Program 1:

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int x1,y1,x2,y2,dx,dy,length,i;

float x,y,xinc,yinc;

int gd=DETECT,gm;

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

printf("Enter the starting coordinates");

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

printf("Enter the ending coordinates");

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

dx=x2-x1;

dy=y2-y1;

if(abs(dx)>abs(dy))

length=abs(dx);

else

length=abs(dy);

xinc=dx/(float)length;

yinc=dy/(float)length;

x=x1;

y=y1;

putpixel(x,y,10);

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

{

putpixel(x,y,10);

x=x+xinc;

y=y+yinc;

delay(10);

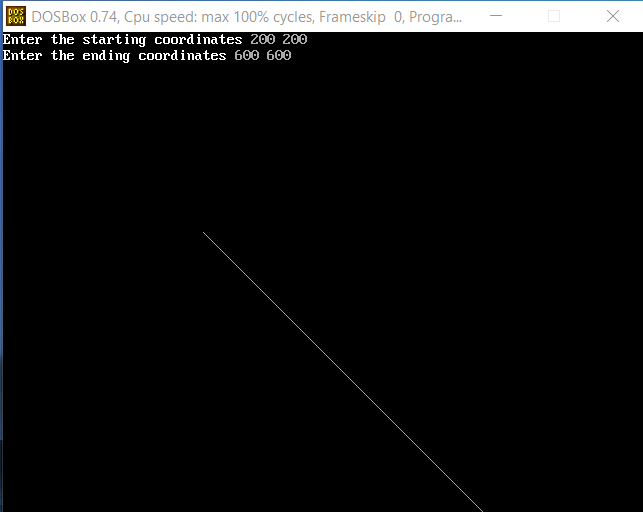
}

getch();

closegraph();

}

Output:



Program 2:

#include<stdio.h>

#include<graphics.h>

void drawline(int x0, int y0, int x1, int y1)

{

int dx, dy, p, x, y;

dx=x1-x0;

dy=y1-y0;

x=x0;

y=y0;

p=2\*dy-dx;

while(x<x1)

{

if(p>=0)

{

putpixel(x,y,7);

y=y+1;

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

}

else

{

putpixel(x,y,7);

p=p+2\*dy;

}

x=x+1;

}

}

int main()

{

int gdriver=DETECT, gmode, error, x0, y0, x1, y1;

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

printf("Enter co-ordinates of first point: ");

scanf("%d%d", &x0, &y0);

printf("Enter co-ordinates of second point: ");

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

drawline(x0, y0, x1, y1);

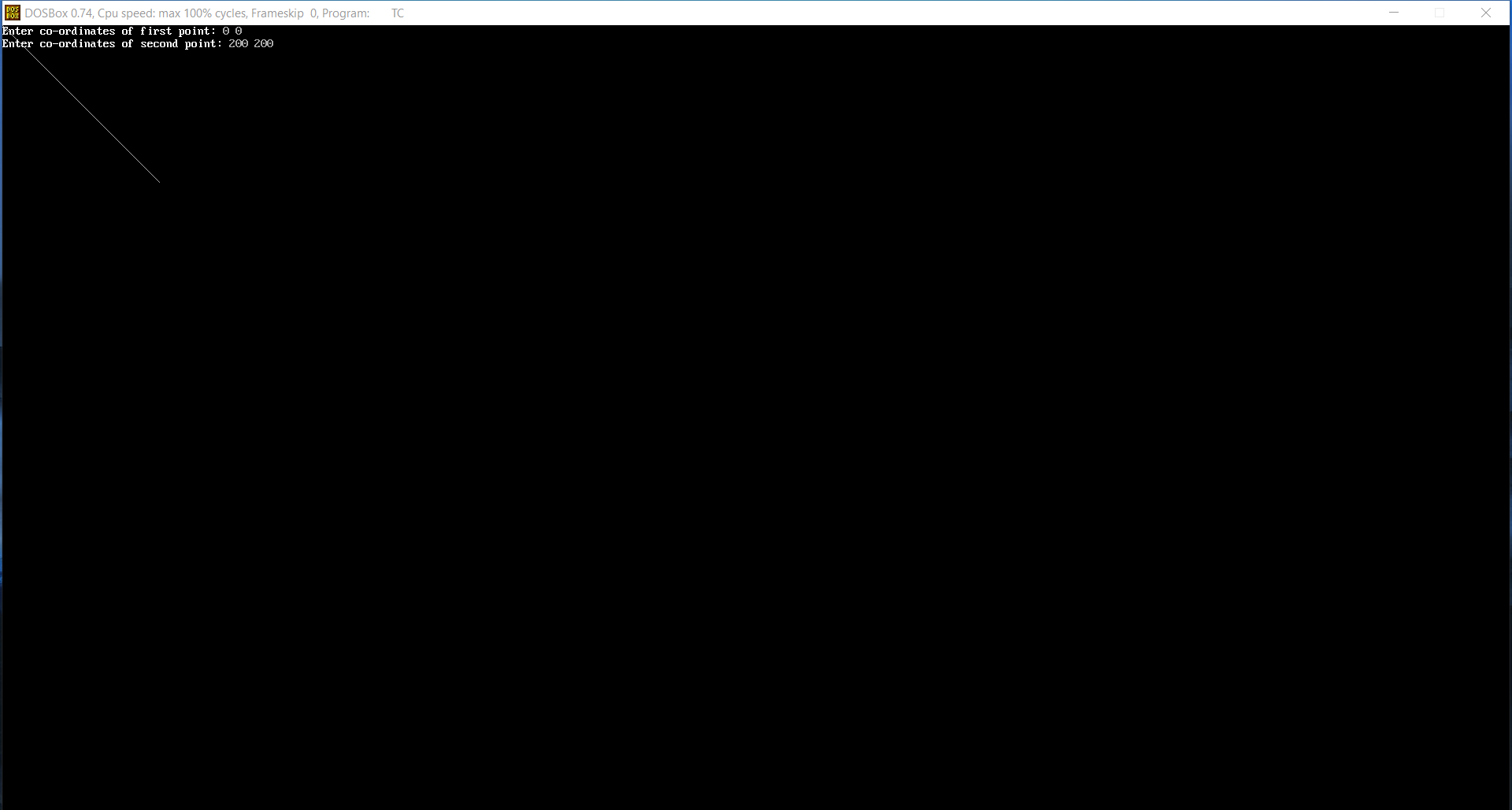
getch();

closegraph();

return 0;

}

Output:



Program 3:

#include<graphics.h>

#include<stdio.h>

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

int main()

{

int gd,gm,xc,yc,r,x,y,p;

detectgraph(&gd,&gm);

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

printf("Enter center of circle :");

scanf("%d%d",&xc,&yc);

printf("Enter radius of circle :");

scanf("%d",&r);

x=0;

y=r;

p=1-r;

pixel(xc,yc,x,y);

while(x<y)

{

if(p<0)

{

x++;

p=p+2\*x+1;

}

else

{

x++;

y--;

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

}

pixel(xc,yc,x,y);

}

getch();

closegraph();

return 0;

}

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

{

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

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

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

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

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

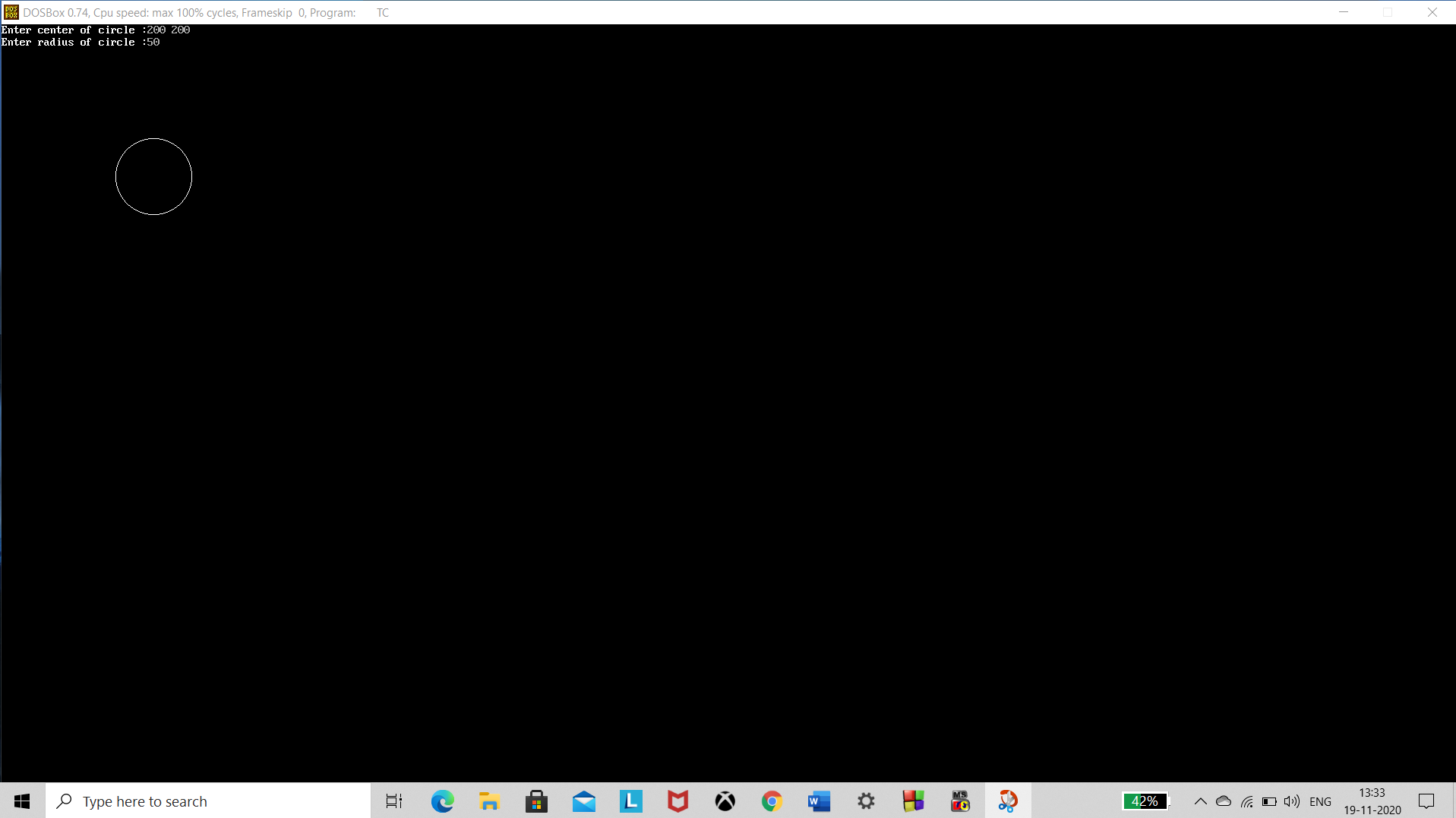
putpixel(xc+y,yc-x,WHITE);

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

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

}

Output:



Program 4:

#include<stdio.h>

#include<graphics.h>

void main()

{

int gd=DETECT,gm;

float p,x,y,xc,yc,a,b;

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

cleardevice();

printf("Enter xc, yc:\n");

scanf("%f%f",&xc,&yc);

printf("Enter a, b:\n");

scanf("%f%f",&a,&b);

x=0;

y=b;

//Region 1

p=(b\*b)-(a\*a\*b)+(0.25\*a\*a);

do

{

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

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

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

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

if(p<0)

{

x=x+1;

p=p+2\*b\*b\*x+b\*b;

}

else

{

x=x+1;

y=y-1;

p=p+2\*b\*b\*x-2\*a\*a\*y+b\*b;

}

}while(2\*b\*b\*x<2\*a\*a\*y);

//Region 2

p=(b\*b\*(x+0.5)\*(x+0.5))+((y-1)\*(y-1)\*a\*a-a\*a\*b\*b);

do

{

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

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

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

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

if(p>0)

{

y=y-1;

p=p-2\*a\*a\*y+a\*a;

}

else

{

x=x+1;

y=y-1;

p=p-2\*a\*a\*y+2\*b\*b\*x+a\*a;

}

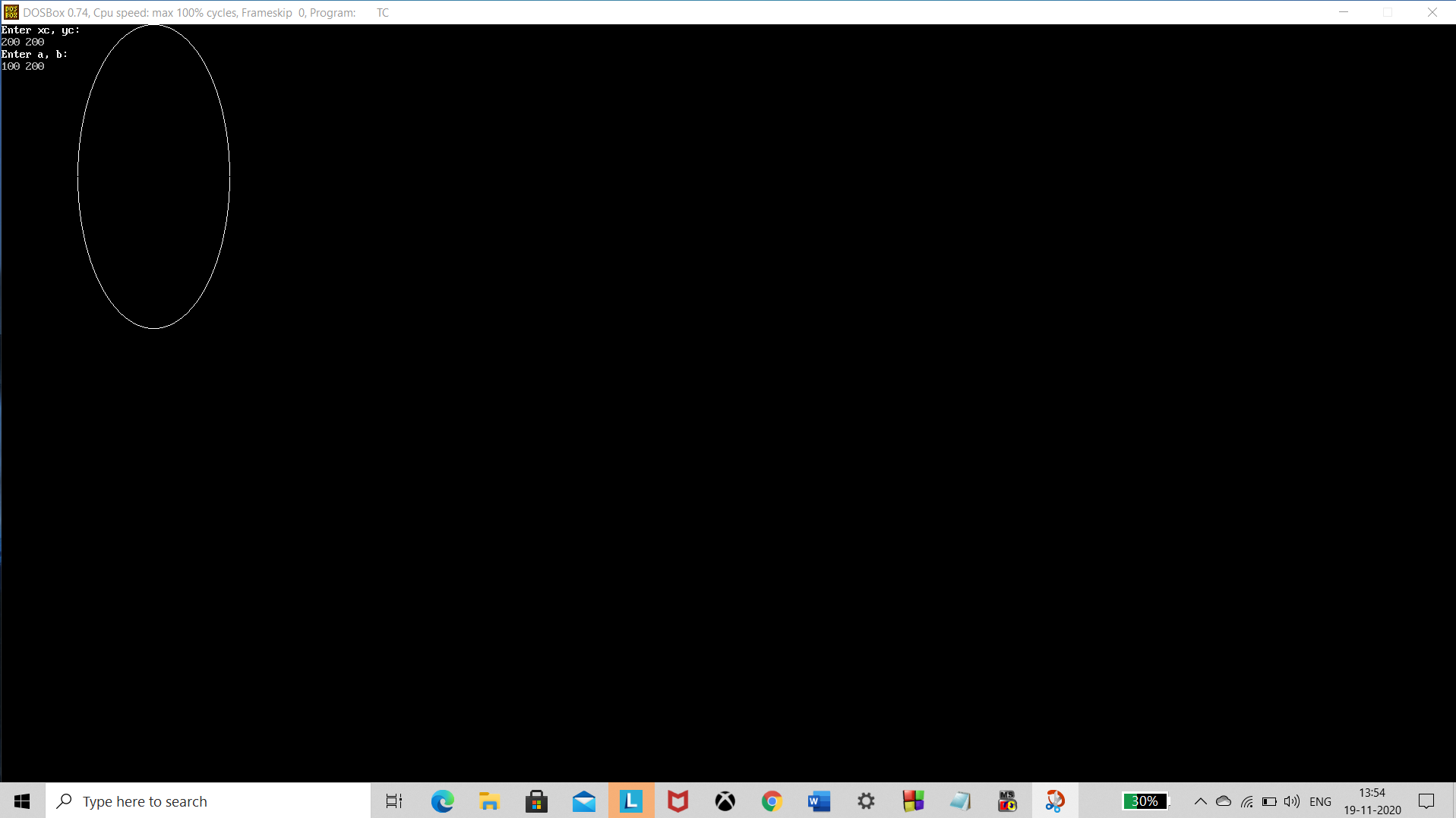
}while(y!=0);

getch();

closegraph();

}

Output:



Program 7:

#include <graphics.h>

#include <stdlib.h>

#include <stdio.h>

#include <conio.h>

#include<math.h>

void main()

{

int gm;

int gd=DETECT;

int x1,x2,x3,y1,y2,y3;

int nx1,nx2,nx3,ny1,ny2,ny3;

float sx,sy,xt,yt,r;

float t;

int c;

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

printf("\t Program for basic transactions");

printf("\n\t Enter the points of triangle");

setcolor(1);

scanf("%d%d%d%d%d%d",&x1,&y1,&x2,&y2,&x3,&y3);

line(x1,y1,x2,y2);

line(x2,y2,x3,y3);

line(x3,y3,x1,y1);

getch();

do

{

printf(" 1.Translation 2.Rotation 3.Scaling 4.Reflection 5.exit ");

printf("Enter your choice:");

scanf("%d",&c);

switch(c)

{

case 1: cleardevice();

printf("Enter the translation factor");

scanf("%f%f",&xt,&yt);

nx1=x1+xt;

ny1=y1+yt;

nx2=x2+xt;

ny2=y2+yt;

nx3=x3+xt;

ny3=y3+yt;

line(nx1,ny1,nx2,ny2);

line(nx2,ny2,nx3,ny3);

line(nx3,ny3,nx1,ny1);

getch();

cleardevice();

break;

case 2: cleardevice();

printf(" Enter the angle of rotation");

scanf("%f",&r);

t=3.14\*r/180;

nx1=abs(x1\*cos(t)-y1\*sin(t));

ny1=abs(x1\*sin(t)+y1\*cos(t));

nx2=abs(x2\*cos(t)-y2\*sin(t));

ny2=abs(x2\*sin(t)+y2\*cos(t));

nx3=abs(x3\*cos(t)-y3\*sin(t));

ny3=abs(x3\*sin(t)+y3\*cos(t));

line(nx1,ny1,nx2,ny2);

line(nx2,ny2,nx3,ny3);

line(nx3,ny3,nx1,ny1);

getch();

cleardevice();

break;

case 3: cleardevice();

printf(" Enter the scaling factor");

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

nx1=abs(x1\*sx);

ny1=abs(y1\*sy);

nx2=abs(x2\*sx);

ny2=abs(y2\*sy);

nx3=abs(x3\*sx);

ny3=abs(y3\*sy);

line(nx1,ny1,nx2,ny2);

line(nx2,ny2,nx3,ny3);

line(nx3,ny3,nx1,ny1);

getch();

cleardevice();

break;

case 4: cleardevice();

nx1 =abs(x1-140);

ny1=abs(y1+140);

nx2=abs(x2-140);

ny2=abs(y2+140);

nx3=abs(x3-140);

ny3=abs(y3+140);

line(nx1,ny1,nx2,ny2);

line(nx2,ny2,nx3,ny3);

line(nx3,ny3,nx1,ny1);

getch();

cleardevice();

break;

case 5:

break;

default:

printf("Enter the correct choice");

break;

}

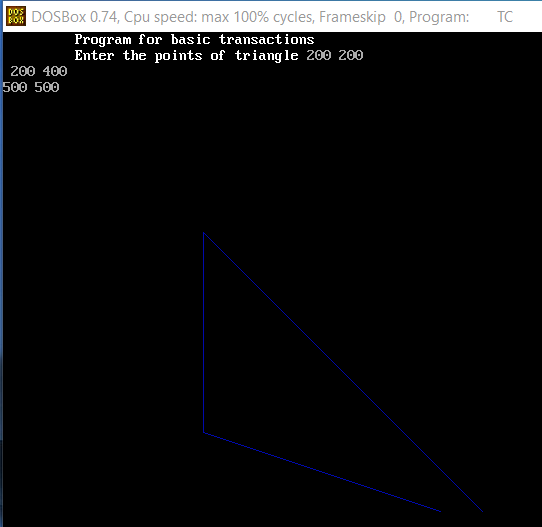
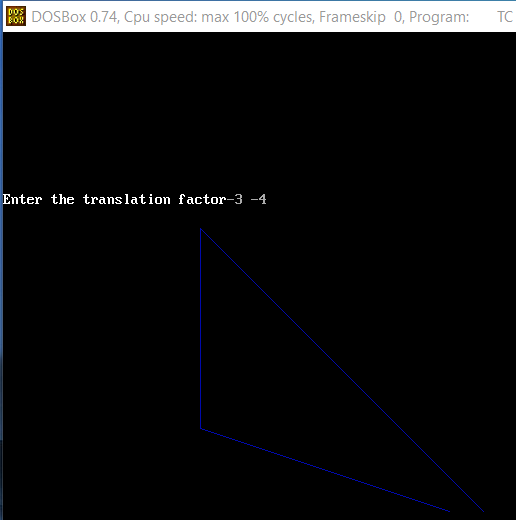
}while(c!=5);

closegraph();

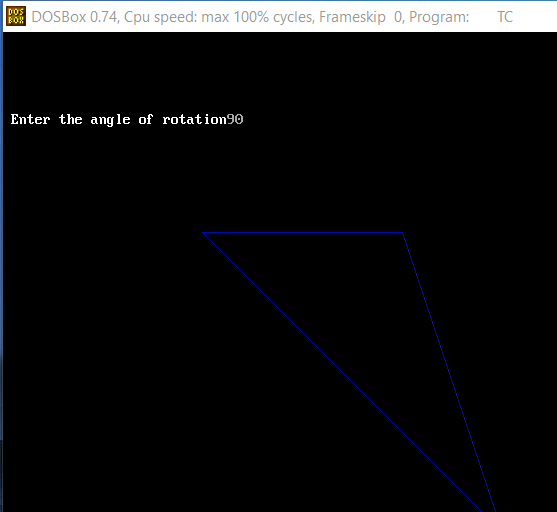
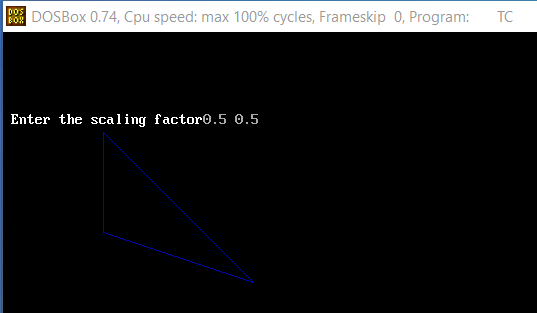
}

Output:

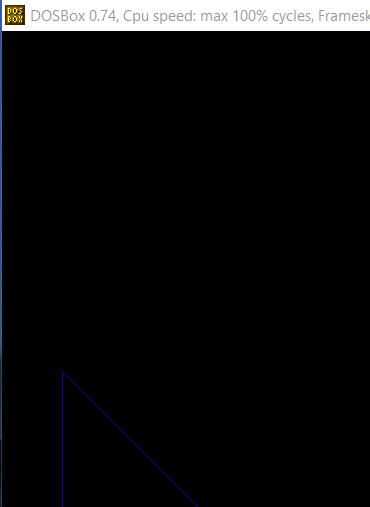
Initial: Translation:

Rotation: Scaling:

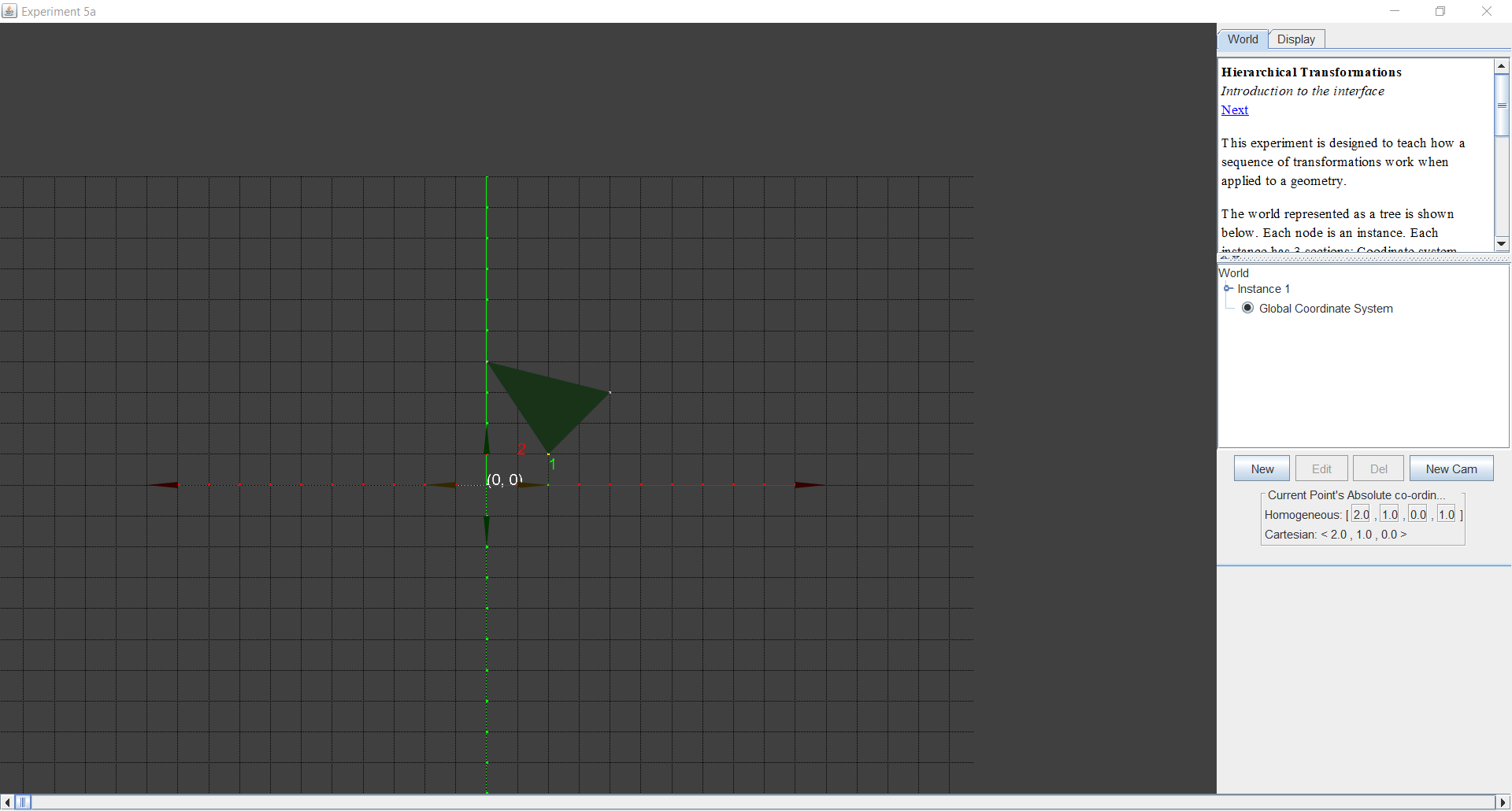
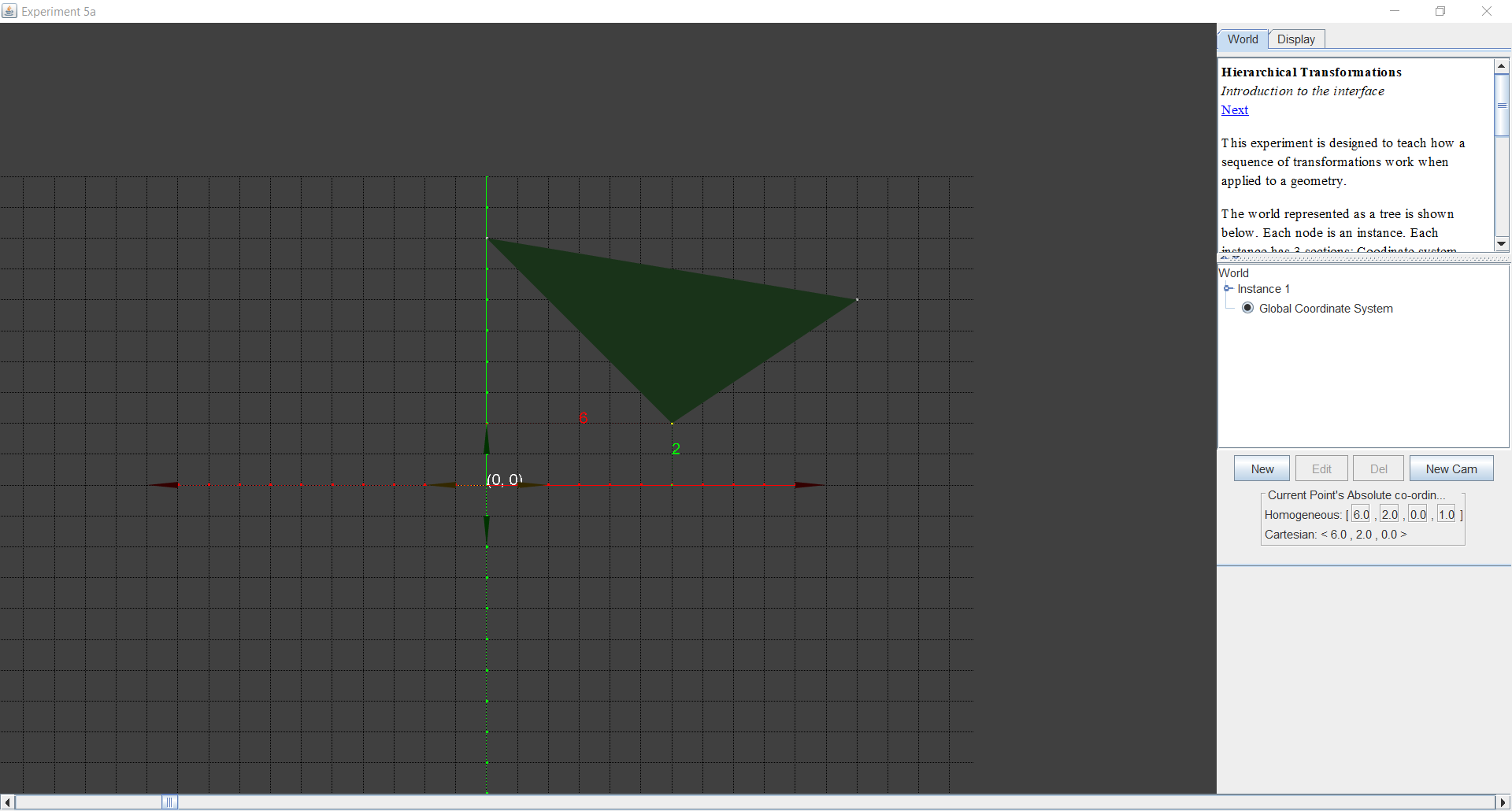
 

Reflection:

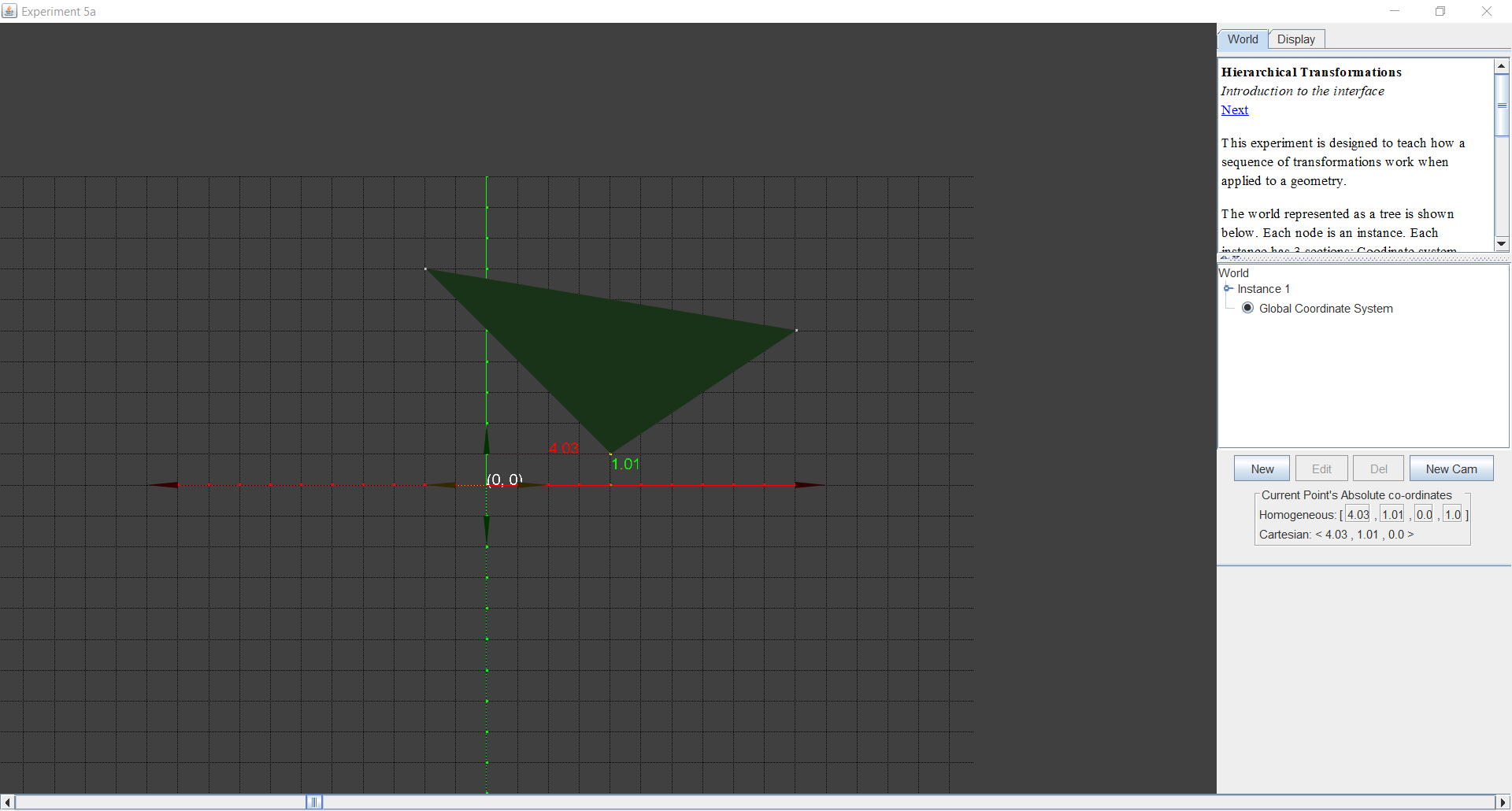
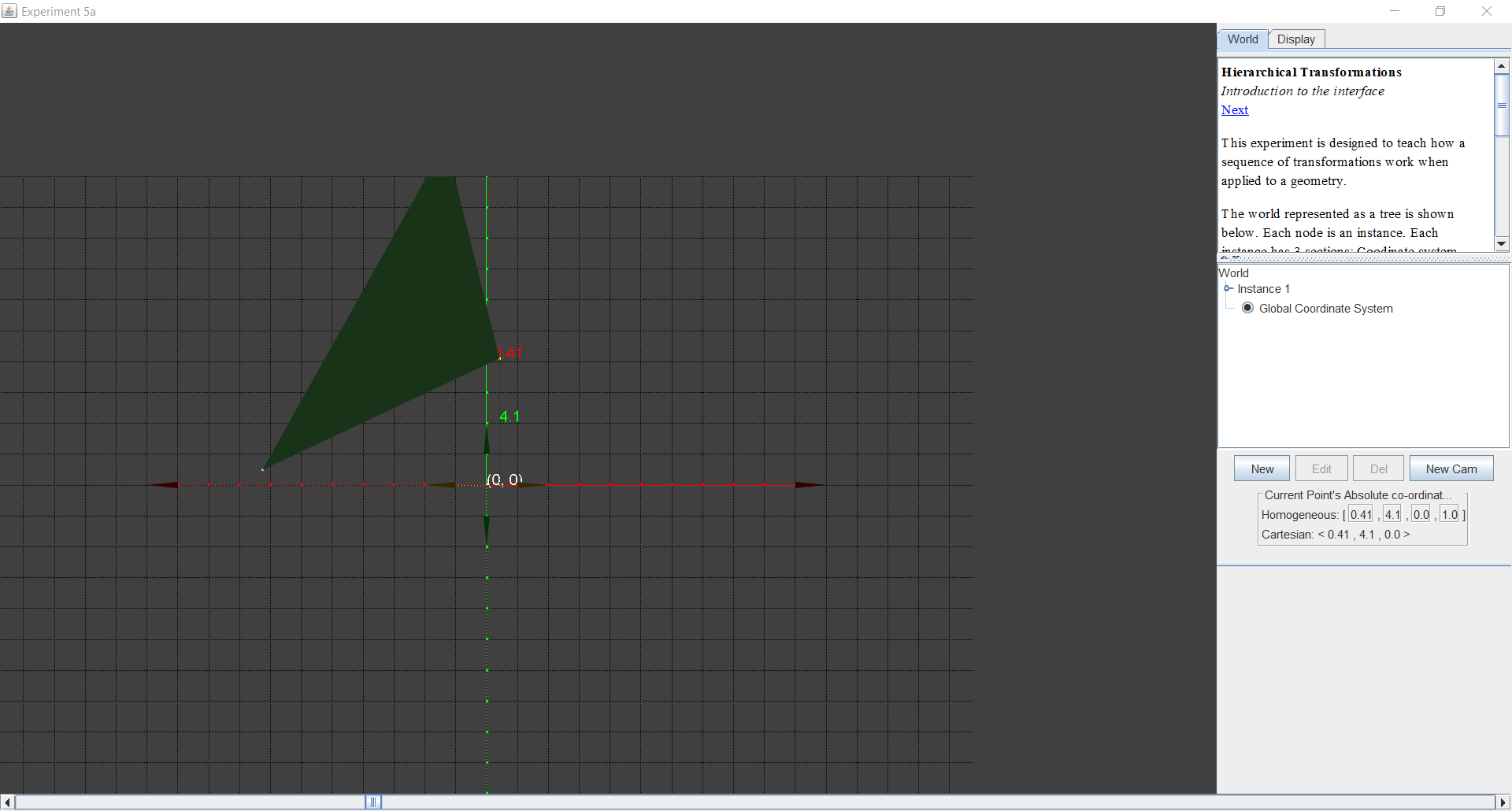


Virtual Lab:

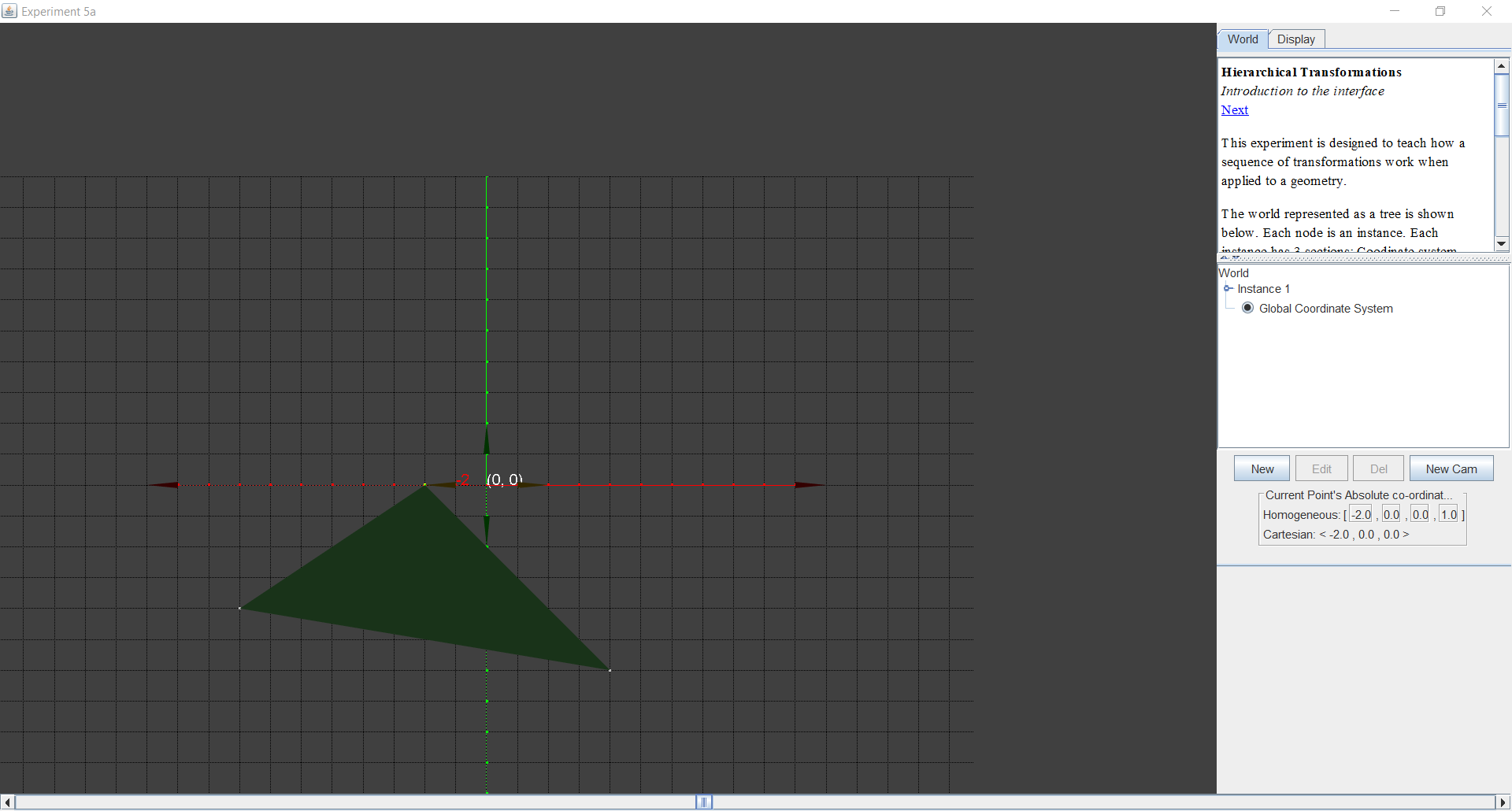
Initial: Scaling:

Translate: Rotate:

Reflection:



Program 5:

Boundary Fill:

#include<graphics.h>

#include<dos.h>

#include<conio.h>

void boundaryFill8(int x, int y, int fill\_color,int boundary\_color)

{

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

getpixel(x, y) != fill\_color)

{

putpixel(x, y, fill\_color);

boundaryFill8(x + 1, y, fill\_color, boundary\_color);

boundaryFill8(x, y + 1, fill\_color, boundary\_color);

boundaryFill8(x - 1, y, fill\_color, boundary\_color);

boundaryFill8(x, y - 1, fill\_color, boundary\_color);

boundaryFill8(x - 1, y - 1, fill\_color, boundary\_color);

boundaryFill8(x - 1, y + 1, fill\_color, boundary\_color);

boundaryFill8(x + 1, y - 1, fill\_color, boundary\_color);

boundaryFill8(x + 1, y + 1, fill\_color, boundary\_color);

}

}

void main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "c:\\Turboc3\\bgi");

rectangle(50, 50, 100, 100);

boundaryFill8(55, 55, 4, 15);

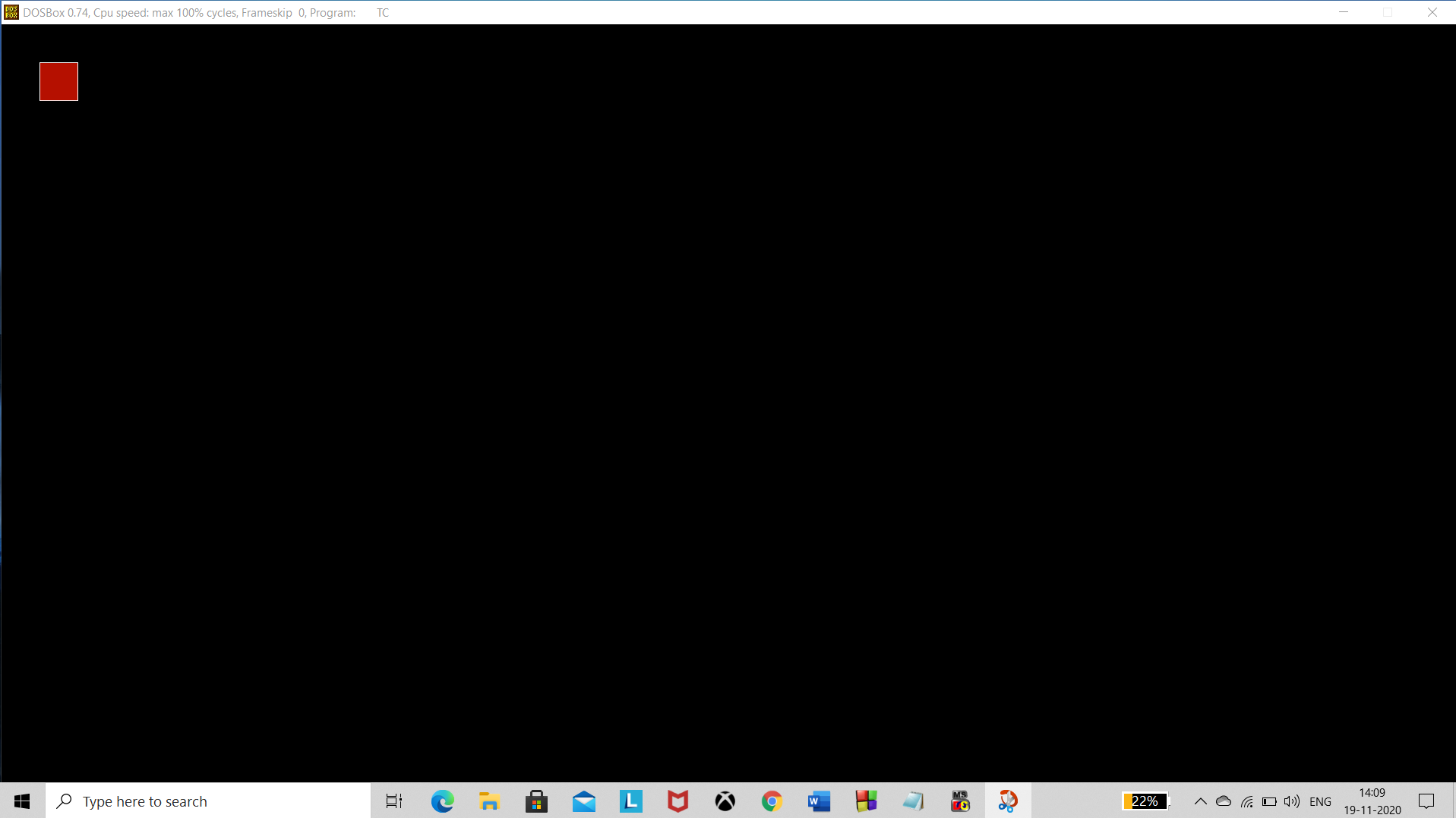
delay(10000);

getch();

closegraph();

}

Output:



Flood Fill:

#include <graphics.h>

#include <stdio.h>

void flood(int x, int y, int new\_col, int old\_col)

{

if (getpixel(x, y) == old\_col) {

putpixel(x, y, new\_col);

flood(x + 1, y, new\_col, old\_col);

flood(x - 1, y, new\_col, old\_col);

flood(x, y + 1, new\_col, old\_col);

flood(x, y - 1, new\_col, old\_col);

}

}

int main()

{

int gm, gd = DETECT;

int top,left,bottom,right;

int x=51,y=51;

int newcolor=12,oldcolor=0;

initgraph(&gd, &gm, "c://turboc3//bgi");

top = left = 50;

bottom = right = 300;

rectangle(left, top, right, bottom);

flood(x, y, newcolor, oldcolor);

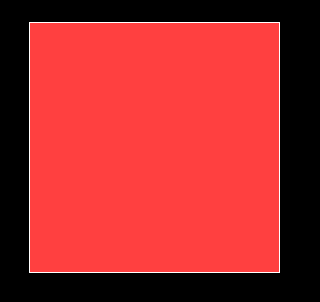
getch();

closegraph();

return 0;

}

Output:



Program 6:

#include <stdio.h>

#include <conio.h>

#include <graphics.h>

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\tEnter the no. of edges of polygon : ");

scanf("%d",&n);

printf("\n\n\tEnter the cordinates 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");

/\*- draw polygon -\*/

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(35);

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

{

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

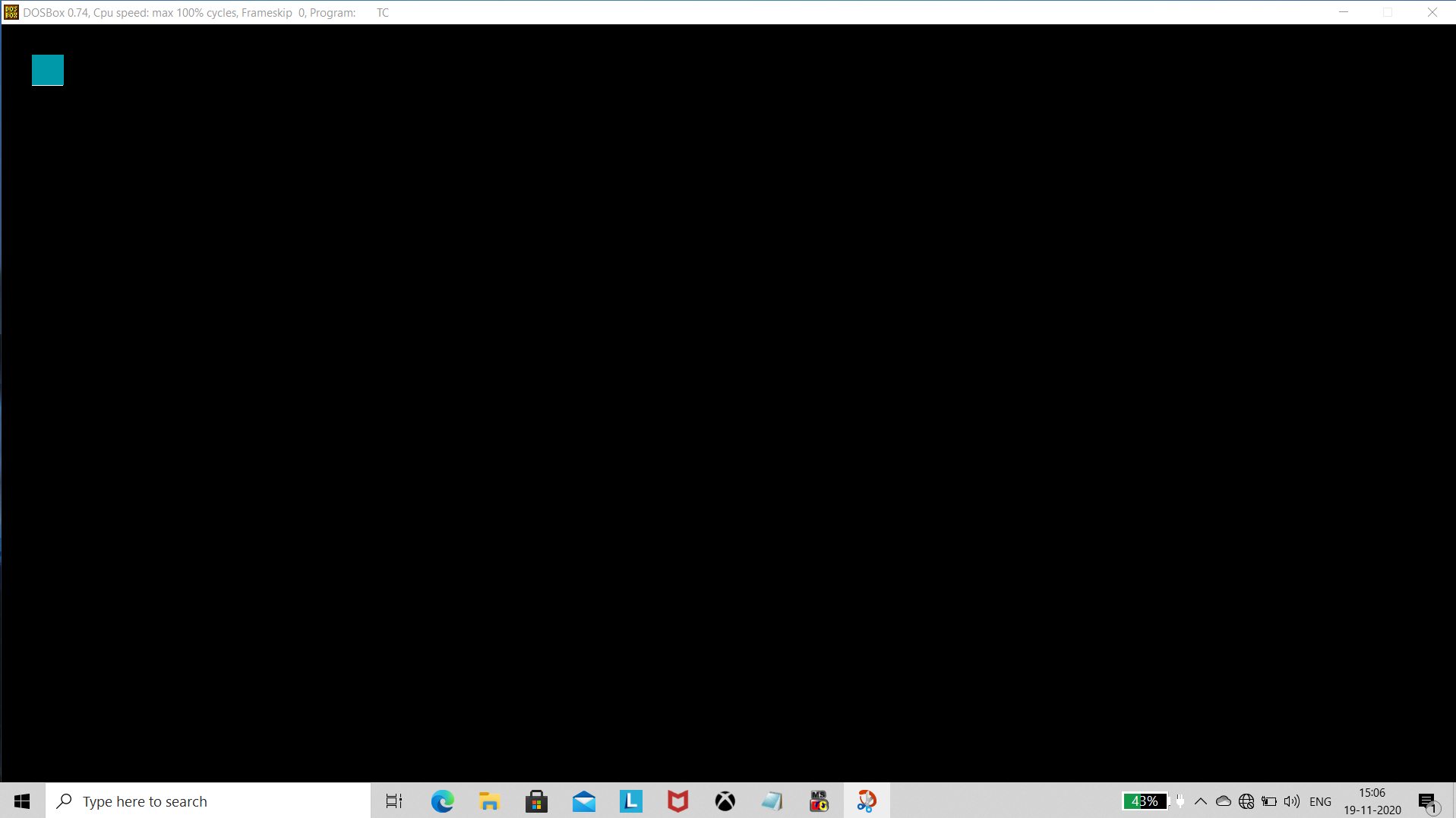
getch();

}

}

}

Output:



Program 8:

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#include<graphics.h>

#include<dos.h>

typedef struct coordinate

{

int x,y;

char code[4];

}PT;

void drawwindow();

void drawline(PT p1,PT p2);

PT setcode(PT p);

int visibility(PT p1,PT p2);

PT resetendpt(PT p1,PT p2);

void main()

{

int gd=DETECT,v,gm;

PT p1,p2,p3,p4,ptemp;

printf("\nEnter x1 and y1\n");

scanf("%d %d",&p1.x,&p1.y);

printf("\nEnter x2 and y2\n");

scanf("%d %d",&p2.x,&p2.y);

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

drawwindow();

delay(500);

drawline(p1,p2);

getch();

cleardevice();

getch();

p1=setcode(p1);

p2=setcode(p2);

v=visibility(p1,p2);

getch();

switch(v)

{

case 0: drawwindow();

delay(500);

drawline(p1,p2);

break;

case 1: drawwindow();

delay(500);

break;

case 2: p3=resetendpt(p1,p2);

p4=resetendpt(p2,p1);

drawwindow();

delay(500);

drawline(p3,p4);

break;

}

getch();

closegraph();

}

void drawwindow()

{

line(150,100,450,100);

line(450,100,450,350);

line(450,350,150,350);

line(150,350,150,100);

}

void drawline(PT p1,PT p2)

{

line(p1.x,p1.y,p2.x,p2.y);

}

PT setcode(PT p) //for setting the 4 bit code

{

PT ptemp;

if(p.y<100)

ptemp.code[0]='1'; //Top

else

ptemp.code[0]='0';

if(p.y>350)

ptemp.code[1]='1'; //Bottom

else

ptemp.code[1]='0';

if(p.x>450)

ptemp.code[2]='1'; //Right

else

ptemp.code[2]='0';

if(p.x<150)

ptemp.code[3]='1'; //Left

else

ptemp.code[3]='0';

ptemp.x=p.x;

ptemp.y=p.y;

return(ptemp);

}

int visibility(PT p1,PT p2)

{

int i,flag=0;

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

{

if((p1.code[i]!='0') || (p2.code[i]!='0'))

flag=1;

}

if(flag==0)

return(0);

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

{

if((p1.code[i]==p2.code[i]) && (p1.code[i]=='1'))

flag='0';

}

if(flag==0)

return(1);

return(2);

}

PT resetendpt(PT p1,PT p2)

{

PT temp;

int x,y,i;

float m,k;

if(p1.code[3]=='1')

x=150;

if(p1.code[2]=='1')

x=450;

if((p1.code[3]=='1') || (p1.code[2]=='1'))

{

m=(float)(p2.y-p1.y)/(p2.x-p1.x);

k=(p1.y+(m\*(x-p1.x)));

temp.y=k;

temp.x=x;

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

temp.code[i]=p1.code[i];

if(temp.y<=350 && temp.y>=100)

return (temp);

}

if(p1.code[0]=='1')

y=100;

if(p1.code[1]=='1')

y=350;

if((p1.code[0]=='1') || (p1.code[1]=='1'))

{

m=(float)(p2.y-p1.y)/(p2.x-p1.x);

k=(float)p1.x+(float)(y-p1.y)/m;

temp.x=k;

temp.y=y;

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

temp.code[i]=p1.code[i];

return(temp);

}

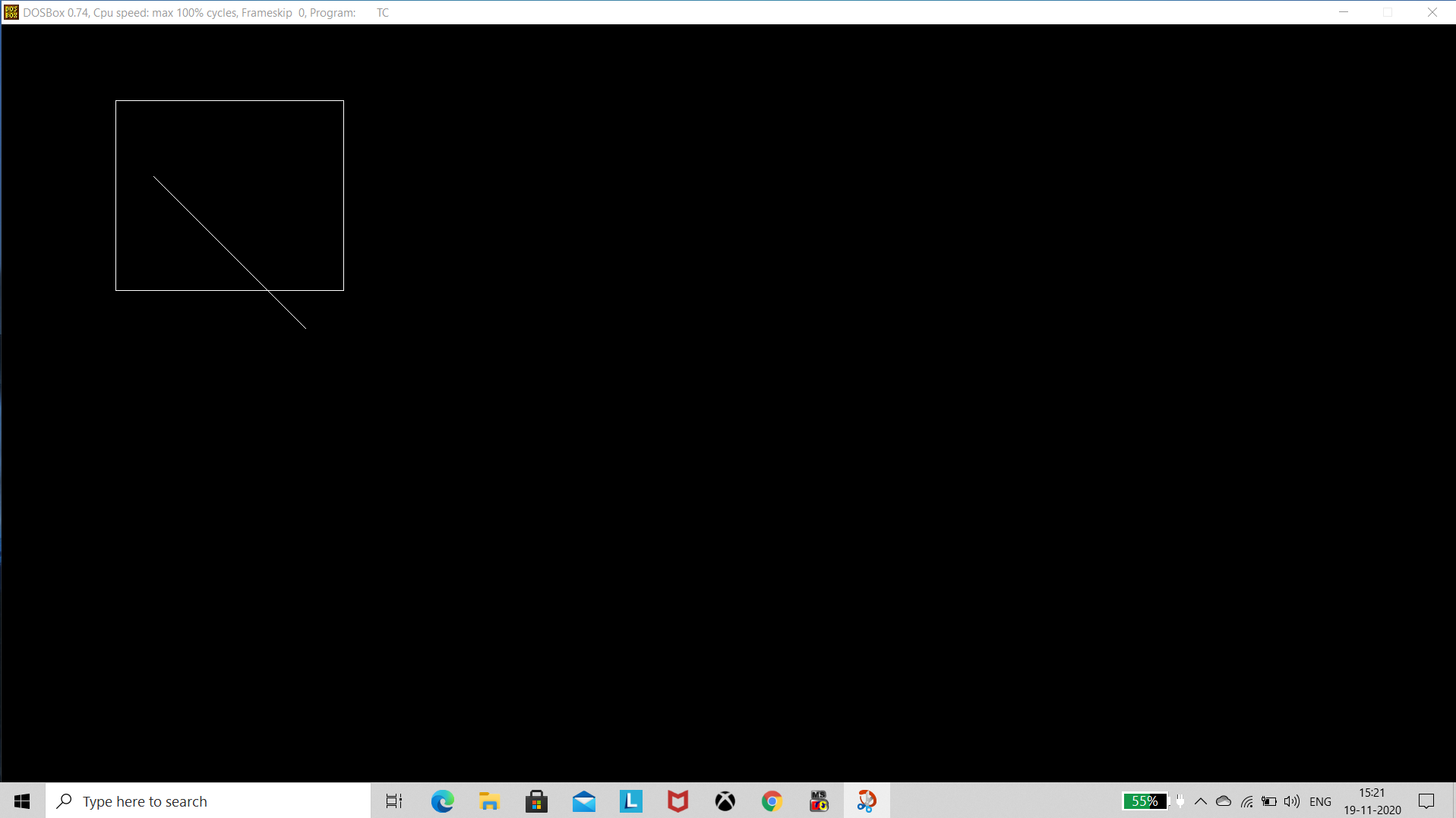
else

return(p1);

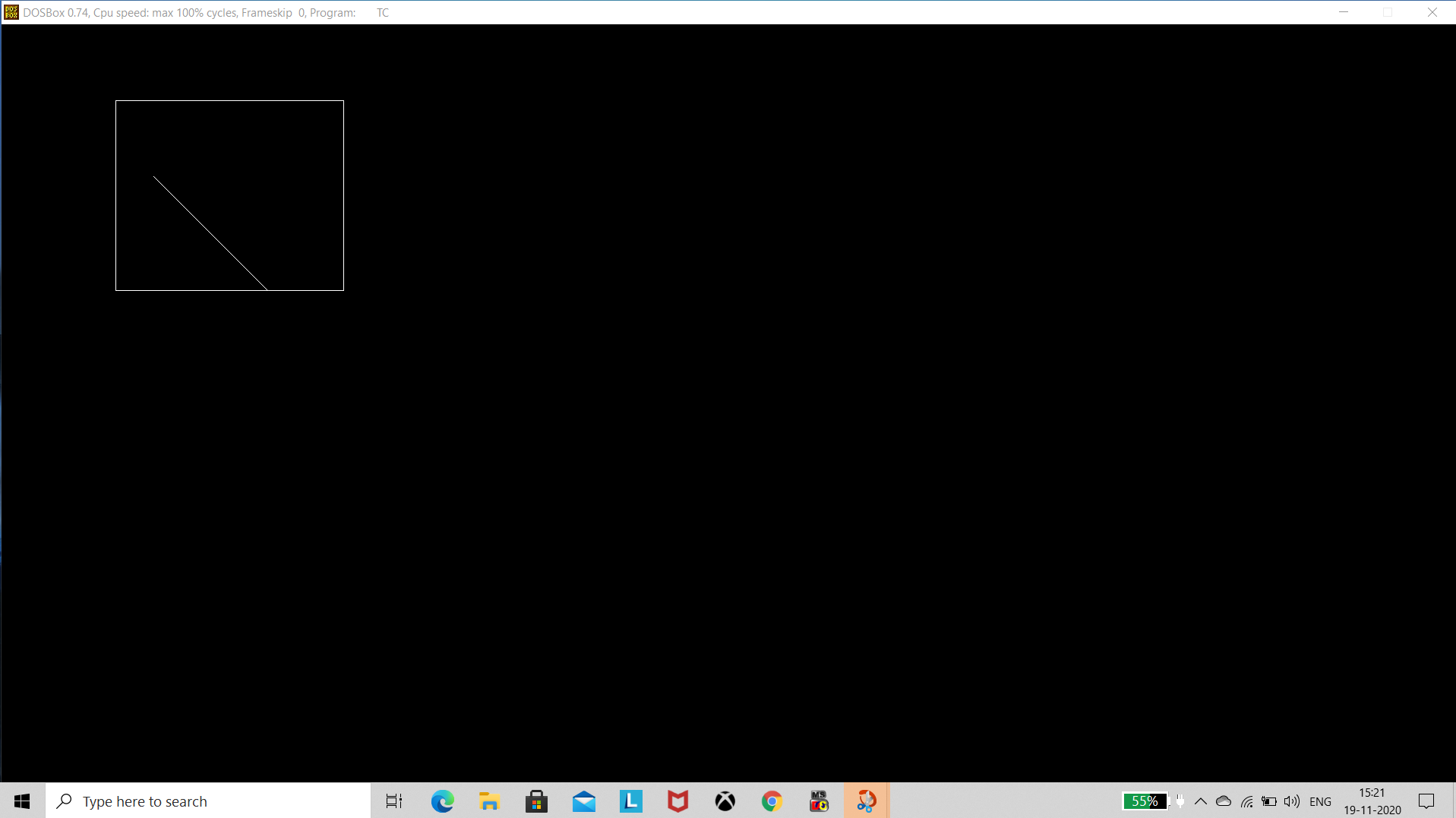
}

Output :

Before Clipping:



After Clipping:



Program 9:

Parallel Projection:

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int ch,x[10],y[10],z[10],a,b,c,j,xr[10],yr[10],zr[10];

int gd=DETECT,gm;

printf("\n\n parallel projection");

printf("\n\n to enter coordinates\n\n") ;

printf("\n else any other value would lead to default value \t:");

scanf("%d",&ch);

if(ch==1)

{

printf("\n follow these steps");

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

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

}

else

{

x[0]=y[0]=z[0]=0;

x[1]=80;y[1]=z[1]=0;

x[2]=y[2]=80;z[2]=0;

x[3]=0;y[3]=80;z[3]=0;

x[4]=0;y[4]=z[4]=80;

x[5]=y[5]=0;z[5]=80;

x[6]=80;y[6]=0;z[6]=80;

x[7]=y[7]=z[7]=80;

printf("\n default values have been set");

}

printf("\n\n now enter the projection vector \t");

scanf("%d%d%d",&a,&b,&c);

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

{

xr[i]=x[i]-(a\*z[i]/c);

yr[i]=y[i]-(b\*z[i]/c);

zr[i]=0;

}

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

setcolor(CYAN);

line(300,0,300,480);

line(0,240,600,240);

setcolor(YELLOW);

for(i=0,j=i+1;i<8;i++,j=(j+1)/8);

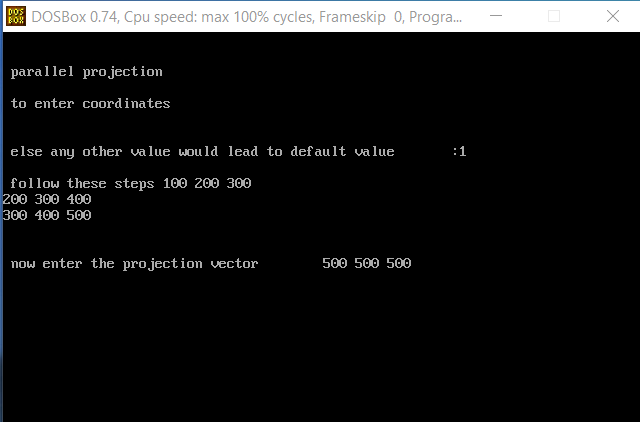
line(xr[i]+300,240-yr[i],xr[j]+300,240-yr[i]);

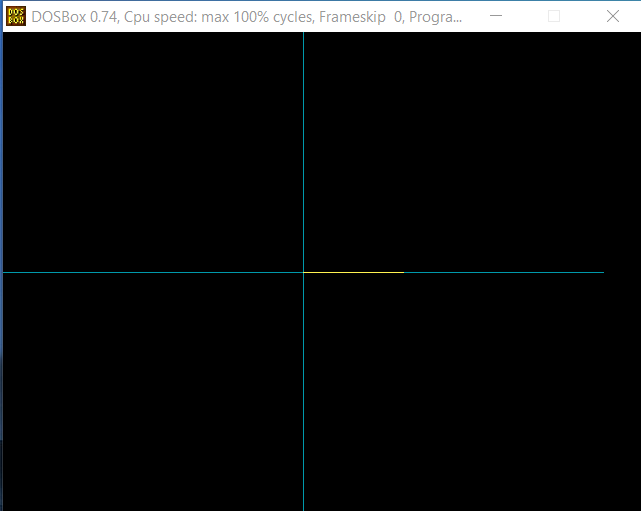
getch();

closegraph();

}

Output:





Perspective:

#include<stdio.h>

#include<math.h>

#include<graphics.h>

main()

{

int x1,y1,x2,y2,gd,gm;

int ymax,a[4][8];

float par[4][4],b[4][8];

int i,j,k,m,n,p;

int xp, yp, zp, x, y, z;

a[0][0] = 100; a[1][0] = 100; a[2][0] = -100;

a[0][1] = 200; a[1][1] = 100; a[2][1] = -100;

a[0][2] = 200; a[1][2] = 200; a[2][2] = -100;

a[0][3] = 100; a[1][3] = 200; a[2][3] = -100;

a[0][4] = 100; a[1][4] = 100; a[2][4] = -200;

a[0][5] = 200; a[1][5] = 100; a[2][5] = -200;

a[0][6] = 200; a[1][6] = 200; a[2][6] = -200;

a[0][7] = 100; a[1][7] = 200; a[2][7] = -200;

detectgraph(&gd,&gm);

initgraph(&gd,&gm, "c:\\tc\\bgi");

ymax = getmaxy();

xp = 300; yp = 320; zp = 100;

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

{

x = a[0][j]; y = a[1][j]; z = a[2][j];

b[0][j] = xp - ( (float)( x - xp )/(z - zp)) \* (zp);

b[1][j] = yp - ( (float)( y - yp )/(z - zp)) \* (zp);

}

/\*- front plane display -\*/

for(j=0;j<3;j++)

{

x1=(int) b[0][j]; y1=(int) b[1][j];

x2=(int) b[0][j+1]; y2=(int) b[1][j+1];

line( x1,ymax-y1,x2,ymax-y2);

}

x1=(int) b[0][3]; y1=(int) b[1][3];

x2=(int) b[0][0]; y2=(int) b[1][0];

line( x1, ymax-y1, x2, ymax-y2);

/\*- back plane display -\*/

setcolor(11);

for(j=4;j<7;j++)

{

x1=(int) b[0][j]; y1=(int) b[1][j];

x2=(int) b[0][j+1]; y2=(int) b[1][j+1];

line( x1, ymax-y1, x2, ymax-y2);

}

x1=(int) b[0][7]; y1=(int) b[1][7];

x2=(int) b[0][4]; y2=(int) b[1][4];

line( x1, ymax-y1, x2, ymax-y2);

setcolor(7);

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

{

x1=(int) b[0][i]; y1=(int) b[1][i];

x2=(int) b[0][4+i]; y2=(int) b[1][4+i];

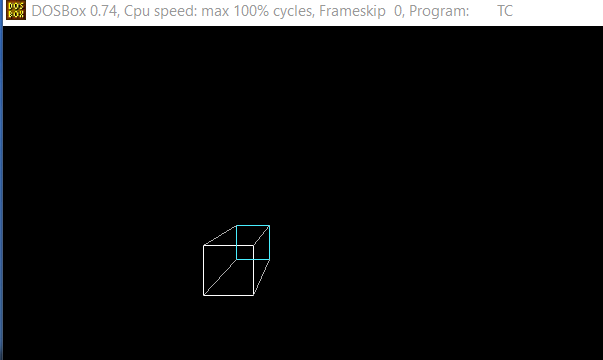
line( x1, ymax-y1, x2, ymax-y2);

}

getch();

}

Output:



Program 10:

#include <stdio.h>

#include <graphics.h>

#include <math.h>

int x[4]={200,100,200,250};

int y[4]={200,150,75,100};

void bezier ()

{

int i;

double t,xt,yt;

for (t = 0.0; t < 1.0; t += 0.0005)

{

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];

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();

}

void main()

{

int gd = DETECT, gm;

initgraph (&gd, &gm, "C:\\Turboc3\\bgi");

bezier ();

}

Output:

