midx+(-250),midy-(200));

printf("\n\n\n\n\n\n\n\n\n\n\n\t wow!! It's an ellipse");

setcolor(YELLOW);

ellipse(midx+(180),midy-(-100),180,0,midx+(-200),midy-(150));

printf("\t\t\t ohhh!!half ellipse! looks like:");

outtextxy(100,50,"VIRAJ");

getch();

}

Pract 2b1- Simple hut.

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int gd=DETECT,gm;

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

 setcolor(WHITE);

rectangle(150,180,250,300);

rectangle(250,180,420,300);

rectangle(180,250,220,300);

outtextxy(350,200,"VIRAJ");

line(200,100,150,180);

line(200,100,250,180);

line(200,100,370,100);

line(370,100,420,180);

getch();

closegraph();

}

Pract 2b2- Colourful hut.

#include<graphics.h>

#include<stdio.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm;

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

setcolor(WHITE);

rectangle(150,180,250,300);

rectangle(250,180,420,300);

rectangle(180,250,220,300);

outtextxy(350,200,"VIRAJ");

line(200,100,150,180);

line(200,100,250,180);

line(200,100,370,100);

line(370,100,420,180);

setfillstyle(SOLID\_FILL,BROWN);

floodfill(152,182,WHITE);

floodfill(252,182,WHITE);

setfillstyle(SLASH\_FILL,BLUE);

floodfill(182,252,WHITE);

setfillstyle(HATCH\_FILL,GREEN);

floodfill(200,105,WHITE);

floodfill(210,105,WHITE);

getch();

closegraph();

}

Pract 3- Basic shapes [circle ,rectangle , square , concentric circle , ellipse ,line].

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int gd=DETECT,gm,left=100,top=100,right=200,bottom=200,x=300,y=150,radius=50;

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

rectangle(120,150,230,200);

circle(x,y,radius);

bar(left+300,top,right+300,bottom);

line(left-10,top+150,left+410,top+150);

ellipse(x,y+200,0,360,80,50);

for(radius=25;radius<=100;radius=radius+20)

circle(500,350,radius);

outtextxy(275,150,"VIRAJ");

getch();

closegraph();

}

Pract 4a- DDA Line Algorithm.

#include<stdio.h>

#include<conio.h>

#include<math.h>

#include<dos.h>

#include<graphics.h>

void main()

{

float x,y,x1,y1,x2,y2,dx,dy,pixel;

int i,gd,gm;

printf("Enter the value of x1:");

scanf("%f",&x1);

printf("Enter the value of y1:");

scanf("%f",&y1);

printf("Enter the value of x2:");

scanf("%f",&x2);

printf("Enter the value of y2:");

scanf("%f",&y2);

detectgraph(&gd,&gm);

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

dx=abs(x2-x1);

dy=abs(y2-y1);

if(dx>=dy)

pixel=dx;

else

pixel=dy;

dx=dx/pixel;

dy=dy/pixel;

x=x1;

y=y1;

i=1;

while(i<=pixel)

{

putpixel(x,y,1);

x=x+dx;

y=y+dy;

i=i+1;

delay(100);

}

getch();

closegraph();

}

Pract 4b- Bresenhams Line Drawing Algorithm.

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

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

int gd,gm;

clrscr();

printf("\n\n\tEnter the co-ordinates of first point:");

scanf("%d%d",&x1,&y1);

printf("\n\n\tEnter the co-ordinates of second point:");

scanf("%d%d",&x2,&y2);

dx=(x2-x1);

dy=(y2-y1);

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

x=x1;

y=y1;

detectgraph(&gd,&gm);

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

putpixel(x,y,WHITE);

while(x<=x2)

{

if(p<0)

{

x=x+1;

y=y;

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

}

else

{

x=x+1;

y=y+1;

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

}

putpixel(x,y,WHITE);

}

getch();

closegraph();

}

Pract5a-midpoint circle drawing algorithm.

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void pixel(int xc,int yc,int x,int y);

void main()

{

int gd=DETECT,gm,xc,yc,r,x,y,pk;

clrscr();

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

printf("\*\*\*Bresenham's Midpoint Algorithm of circle\*\*\*\n");

printf("Enter the value of xc\t");

scanf("%d",&xc);

printf("Enter the value of yc\t");

scanf("%d",&yc);

printf("Enter the radius of circle\t");

scanf("%d",&r);

x=0;

y=r;

pk=1-r;

pixel(xc,yc,x,y);

while(x<y)

{

if(pk<0)

{

x=x+1;

pk=pk+(2\*x)+1;

}

else

{

x=x+1;

y=y-1;

pk=pk+(2\*x)-(2\*y)+1;

}

pixel(xc,yc,x,y);

}

getch();

closegraph();

}

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

{

putpixel(xc+x,yc+y,7);

putpixel(xc+y,yc+x,7);

putpixel(xc-y,yc+x,7);

putpixel(xc-x,yc+y,7);

putpixel(xc-x,yc-y,7);

putpixel(xc-y,yc-x,7);

putpixel(xc+y,yc-x,7);

putpixel(xc+x,yc-y,7);

}

Pract5b- mid point ellipse drawing algorithm

#include<conio.h>

#include<stdio.h>

#include<graphics.h>

#include<math.h>

void disp();

float x,y;

int xc,yc;

void main()

{

int gd=DETECT,gm;

int a,b;

float p1,p2;

clrscr();

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

printf("Enter xc:\t");

scanf("%d",&xc);

printf("Enter yc:\t");

scanf("%d",&yc);

printf("Enter a:\t");

scanf("%d",&a);

printf("Enter b:\t");

scanf("%d",&b);

x=0;y=b;

disp();

p1=(b\*b)-(a\*a\*b)+(a\*a)/4;

while((2.0\*b\*b\*x)<=(2.0\*a\*a\*y))

{

x++;

if(p1<=0)

p1=p1+(2.0\*b\*b\*x)+(b\*b);

else

{

y--;

p1=p1+(2.0\*b\*b\*x)+(b\*b)-(2.0\*a\*a\*y);

}

disp();

x=-x;

disp();

x=-x;

}

x=a;

y=0;

disp();

p2=(a\*a)+2.0\*(b\*b\*a)+(b\*b)/4 ;

while((2.0\*b\*b\*x)>(2.0\*a\*a\*y))

{

y++;

if(p2>0)

p2=p2+(a\*a)-(2.0\*a\*a\*y);

else

{

x--;

p2=p2+(2.0\*b\*b\*x)-(2.0\*a\*a\*y)+(a\*a);

}

disp();

y=-y;

disp();

y=-y;

}

getch();

closegraph();

}

void disp()

{

putpixel(xc+x,yc+y,10);

putpixel(xc-x,yc+y,10);

putpixel(xc+x,yc-y,10);

putpixel(xc+x,yc-y,10);

}

Pract6a-implementing 2D scaling

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int i;

int gd=DETECT,gm;

int x2,y2,x1,y1,x,y;

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

printf("Enter the 2 line end points:x1,y1,x2,y2:\n");

scanf("%d\n%d\n%d\n%d",&x1,&y1,&x2,&y2);

line(x1,y1,x2,y2);

printf("\nEnter scaling co-ordinates;x\t y\t \n");

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

x1=(x1\*x);

y1=(y1\*y);

x2=(x2\*x);

y2=(y2\*y);

printf("Line after scaling");

line(x1,y1,x2,y2);

getch();

closegraph();

}

Pract6b- to perform 2D translation.

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int i;

int gd=DETECT,gm;

int x2,y2,x1,y1,x,y;

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

printf("Enter the 2 line end points:x1,y1,x2,y2:\n");

scanf("%d\n%d\n%d\n%d",&x1,&y1,&x2,&y2);

line(x1,y1,x2,y2);

printf("\nEnter scaling co-ordinates;x\t y\t \n");

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

x1=x1+x;

y1=y1+y;

x2=x2+x;;

y2=y2+y;

printf("Line after translation");

line(x1,y1,x2,y2);

getch();

closegraph();

}

Pract7a- perform 2D rotation on a given object.

#include<stdio.h>

#include<conio.h>

#include<math.h>

#include<graphics.h>

void main()

{

int gd=DETECT,gm;

int i;

int x2,y2,x1,y1,x,y,xn,yn;

double r11,r12,r21,r22,th;

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

printf("Enter the 2 line end points x1,y1,x2,y2:\n");

scanf("%d%d%d%d",&x1,&y1,&x2,&y2);

line(x1,y1,x2,y2);

printf("\n\n\n Enter the angle:\t");

scanf("%lf",&th);

r11=cos((th\*3.1428)/180);

r12=sin((th\*3.1428)/180);

r21=(-sin((th\*3.1428)/180));

r22=cos((th\*3.1428)/180);

xn=((x2\*r11)-(y2\*r21));

yn=((x2\*r12)+(y2\*r22));

line(x1,y1,xn,yn);

getch();

closegraph();

}

Pract7b-create house like figure and perform

1. Scaling about the origin followed by translation
2. Scaling with reference to an arbitrary point.
3. Reflect about the line y=mx+c.

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<math.h>

#include<graphics.h>

void reset(int h[][2])

{

int val[9][2]={

{50,50},{75,50},{75,75},{100,75},{100,50},{125,50},{125,100},{87,125},{50,100}

};

int i;

for(i=0;i<9;i++)

{

h[i][0] = val[i][0]-50;

h[i][1] = val[i][1]-50;

}

}

void draw(int h[][2])

{

int i;

setlinestyle(DOTTED\_LINE,0,1);

line(320,0,320,480);

line(0,240,640,240);

setlinestyle(SOLID\_LINE, 0,1);

for(i=0;i<8;i++)

line(320+h[i][0],240-h[i][1],320+h[i+1][0],240-h[i+1][1]);

line(320+h[0][0],240-h[0][1],320+h[8][0],240-h[8][1]);

}

void rotate(int h[][2],float angle)

{

int i;

for(i=0;i<9;i++)

{

int xnew , ynew;

xnew = h[i][0]\*cos(angle)-h[i][1] \*sin(angle);

ynew = h[i][0] \*sin(angle)+ h[i][1]\*cos(angle);

h[i][0] = xnew; h[i][1]= ynew;

}

}

void scale(int h[][2],int sx,int sy)

{

int i;

for(i=0;i<9;i++)

{

h[i][0]\*=sx;

h[i][1]\*=sy;

}

}

void translate(int h[][2],int dx, int dy)

{

int i;

for(i=0;i<9;i++)

{

h[i][0]+=dx;

h[i][1]+=dy;

}

}

void reflect(int h[][2], int m, int c)

{

int i;

float angle;

for(i=0;i<9;i++)

h[i][1]-=c;

angle = M\_PI/2-atan(m);

rotate(h,angle);

for(i=0;i<9;i++)

h[i][0]=-h[i][0];

angle = -angle;

rotate(h,angle);

for(i=0;i<9;i++)

h[i][1] +=c ;

}

void ini()

{

int gd = DETECT ,gm;

initgraph(&gd,&gm,"..\\bgi");

}

void dini()

{

getch();

closegraph();

}

void main()

{

int h[9][2],sx,sy,x,y,m,c,choice;

do

{

clrscr();

printf("1.Scaling about the origin.\n");

printf("2.Scaling about an arbitrary point.\n");

printf("3.Reflection about the line y = mx+c.\n");

printf("4.Exit\n");

printf("Enter the choice :");

scanf("%d",&choice);

switch(choice)

{

case 1:printf("Enter the x- and y- scaling factors:");

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

ini();

reset(h);

draw(h);getch();

scale(h, sx,sy);

cleardevice();

draw(h);

dini();

break;

case 2:printf("Enter the x- and y-scaling factors:");

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

printf("Enter the x- and y-coordinates of the point:");

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

ini();

reset(h);

translate(h,x,y);

draw(h);getch();

cleardevice();

translate(h,-x,-y);

draw(h);

getch();

cleardevice();

scale(h,sx,sy);

draw(h);

getch();

translate(h,x,y);

cleardevice();

draw(h);

putpixel(320+x,240-y,WHITE);

dini();

break;

case 3:printf("Enter the values of m and c:");

scanf("%d%d",&m, &c);

ini();

reset(h);

draw(h);getch();

reflect(h,m,c);

cleardevice();

draw(h);

dini();

break;

case 4:exit(0);

}

}while(choice!=4);

}

Pract8a- implement cohen Sutherland clipping

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#include<graphics.h>

#define MAX 20

enum{TOP=0x1,BOTTOM=0x2,RIGHT=0x4,LEFT=0x8};

enum{FALSE,TRUE};

typedef unsigned int outcode;

outcode compute\_outcode(int x,int y,int xmin,int ymin,int xmax,int ymax)

{

outcode oc=0;

if (y>ymax)

oc=TOP;

else if (y<ymin)

oc=BOTTOM;

if (x>xmax)

oc=RIGHT;

else if (x < xmin)

oc=LEFT;

return oc;

}

void cohen\_sutherland(double x1,double y1,double x2,double y2,double xmin,double ymin,double xmax,double ymax)

{

int accept;

int done;

outcode outcodel, outcode2;

accept = FALSE;

done = FALSE;

outcodel=compute\_outcode(x1,y1,xmin,ymin,xmax,ymax);

outcode2=compute\_outcode(x2,y2,xmin,ymin,xmax,ymax);

do

{

if (outcodel ==0 && outcode2 ==0 )

{

accept = TRUE;

done = TRUE;

}

else if (outcodel & outcode2)

{

 done = TRUE;

}

else

{

double x, y;

int outcode\_ex = outcodel? outcodel: outcode2;

if (outcode\_ex & TOP)

{

x = x1+(x2-x1)\*(ymax-y1)/(y2-y1);

 y = ymax;

}

else if (outcode\_ex & BOTTOM)

{

x = x1 + (x2-x1) \* (ymin-y1)/(y2-y1);

 y = ymin;

}

else if (outcode\_ex & RIGHT)

{

 y=y1+(y2-y1)\*(xmax-x1) /( x2-x1);

x=xmax;

}

else

{

y=y1+(y2-y1)\*(xmin-x1)/(x2-x1);

x = xmin;

}

if (outcode\_ex == outcodel)

{

x1=x;

y1 = y;

outcodel = compute\_outcode (x1, y1, xmin, ymin, xmax, ymax);

}

else

{

x2 = x;

y2 = y;

outcode2 = compute\_outcode (x2,y2, xmin,ymin, xmax, ymax);

}

}

}

while (done == FALSE);

if (accept== TRUE)

line (x1, y1, x2, y2);

}

void main()

{

int n;

int i, j;

int ln[MAX][4];

int clip[4];

int gd=DETECT, gm;

clrscr();

printf ("Enter the number of lines to be clipped:\n");

scanf ("%d", &n);

printf ("Enter the x- and y-coordinates of the line- endpoints:\n");

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

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

scanf ("%d", &ln[i][j]);

printf ("Enter the x- and y-coordinates of the left-top and right-");

printf ("bottom corners\nof the clip window:\n");

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

scanf ("%d", &clip[i]);

initgraph(&gd, &gm, "C:\\turboc3\\bgi");

rectangle(clip[0], clip[1], clip[2], clip[3]);

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

line (ln[i][0], ln[i][1], ln[i][2], ln[i][3]);

getch();

cleardevice();

rectangle (clip[0], clip[1], clip[2], clip[3]);

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

{

cohen\_sutherland (ln[i][0], ln[i][1], ln[i][2], ln[i][3], clip[0], clip[1], clip[2], clip[3]);

getch();

}

closegraph();

}

Pract8b- liang barsky line clipping algorithm

#include<stdio.h>

#include<math.h>

#include<conio.h>

#include<dos.h>

#include<graphics.h>

void main()

{

int i,gd=DETECT,gm;

int x1,y1,x2,y2,xmin,xmax,ymin,ymax,xx1,xx2,yy1,yy2,dx,dy;

float t1,t2,p[4],q[4],temp;

clrscr();

printf("Enter line co ordinates x1,y1:");

scanf("%d%d",&x1,&y1);

printf("Enter line co ordinates x2,y2:");

scanf("%d%d",&x2,&y2);

xmin=100;

ymin=100;

xmax=250;

ymax=250;

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

rectangle(xmin,ymin,xmax,ymax);

dx=x2-x1;

dy=y2-y1;

p[0]=-dx;

p[1]=dx;

p[2]=-dy;

p[3]=dy;

q[0]=x1-xmin;

q[1]=xmax-x1;

q[2]=y1-ymin;

q[3]=ymax-y1;

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

{

if(p[i]==0)

{

printf("Line is parallel to one of the clipping boundary");

if(q[i]>=0)

{

if(i<2)

{

if(y1<ymin)

{

y1=ymin;

}

if(y2>ymax)

{

y2=ymax;

}

line(x1,y1,x2,y2);

}

if(i>1)

{

if(x1<xmin)

{

x1=xmin;

}

if(x2>xmax)

{

x2=xmax;

}

line(x1,y1,x2,y2);

}

}

}

}

t1=0;

t2=1;

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

{

temp=q[i]/p[i];

if(p[i]<0)

{

if(t1<=temp)

t1=temp;

}

else

{

if(t2>temp)

t2=temp;

}

}

if(t1<t2)

{

xx1=x1+t1\*p[1];

xx2=x1+t2\*p[1];

yy1=y1+t1\*p[3];

yy2=y1+t2\*p[3];

line(xx1,yy1,xx2,yy2);

}

delay(5000);

closegraph();

}

Pract9a- to fill a circle using floodfill algorithm

#include<stdio.h>

#include<dos.h>

#include<conio.h>

#include<graphics.h>

void floodFill(int x,int y,int oldcolor,int newcolor)

{

if(getpixel(x,y)==oldcolor)

{

putpixel(x,y,newcolor);

floodFill(x+1,y,oldcolor,newcolor);

floodFill(x,y+1,oldcolor,newcolor);

floodFill(x-1,y,oldcolor,newcolor);

floodFill(x,y-1,oldcolor,newcolor);

}

}

//getpixel(x,y)given the color of specified pixel

void main()

{

int gm,gd=DETECT,radius;

int x,y;

printf("Enter x and y positions for circle\n");

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

printf("Enter radius of circle\n");

scanf("%d",&radius);

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

circle(x,y,radius);

floodFill(x,y,6,15);

delay(5000);

closegraph();

}

Pract9b- to fill a circle using boundaryfill algorithm.

#include<stdio.h>

#include<dos.h>

#include<conio.h>

#include<graphics.h>

void boundaryfill(int x,int y,int f\_color,int b\_color)

{

if(getpixel(x,y)!=b\_color && getpixel(x,y)!=f\_color)

{

putpixel(x,y,f\_color);

boundaryfill(x+1,y,f\_color,b\_color);

boundaryfill(x,y+1,f\_color,b\_color);

boundaryfill(x-1,y,f\_color,b\_color);

boundaryfill(x,y-1,f\_color,b\_color);

}

}

void main()

{

int gm,gd=DETECT,radius;

int x,y;

printf("Enter x and y positions for circle\n");

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

printf("Enter radius of circle\n");

scanf("%d",&radius);

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

circle(x,y,radius);

boundaryfill(x,y,4,15);

delay(5000);

closegraph();

}

Pract10a- simple text screen saver using graphic function.

#include<stdlib.h>

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int gd=DETECT,gm,x=600,i;

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

for(x=0;x<250;x++)

{

x%=250;

setcolor(random(16));

circle(random(635),random(70),50);

circle(random(635),random(70),50);

circle(random(635),random(70),50);

circle(random(635),random(70),50);

circle(random(635),random(70),50);

clearviewport();

settextstyle(1,0,5);

setcolor(RED);

outtextxy(50,415-2\*x,"\*World\*");

setcolor(GREEN);

outtextxy(200,415-2\*x,"\*of\*");

setcolor(YELLOW);

settextstyle(3,0,5);

outtextxy(350,415-2\*x,"\*graphics\*");

}

getch();

}

Pract10b- smiling face animation .

#include<stdio.h>

#include<graphics.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm;

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

circle(200,200,30);

circle(190,190,5);

arc(190,190,50,130,10);

circle(210,190,5);

arc(210,190,50,130,10);

arc(200,210,180,360,10);

line(187,210,193,210);

line(207,210,213,210);

line(198,195,195,200);

line(202,195,205,200);

line(195,200,200,205);

line(205,200,200,205);

getch();

closegraph();

}

Pract10c- moving car on the screen.

#include<conio.h>

#include<stdio.h>

#include<graphics.h>

#include<dos.h>

void main()

{

int gd=DETECT,gm;

int i,maxx,midy;

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

maxx=getmaxx();

midy=getmaxy()/2;

for(i=0;i<maxx-150;i=i+5)

{

cleardevice();

setcolor(WHITE);

line(0,midy+37,maxx,midy+37);

setcolor(YELLOW);

setfillstyle(SOLID\_FILL,RED);

line(i,midy+23,i,midy);

line(i,midy,40+i,midy-20);

line(40+i,midy-20,80+i,midy-20);

line(80+i,midy-20,100+i,midy);

line(100+i,midy,120+i,midy);

line(120+i,midy,120+i,midy+23);

line(0+i,midy+23,18+i,midy+23);

arc(30+i,midy+23,0,180,12);

line(42+i,midy+23,78+i,midy+23);

arc(90+i,midy+23,0,180,12);

line(102+i,midy+23,120+i,midy+23);

line(28+i,midy,43+i,midy-15);

line(43+i,midy-15,57+i,midy-15);

line(57+i,midy-15,57+i,midy);

line(57+i,midy,28+i,midy);

line(62+i,midy-15,77+i,midy-15);

line(77+i,midy-15,92+i,midy);

line(92+i,midy,62+i,midy);

line(62+i,midy,62+i,midy-15);

floodfill(5+i,midy+22,YELLOW);

setcolor(BLUE);

setfillstyle(SOLID\_FILL,DARKGRAY);

circle(30+i,midy+25,9);

circle(90+i,midy+25,9);

floodfill(30+i,midy+25,BLUE);

floodfill(90+i,midy+25,BLUE);

delay(5);

}

getch();

closegraph();

}