#include<iostream>

#include<graphics.h>

#include<cstdlib>

#include<dos.h>

#include<cmath>

using namespace std;

int main()

{

initwindow(800,500);

int x0,y0;

int gdriver = DETECT,gmode,errorcode;

int xmax,ymax;

errorcode=graphresult();//graphresult() function returns the error code for any failed graphics operation and that error code is accepted by the grapherrormsg() function. If not any graphic operations fails then it returns grOk.

if(errorcode!=0)

{

cout<<"Graphics error:"<<grapherrormsg(errorcode);

cout<<"Press any key to halt";

exit(1);

}

int i,j;

setbkcolor(BLUE);

setcolor(RED);

rectangle(0,0,getmaxx(),getmaxy());

outtextxy(250,240,"::::PRESS ANY KEY TO CONTINUE:::::");

while(!kbhit());//kbhit() is present in conio.h and used to determine if a key has been pressed or not. To use kbhit function in your program you should include the header file “conio.h”. If a key has been pressed then it returns a non zero value otherwise returns zero.

for(i=50,j=0;i<=250,j<=250;i+=5,j+=5)

{

delay(120);

cleardevice();

if(i<=150)

{

setcolor(YELLOW);

setfillstyle(1,YELLOW);// setfillstyle() function contains two arguments pattern and color. Various fill patterns are already enumerated in graphics.h header file

fillellipse(i,300-j,20,20);// Fills the interior of an ellipse defined by a bounding rectangle specified by a pair of coordinates, a width, and a height.

}

else

{

setcolor(GREEN^RED);

setfillstyle(1,GREEN^RED);

fillellipse(i,300-j,20,20);

}

}

delay(1000);

cleardevice();

setcolor(RED);

setfillstyle(1,RED);

fillellipse(300,50,20,20);

delay(150);

int k,l;

for(k=305,l=55;k<=550,l<=300;k+=5,l+=5)

{

delay(120);

cleardevice();

if(k<=450)

{

setcolor(GREEN^RED);

setfillstyle(1,GREEN^RED);

fillellipse(k,l,20,20);// he header file graphics.h contains **fillellipse()** function which draws and fills an ellipse with center at (x, y) and (x\_radius, y\_radius) as x and y radius of ellipse.

}

else

{

setcolor(YELLOW);

setfillstyle(1,YELLOW);

fillellipse(k,l,20,20);

}

}

return 0;

}