MD AKRAM KHAN

University Roll Number: 50011801 1055

University Registration Number: 181430110051

Subject Name: Computer Graphics

Subject Code: CS591

Semester: 5th

Session: 2020-21

INDEX

Write a c program to draw a house, boy, tree, moon and write your name using the concept of Computer Graphics.

Code:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
int main()
{
    int gd=DETECT, gm;
    int x, y, i;
    initgraph(&gd, &gm,"c:\\TC\\BGI");
    x = getmaxx();
    y = getmaxy();
    circle(250,250,50);
    line(250,300,250,450);
    line(250,345,320,300);
    line(250,450,320,480);
    line(250,345,180,300);
    line(250,450,180,480);
    getch();
    closegraph();
}
```

Output:

Write a c program to draw a house, boy, tree,sun and write your name using the concept of Computer Graphics.

Code:

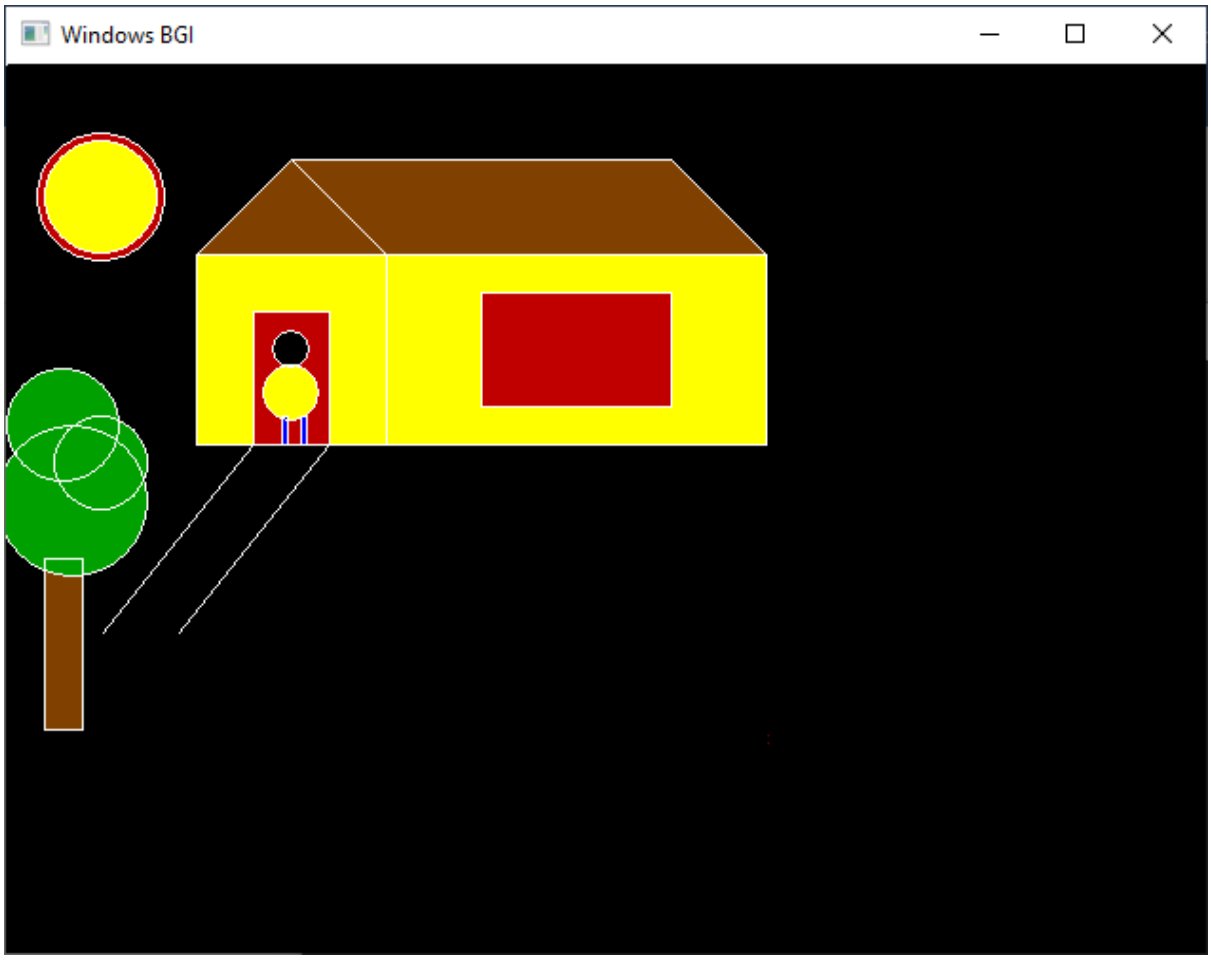
```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
int main()
{
    int gdriver=DETECT, gmode;
    initgraph(&gdriver, &gmode,"c:\\TC\\BGI");
```

```
// House
rectangle(100,100,200,200);
rectangle(200,200,400,100);
rectangle(130,130,170,200);
rectangle(250,120,350,180);

line(100,100,150,50);
line(150,50,200,100);
line(150,50,350,50);
line(350,50,400,100);

//Road
line(130,200,50,300);
line(170,200,90,300);

//Tree
circle(30,190,30);
circle(35,230,40);
circle(50,210,25);
rectangle(20,260,40,350);

//Sun
circle(50,70,30);
circle(50,70,34);
```

```
// Man
circle(150,150,10);
circle(150,173,15);

rectangle(145,185,148,200);
rectangle(155,185,158,200);

setfillstyle(SOLID_FILL,RED);
floodfill(131,131,WHITE);
floodfill(251,121,WHITE);

setfillstyle(SOLID_FILL,YELLOW);
floodfill(101,101,WHITE);
floodfill(201,101,WHITE);

setfillstyle(SOLID_FILL,BROWN);
floodfill(163,55,WHITE);
floodfill(150,52,WHITE);

setfillstyle(SOLID_FILL,YELLOW);
floodfill(51,71,WHITE);
setfillstyle(SOLID_FILL,RED);
floodfill(80,70,WHITE);
setfillstyle(SOLID_FILL,GREEN);
floodfill(31,180,WHITE);
```

```
    floodfill(25,215,WHITE);
    floodfill(40,215,WHITE);
    floodfill(55,195,WHITE);
    floodfill(70,205,WHITE);
    floodfill(36,231,WHITE);
    floodfill(51,211,WHITE);
    floodfill(21,261,WHITE);
    setfillstyle(SOLID_FILL,BROWN);
    floodfill(21,270,WHITE);
    setfillstyle(SOLID_FILL,RED);
    floodfill(150,190,WHITE);
    setfillstyle(SOLID_FILL,YELLOW);
    floodfill(150,160,WHITE);
    setfillstyle(SOLID_FILL,BLUE);
    floodfill(146,190,WHITE);
    setfillstyle(SOLID_FILL,BLUE);
    floodfill(156,190,WHITE);
    setcolor(RED);
    outtextxy(400,350,"Md Akram khan");
    getch();
    closegraph();
}
```

Output:



Write a c program to implement DDA line drawing algorithm.

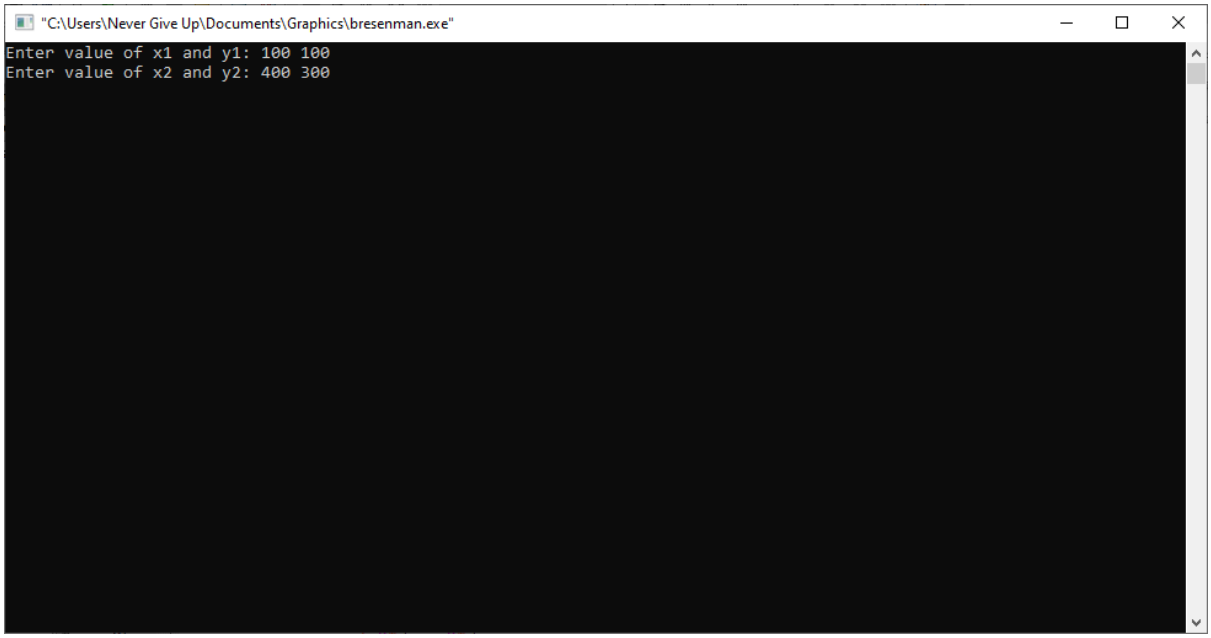
Code:

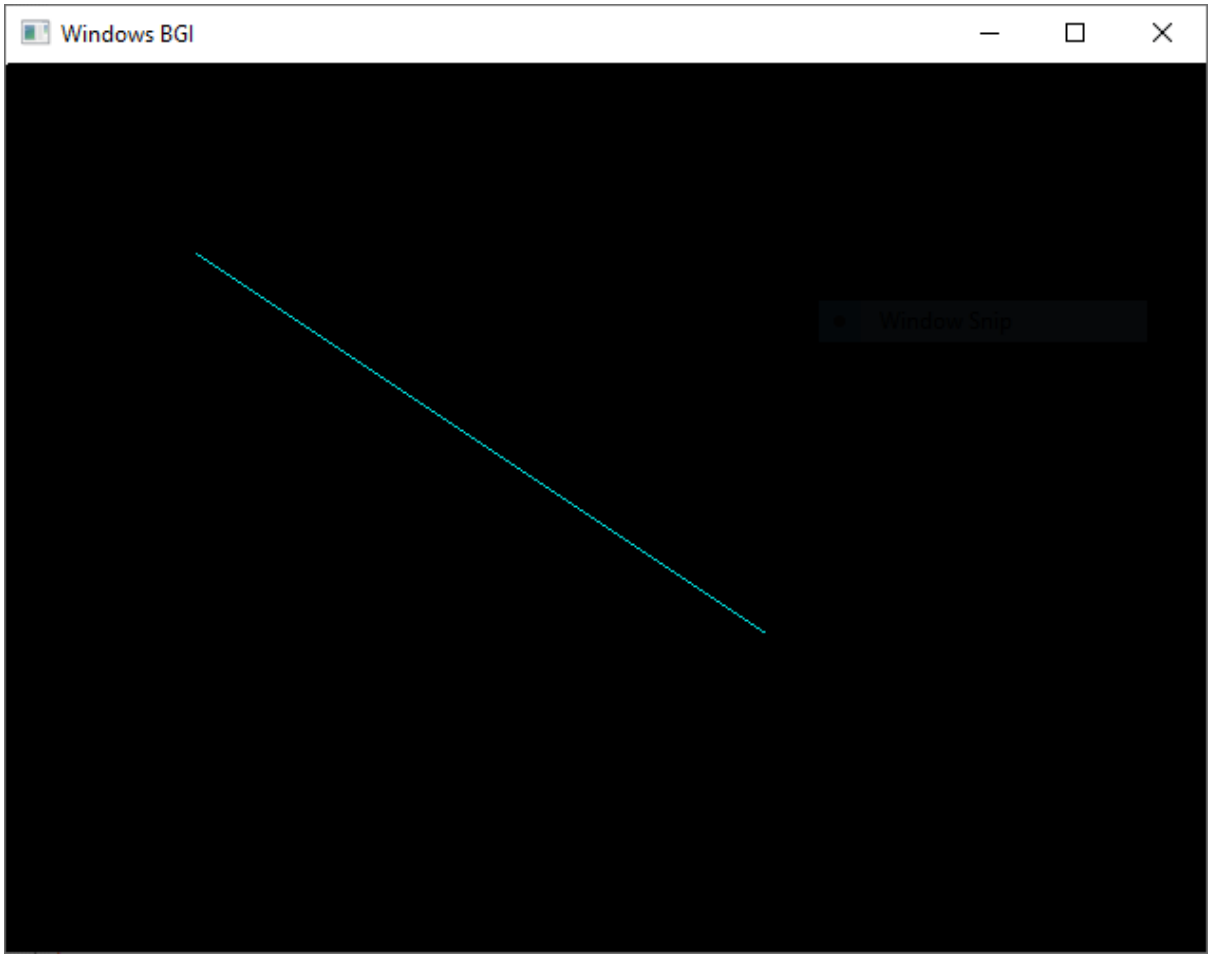
```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
```

```
int main(){
    int gdriver = DETECT, gmode, x1, y1, x2, y2, i;
    float x, y, dx, dy, steps, Xinc, Yinc;
    printf("Enter value of x1 and y1: ");
    scanf("%d %d",&x1, &y1);
    printf("Enter value of x2 and y2: ");
    scanf("%d %d",&x2, &y2);
    initgraph(&gdriver, &gmode,"c:\\TC\\BGI");
    dx = x2 - x1;
    dy = y2 - y1;
    if(abs(dx)>abs(dy))
        steps = abs(dx);
    else
        steps = abs(dy);
    Xinc = dx/steps;
    Yinc = dy/steps;
    x = x1;
    y = y1;
    for(i = 0; i < steps; i++){
        putpixel(x, y, 3);
        x = x + Xinc;
        y = y + Yinc;
    }
    getch();
    closegraph();
```

}

Output:


"C:\Users\Never Give Up\Documents\Graphics\bresenman.exe"
Enter value of x1 and y1: 100 100
Enter value of x2 and y2: 400 300


Windows BGI

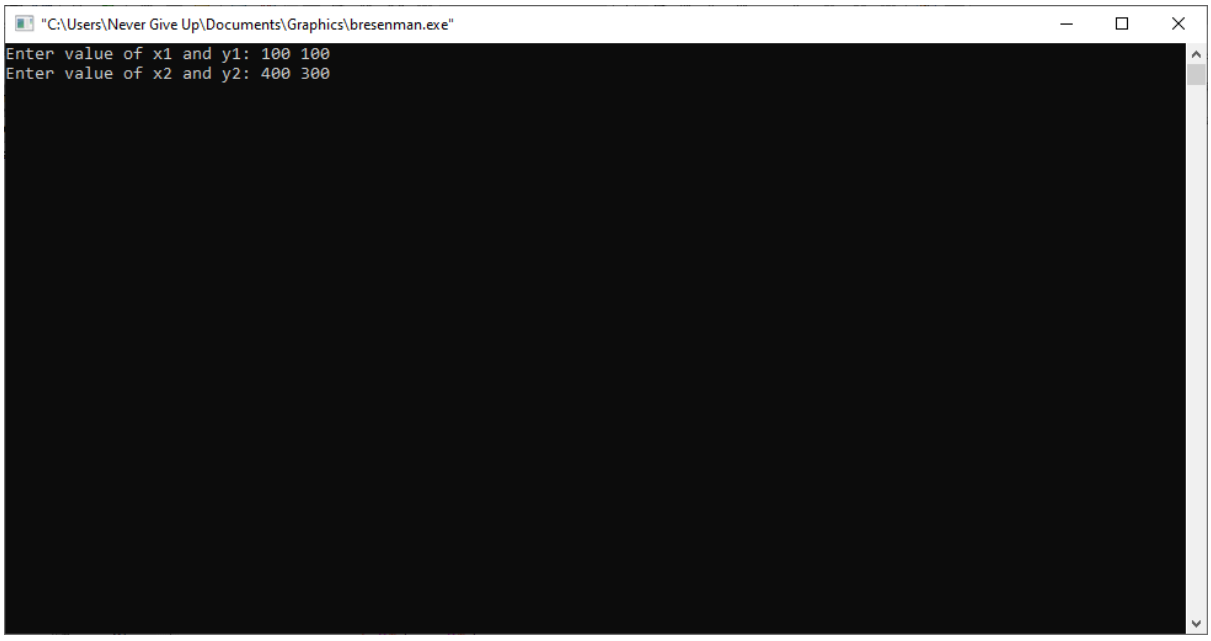
Write a c program to implement Bresenham's line drawing algorithm.

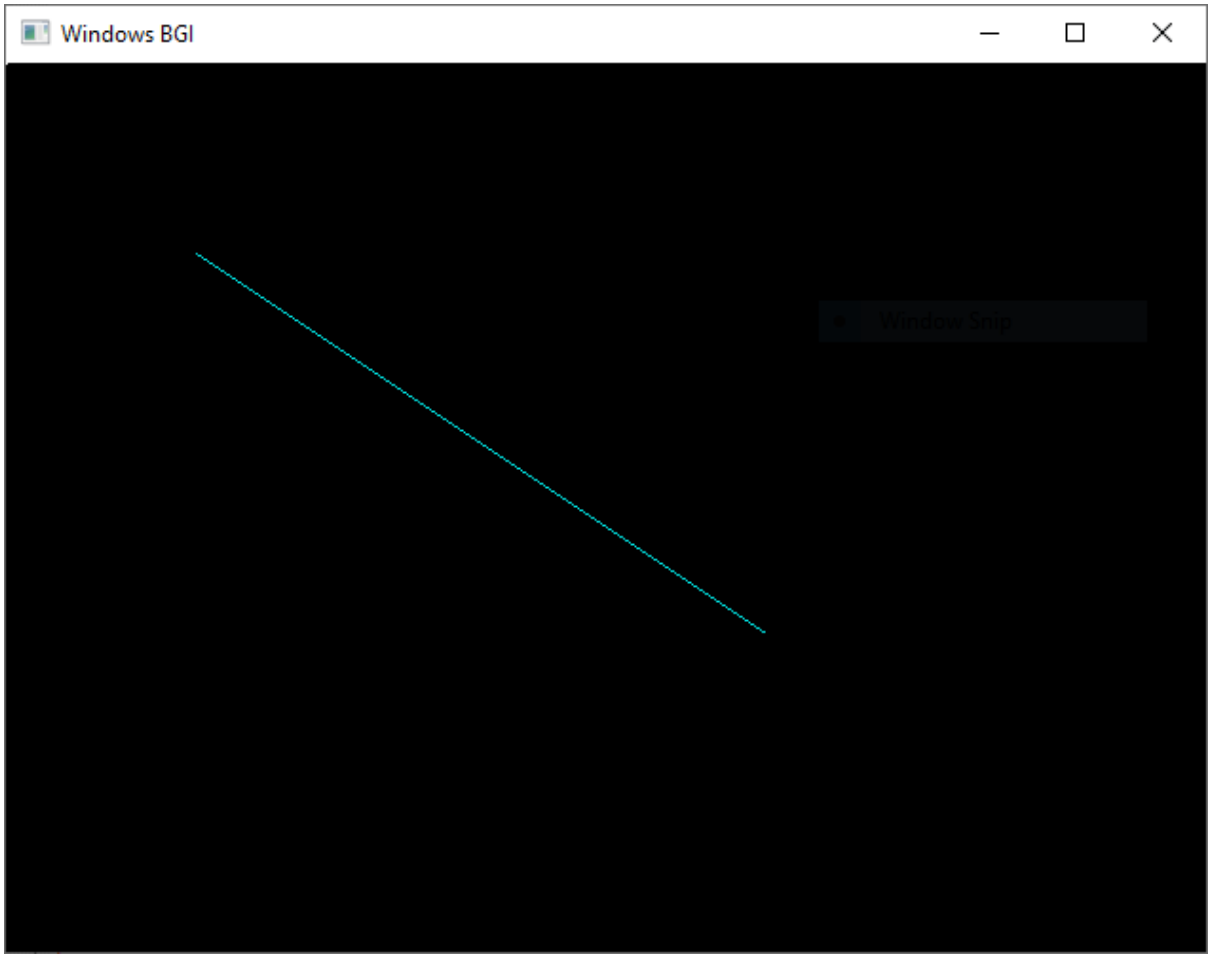
Code:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
int main(){
    int gdriver = DETECT, gmode, x1, y1, x2, y2, x, y, dx, dy, p, i;
    printf("Enter value of x1 and y1: ");
    scanf("%d %d",&x1, &y1);
    printf("Enter value of x2 and y2: ");
    scanf("%d %d",&x2, &y2);
    initgraph(&gdriver, &gmode,"c:\\TC\\BGI");
    dx = x2 - x1;
    dy = y2 - y1;
    p = 2*dy - dx;
    x = x1;
    y = y1;
    for(i = 0; i < dx; i++){
        putpixel(x, y, 3);
        if(p < 0){
            x = x+1;
            p = p + 2*dy;
```

```
        }
        else{
            x = x+1;
            y = y+1;
            p = p + 2*dy - 2*dx;
        }
    }
    getch();
    closegraph();
}
```

Output:





Write a c program to generate a circle.

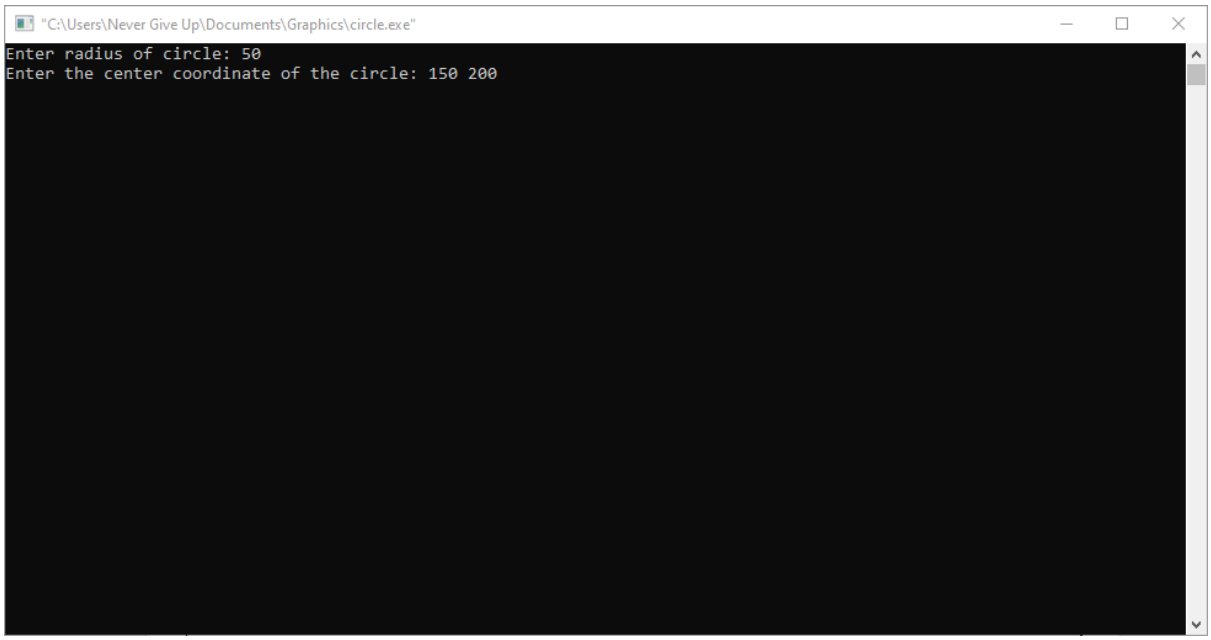
Code:

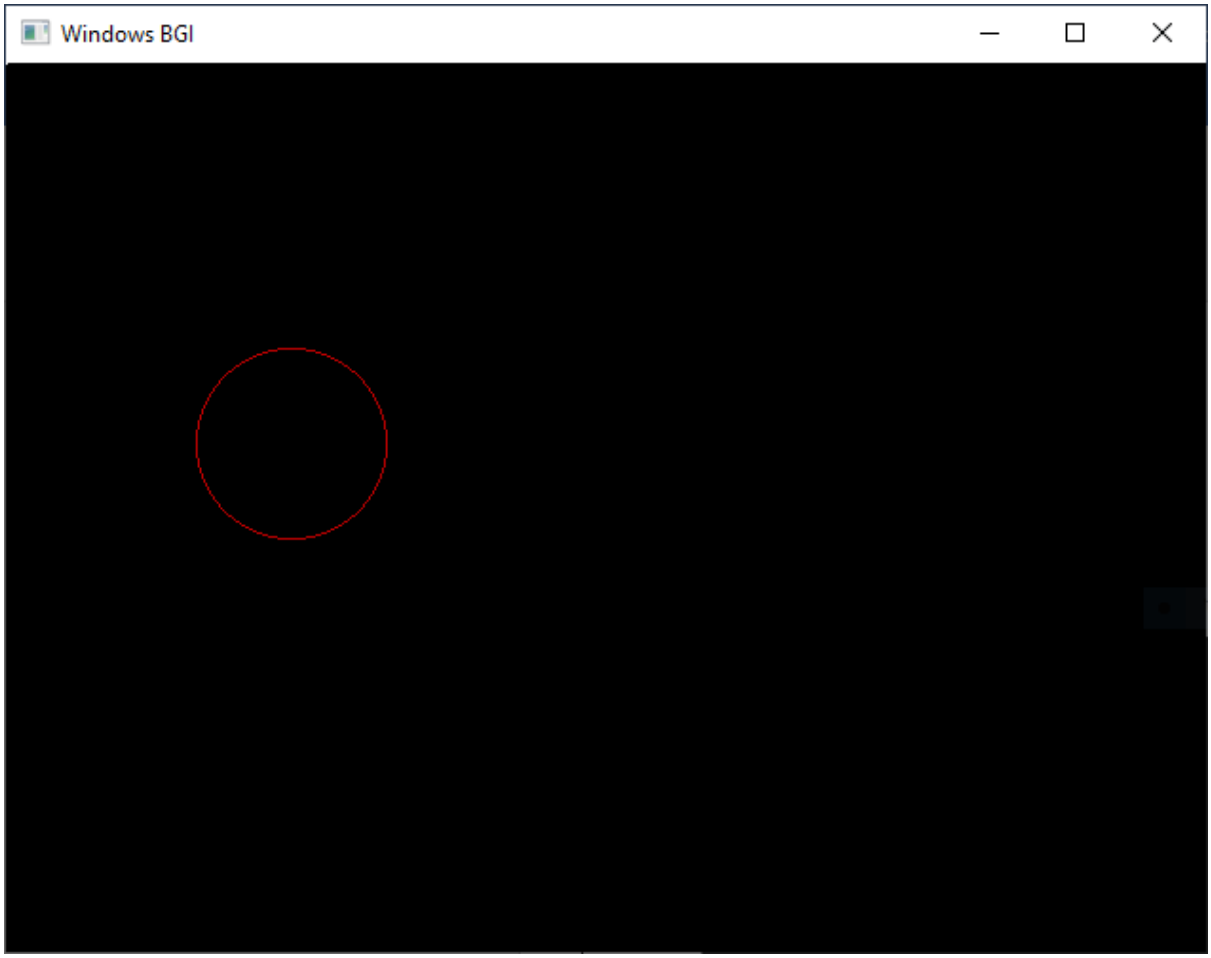
```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
int main(){
    int gdriver = DETECT, gmode, xc, yc, x, y, r, d;
    printf("Enter radius of circle: ");
    scanf("%d",&r);
    printf("Enter the center coordinate of the circle: ");
```

```
scanf("%d %d", &xc, &yc);
x = 0;
y = r;
d = 1-r;
initgraph(&gdriver, &gmode,"");

do{
    putpixel(xc+x, yc+y,4);
    putpixel(xc+x, yc-y,4);
    putpixel(xc-x, yc+y,4);
    putpixel(xc-x, yc-y,4);
    putpixel(xc+y, yc+x,4);
    putpixel(xc+y, yc-x,4);
    putpixel(xc-y, yc+x,4);
    putpixel(xc-y, yc-x,4);
    if(d<0){
        x = x+1;
        d = d + 2*x + 1;
    }
    else{
        x = x+1;
        y = y-1;
        d = d + 2*(x-y) + 1;
    }
}while(x<y);
```

```
    getch();
}
```

Output:





Write a c program to generate an ellipse.

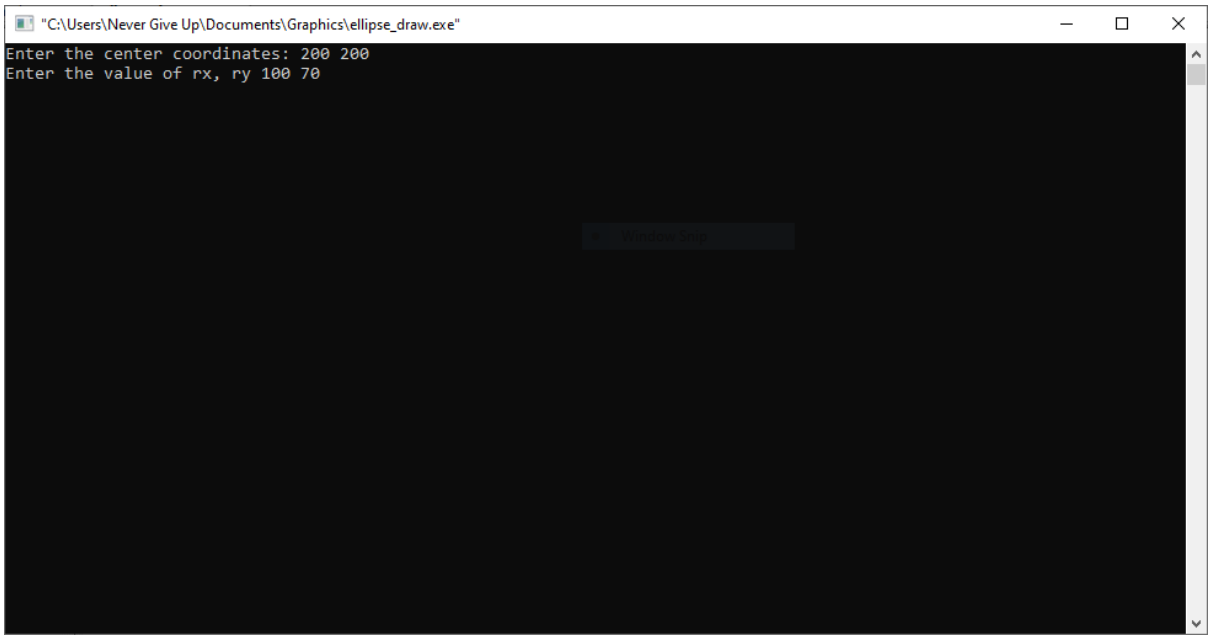
Code:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void disp();
float x, y;
int xc, yc;
main(){
    int gd = DETECT, gm;
    int rx, ry;
    float p1, p2;
    printf("Enter the center coordinates: ");
    scanf("%d %d", &xc, &yc);
    printf("Enter the value of rx, ry");
    scanf("%d %d", &rx, &ry);
    initgraph(&gd, &gm, " ");
    x = 0;
    y = ry;
    disp();
    p1 = (ry*ry)-(rx*rx*ry)+(rx*rx)/4;
    while((2.0*ry*ry*x) <= (2.0*rx*rx*y)){
        x++;
        if(p1<=0){
            p1 = p1+(2.0*ry*ry*x)+(ry*ry);
```

```
    }
    else{
        y--;
        p1 = p1+(2.0*ry*ry*x)-(2.0*rx*rx*y)+(ry*ry);
    }
    disp();
    x = -x;
    disp();
    x = -x;
}
x = rx;
y = 0;
disp();
p2 = (rx*rx)+(2.0*ry*ry*rx)+(ry*ry)/4;
while((2.0*ry*ry*x)>(2.0*rx*rx*y)){
    y++;
    if(p2>0){
        p2 = p2+(rx*rx)-(2.0*rx*rx*y);
    }
    else{
        x--;
        p2 = p2 + (2.0*ry*ry*x)-(2.0*rx*rx*y)+(rx*rx);
    }
    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);
}
```

Output:



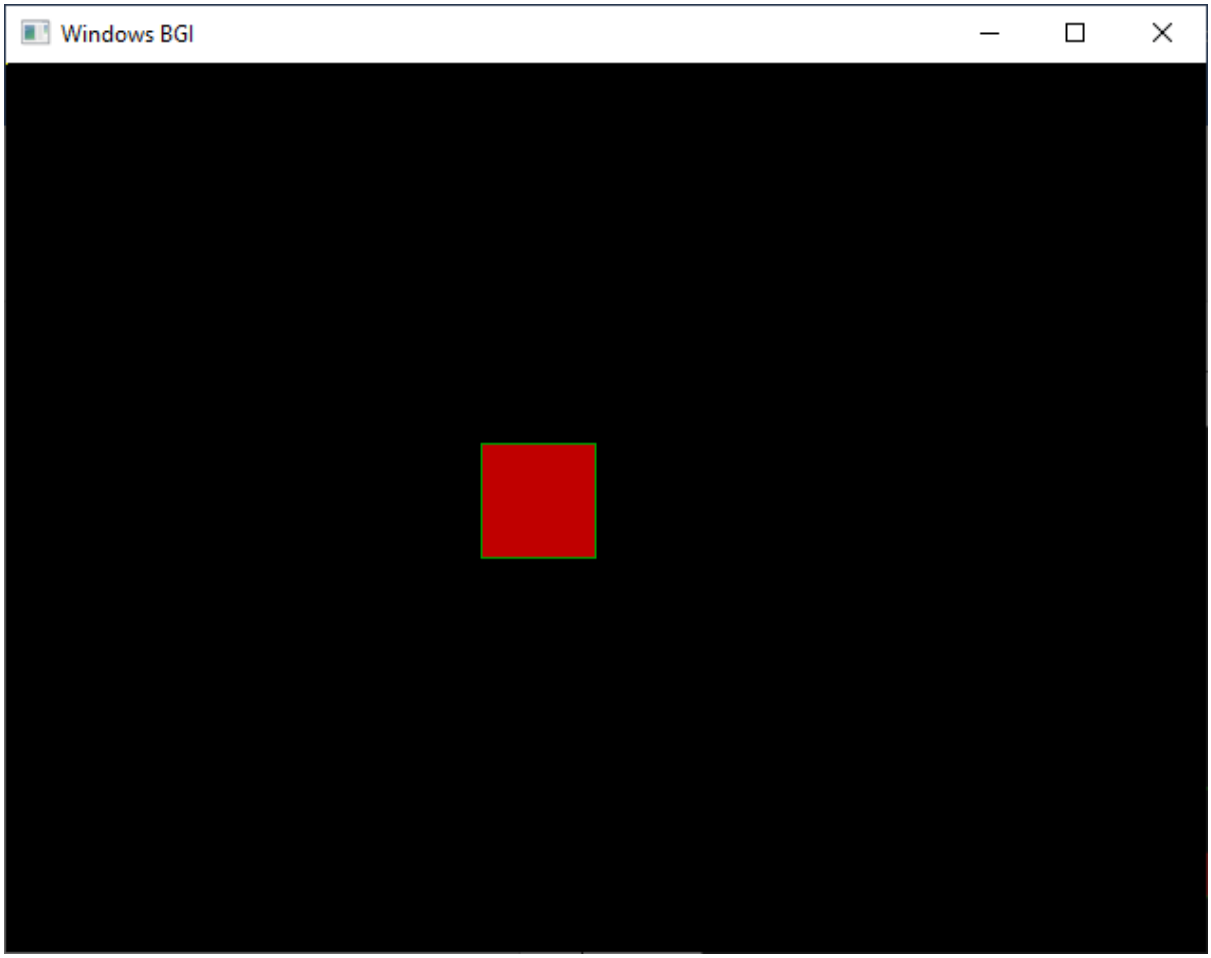
Write a c program to implement Boundary Fill algorithm.

Code:

```
#include<iostream>
#include<conio.h>
#include<graphics.h>
void boundFill(int x, int y, int f_col, int b_col){
    int current;
    current = getpixel(x, y);
        if(current != b_col && current != f_col){
                putpixel(x,y,f_col);
```

```
            delay(0.1);
            boundFill(x+1, y, f_col, b_col);
            boundFill(x-1, y, f_col, b_col);
            boundFill(x, y+1, f_col, b_col);
            boundFill(x, y-1, f_col, b_col);
        }
}
int main(){
        int gd, gm;
        gd = DETECT;
        initgraph(&gd, &gm, " ");
        setcolor(GREEN);
        rectangle(250, 200, 310, 260);
        boundFill(280, 250, RED, GREEN);
        getch();
        closegraph();

}
```

Output:



Write a c program to Flood Fill algorithm.

Code:

```
#include<iostream>
#include<conio.h>
#include<graphics.h>
void floodFill(int x, int y, int newcol, int oldcol){
        if(getpixel(x,y) == oldcol){
                putpixel(x,y,newcol);
                delay(0.1);
                floodFill(x+1, y, newcol, oldcol);
```

```
            floodFill(x-1, y, newcol, oldcol);
            floodFill(x, y+1, newcol, oldcol);
            floodFill(x, y-1, newcol, oldcol);
        }
}
int main(){
        int x, y,gm, gd = DETECT;
        initgraph(&gd, &gm, " ");
        rectangle(50,50,100,100);
        floodFill(55, 55, 4, 0);
        getch();
        closegraph();
}
```

Output:

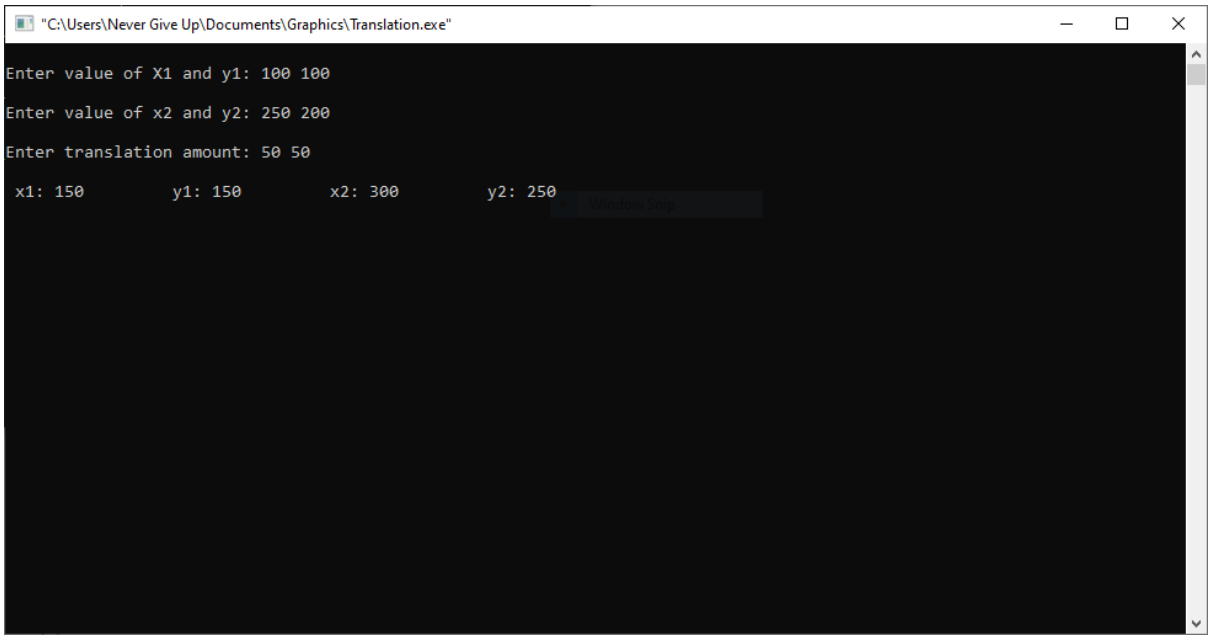
Write a c program to implement Translation.

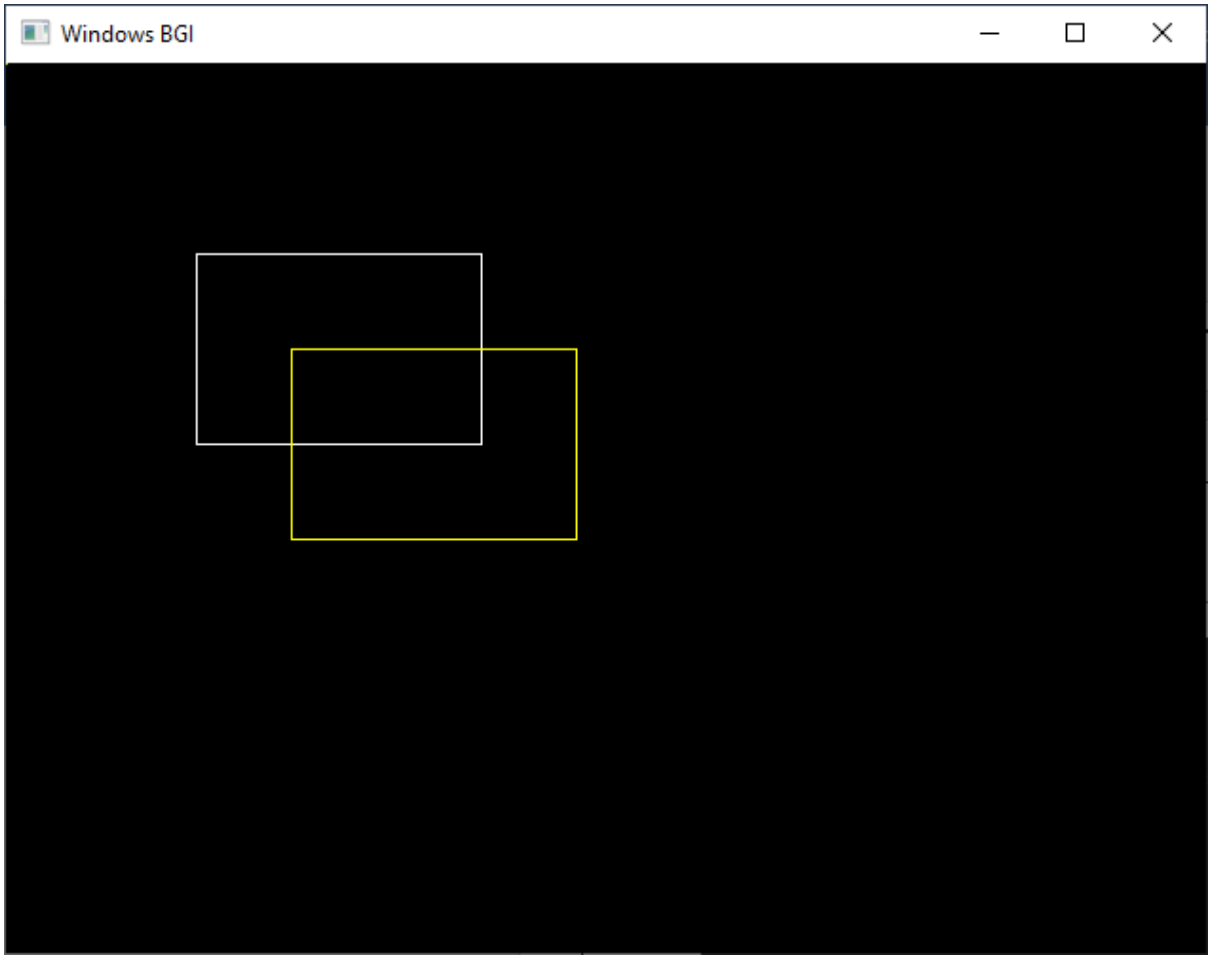
Code:

```
#include<stdio.h>
#include<stdlib.h>
#include<graphics.h>
#include<dos.h>
void translation(int, int, int, int, double, double);
int main(){
    int x1, x2, y1, y2;
    double tx, ty;
    int gdriver = DETECT, gmode;
```

```
    printf("\nEnter value of X1 and y1: ");
    scanf("%d %d", &x1, &y1);
    printf("\nEnter value of x2 and y2: ");
    scanf("%d %d", &x2, &y2);
    printf("\nEnter translation amount: ");
    scanf("%lf %lf", &tx, &ty);
    initgraph(&gdriver, &gmode, " ");
    rectangle(x1, y1, x2, y2);
    delay(1000);
    translation(x1, y1, x2, y2, tx, ty);
    closegraph();
}
void translation(int x1, int y1, int x2, int y2, double tx, double ty){
    x1 = x1+tx;
    y1 = y1+ty;
    x2 = x2+tx;
    y2 = y2+ty;
    printf("\n x1: %d\t y1: %d\t x2: %d\t y2: %d", x1, y1, x2, y2);
    setcolor(YELLOW);
    rectangle(x1, y1, x2, y2);
    getch();
}
```

Output:


"C:\Users\Never Give Up\Documents\Graphics\Translation.exe"
Enter value of X1 and y1: 100 100
Enter value of x2 and y2: 250 200
Enter translation amount: 50 50
x1: 150 y1: 150 x2: 300 y2: 250


Windows BGI

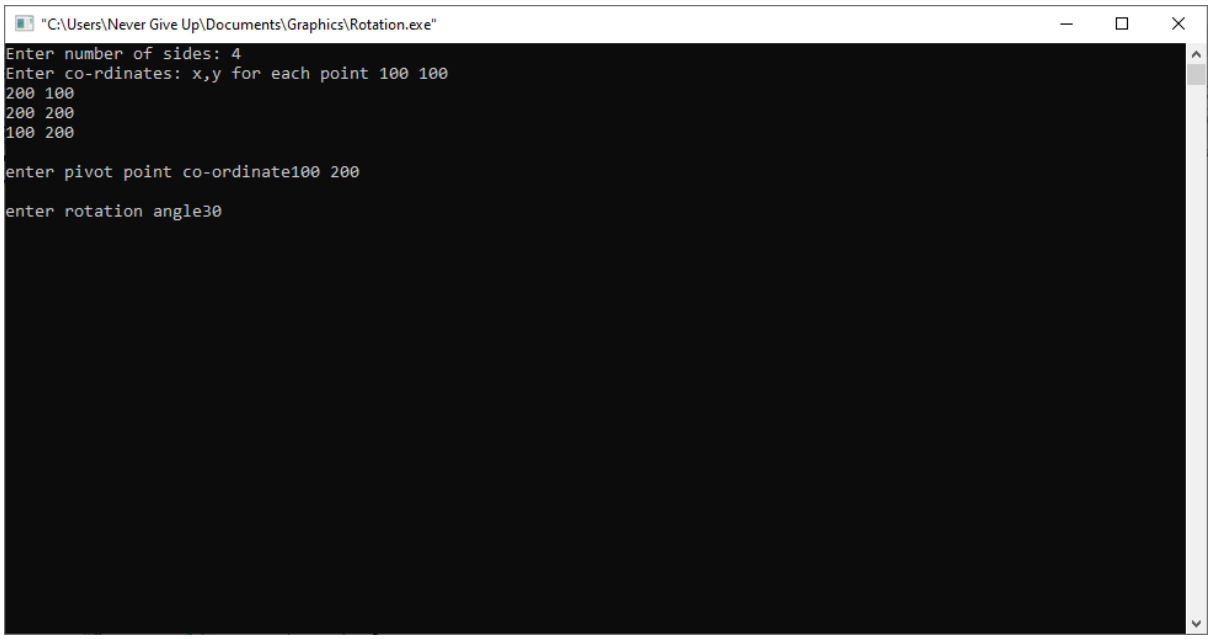
Write a c program to implement Rotation.

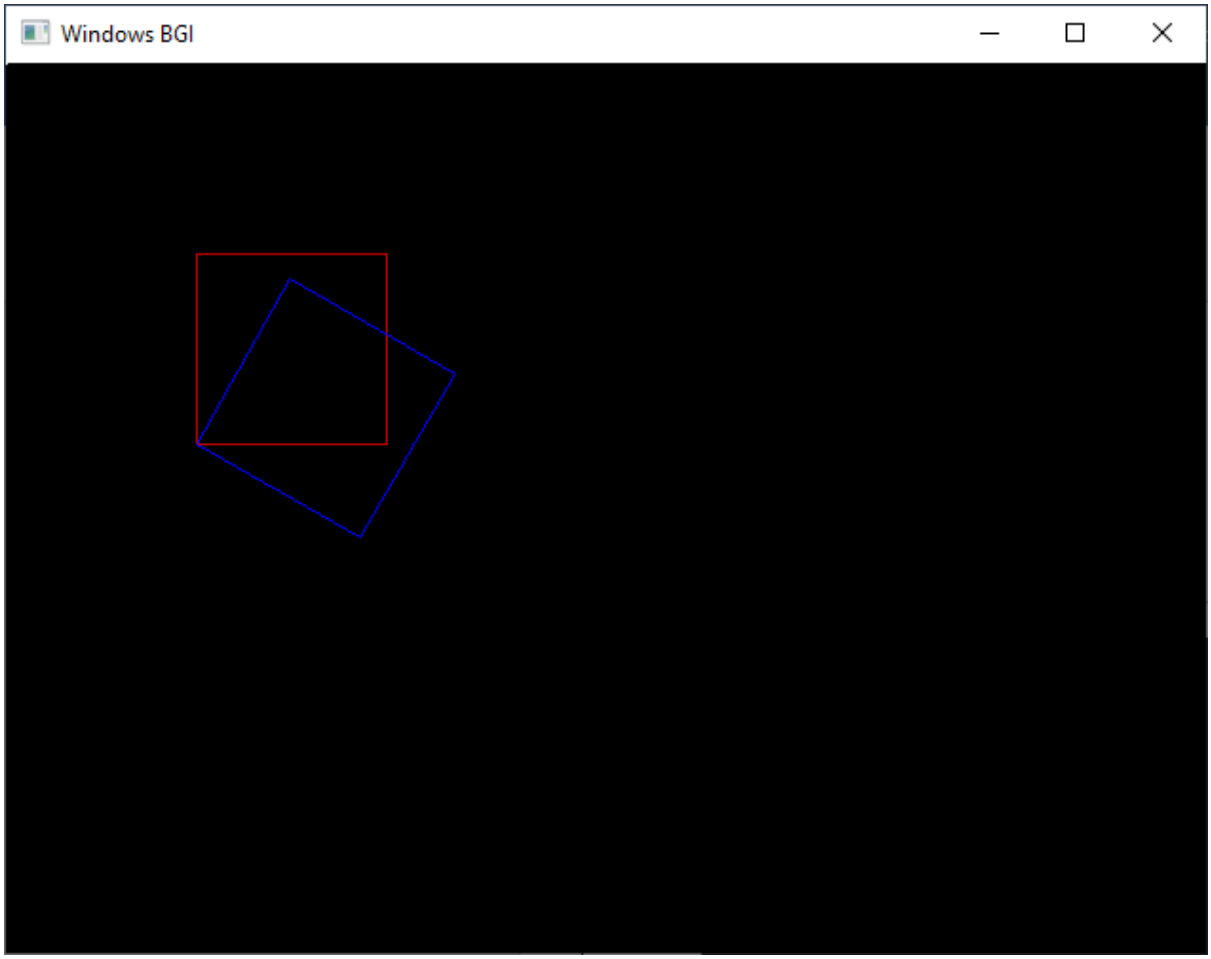
Code:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
int gdriver=DETECT,gmode;
int n,xs[100],ys[100],i,xp,yp,degree;
float radian;
void rotation();
void Draw();
void main()
{
        printf("Enter number of sides: ");
        scanf("%d",&n);
        printf("Enter co-rdinates: x,y for each point ");
        for(i=0;i<n;i++)
                scanf("%d%d",&xs[i],&ys[i]);
        printf("\nenter pivot point co-ordinate");
        scanf("%d%d",&xp,&yp);
        printf("\nenter rotation angle");
        scanf("%d",&degree);
        radian=(float)degree*3.14f/180;
        initgraph(&gdriver,&gmode,"C:\\TURBOC3\\BGI\\");
```

```
        setcolor(RED);
        Draw();
        rotation();
        setcolor(BLUE);
        Draw();
        getch();
}
void Draw()
{
for(i=0;i<n;i++)
        line(xs[i],ys[i],xs[(i+1)%n],ys[(i+1)%n]);
}
void rotation()
{
        float t,v;
        for(i=0;i<n;i++){
                t=xs[i]-xp;
                v=ys[i]-yp;
                xs[i]=xp+floor(t*cos(radian)-v*sin(radian));
                ys[i]=yp+floor(v*cos(radian)+t*sin(radian));
        }
}
```

Output:


"C:\Users\Never Give Up\Documents\Graphics\Rotation.exe"
Enter number of sides: 4
Enter co-rdinates: x,y for each point 100 100
200 100
200 200
100 200
enter pivot point co-ordinate100 200
enter rotation angle30


Windows BGI

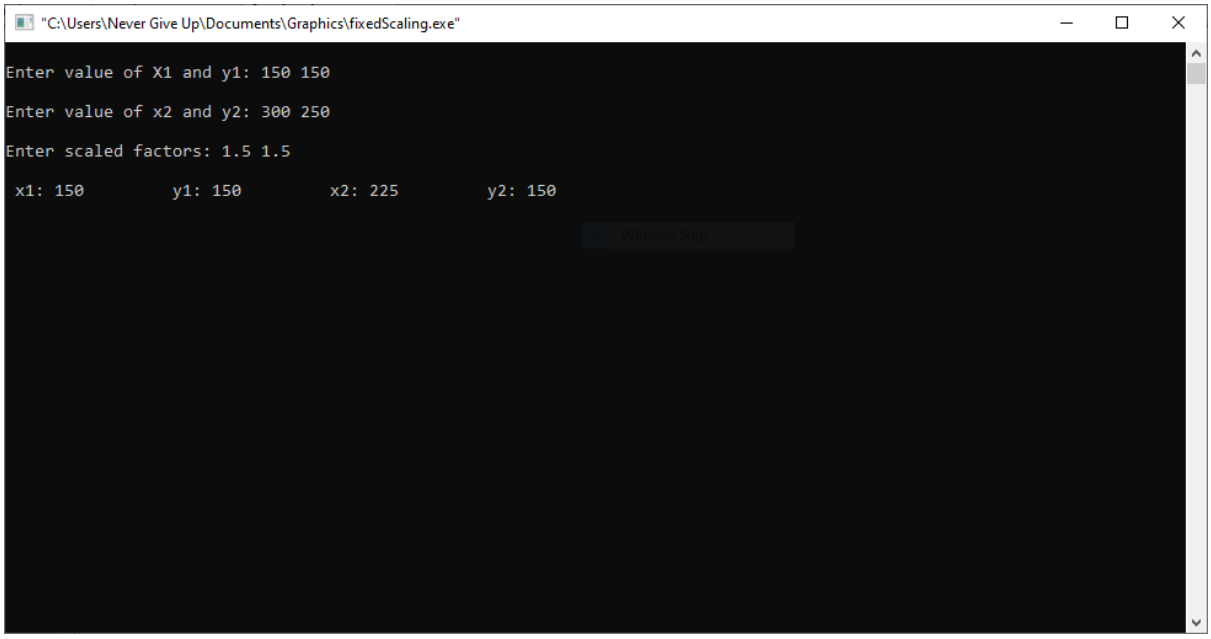
Write a c program to implement Fixed Scaling.

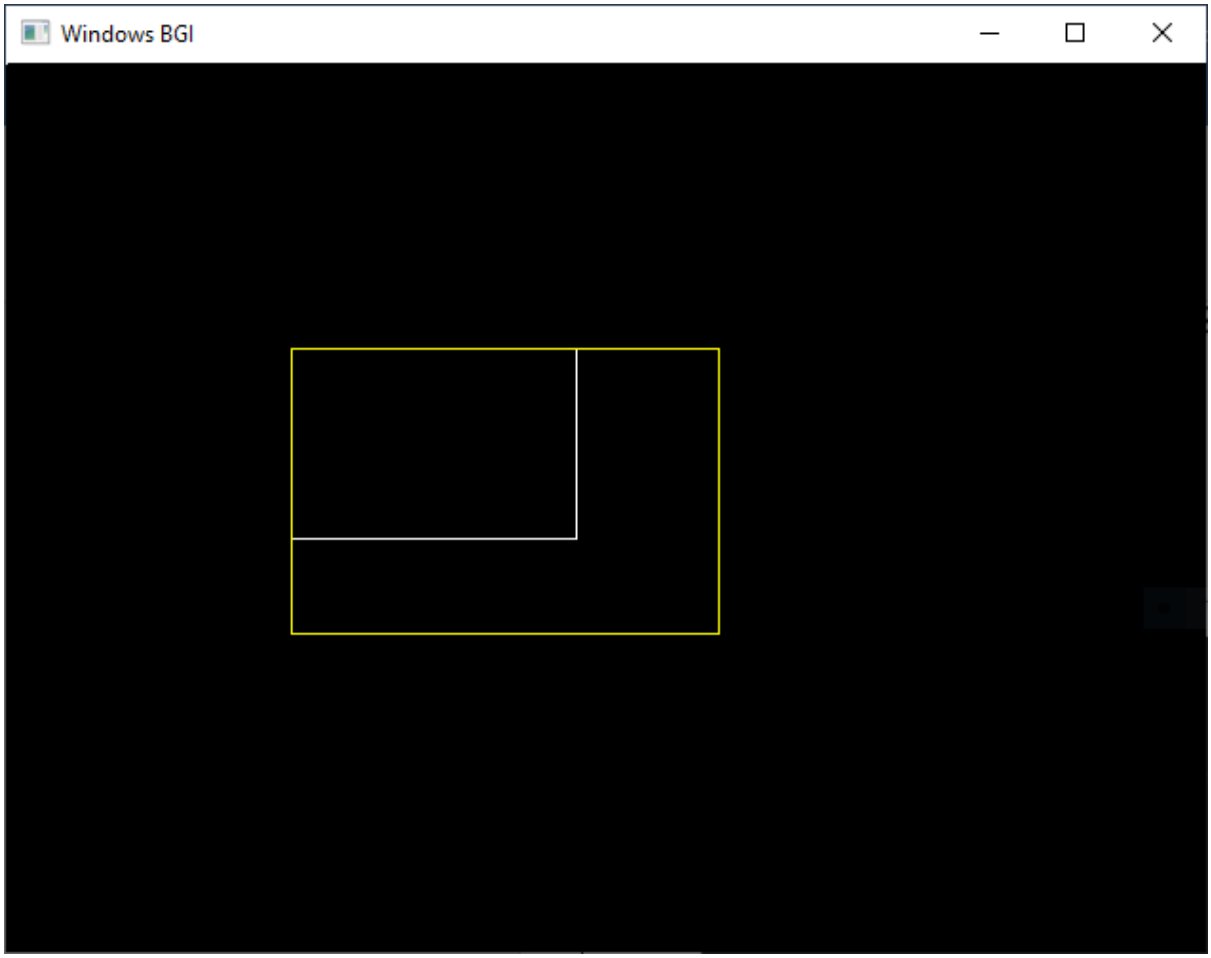
Code:

```
#include<stdio.h>
#include<stdlib.h>
#include<graphics.h>
#include<dos.h>
void fixedScaling(int, int, int, int, double, double);
int main(){
    int x1, x2, y1, y2;
    double sx, sy;
    int gdriver = DETECT, gmode;
    printf("\nEnter value of X1 and y1: ");
    scanf("%d %d", &x1, &y1);
    printf("\nEnter value of x2 and y2: ");
    scanf("%d %d", &x2, &y2);
    printf("\nEnter scaled factors: ");
    scanf("%lf %lf", &sx, &sy);
    initgraph(&gdriver, &gmode, " ");
    rectangle (x1, y1, x2, y2);
    delay(1000);
    fixedScaling (x1, y1, x2, y2, sx, sy);
    closegraph();
}
void fixedScaling(int x1, int y1, int x2, int y2, double sx, double sy){
    x2  = (x2-x1)*sx;
```

```
    y2 = (y2-y1)*sy;
    printf("\n x1: %d\t y1: %d\t x2: %d\t y2: %d", x1, y1, x2, y2);
    setcolor(YELLOW);
    rectangle(x1,y1, x1+x2, y1+y2);
    getch();
}
```

Output:





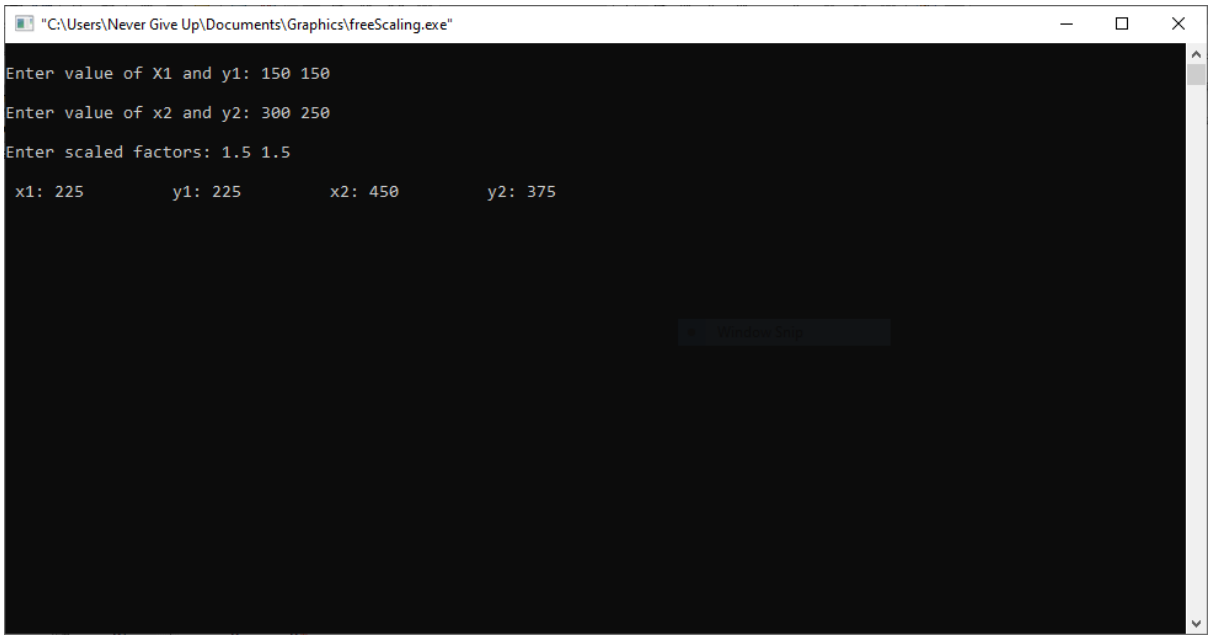
Write a c program to implement Free Scaling.

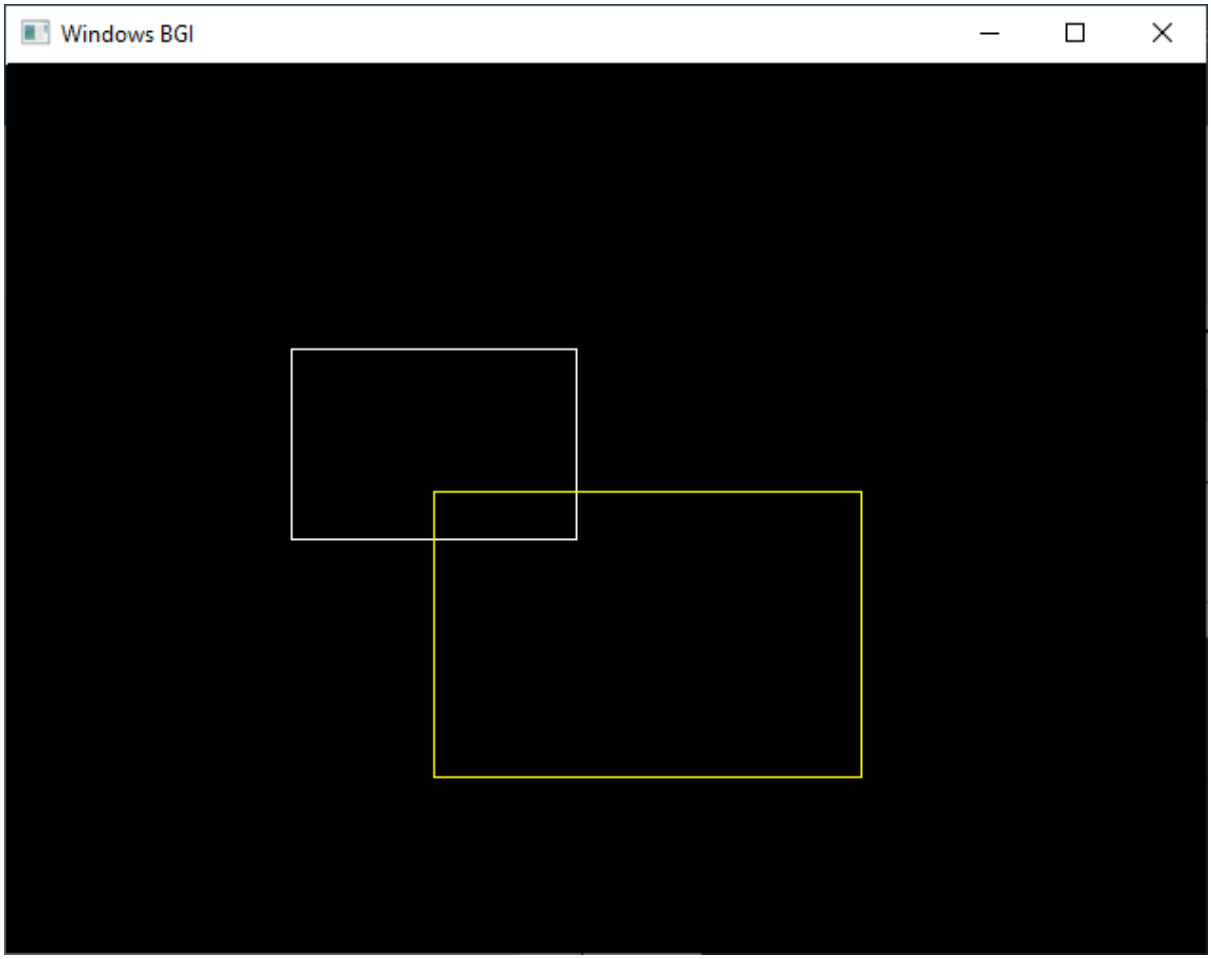
Code:

```
#include<stdio.h>
#include<stdlib.h>
#include<graphics.h>
#include<dos.h>
void freeScaling(int, int, int, int, double, double);
int main(){
    int x1, x2, y1, y2;
    double sx, sy;
```

```
    int gdriver = DETECT, gmode;
    printf("\nEnter value of X1 and y1: ");
    scanf("%d %d", &x1, &y1);
    printf("\nEnter value of x2 and y2: ");
    scanf("%d %d", &x2, &y2);
    printf("\nEnter scaled factors: ");
    scanf("%lf %lf", &sx, &sy);
    initgraph(&gdriver, &gmode, " ");
    rectangle(x1, y1, x2, y2);
    delay(10000);
    freeScaling (x1, y1, x2, y2, sx, sy);
    closegraph();
}
void freeScaling(int x1, int y1, int x2, int y2, double sx, double sy){
    x1 = x1*sx;
    y1 = y1*sy;
    x2  = x2*sx;
    y2 = y2*sy;
    printf("\n x1: %d\t y1: %d\t x2: %d\t y2: %d", x1, y1, x2, y2);
    setcolor(YELLOW);
    rectangle(x1, y1, x2, y2);
    getch();
}
```

Output:


"C:\Users\Never Give Up\Documents\Graphics\freeScaling.exe"
Enter value of X1 and y1: 150 150
Enter value of x2 and y2: 300 250
Enter scaled factors: 1.5 1.5
x1: 225 y1: 225 x2: 450 y2: 375


Windows BGI

Write a c program to implement Reflection.

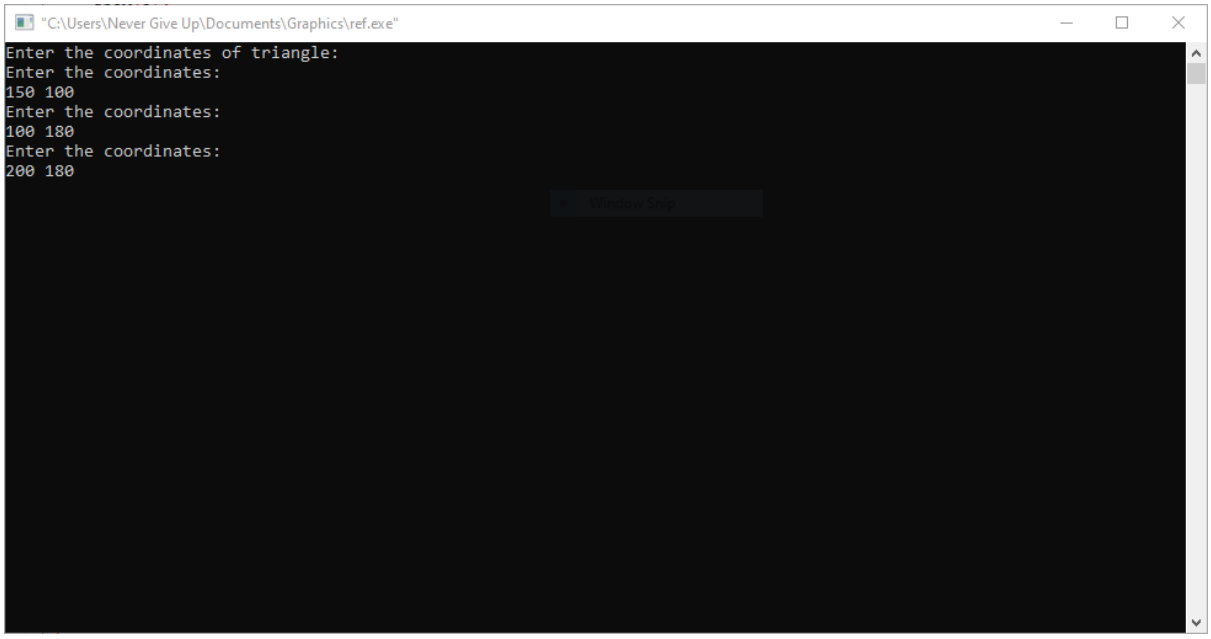
Code:

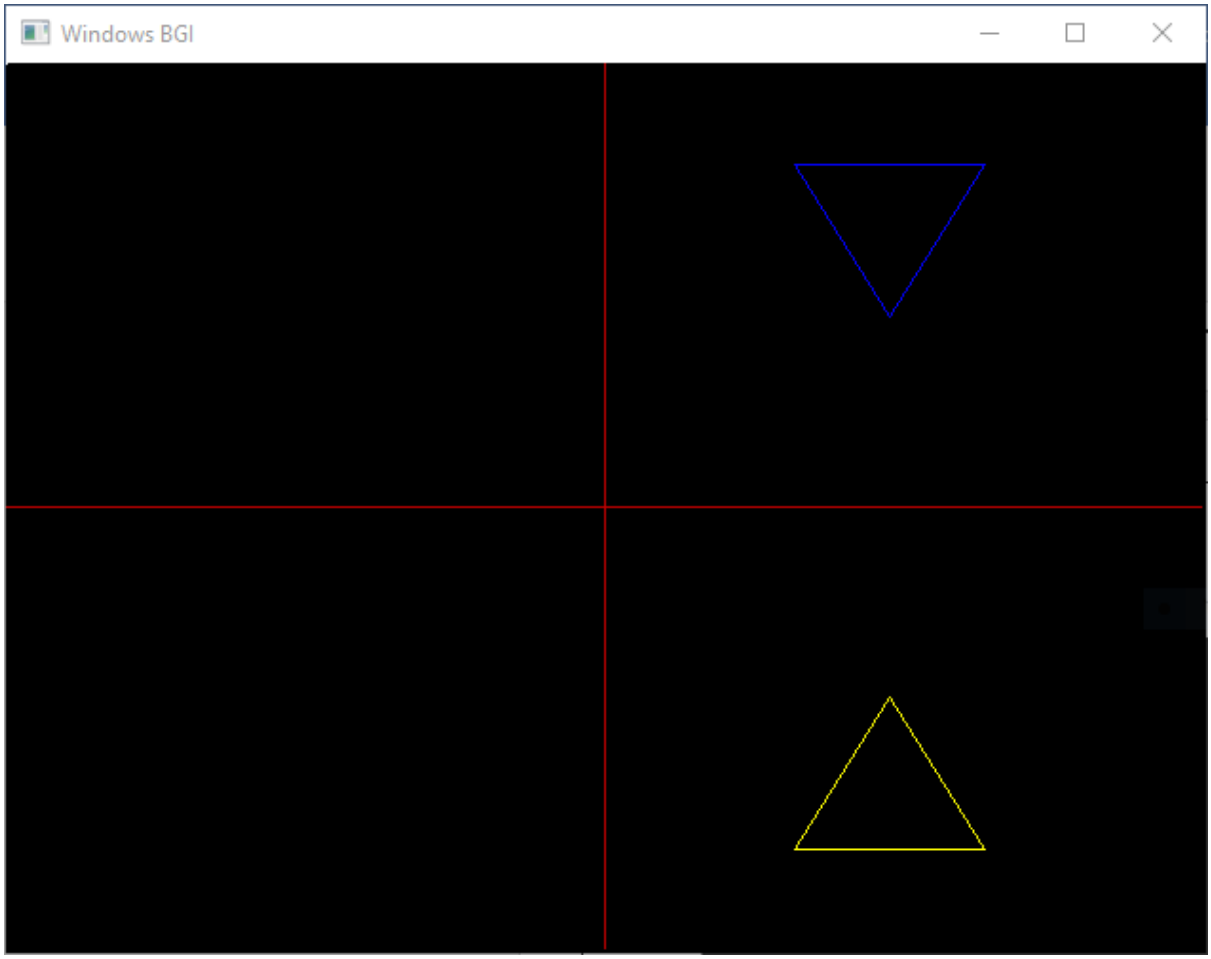
```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
void show(float a[][3]){
    float xm,ym;
    int i=0;
    xm=getmaxx()/2;
    ym=getmaxy()/2;
    while(i<2){
        line(xm+a[i][0],ym-a[i][1],xm+a[i+1][0],ym-a[i+1][1]);
        i++;
    }
    i=2;

    line(xm+a[i][0],ym-a[i][1],xm+a[0][0],ym-a[0][1]);
    setcolor(RED);
    line(0,ym,xm*2,ym);
    line(xm,0,xm,ym*2);
    setcolor(BLUE);
}
void matmul(float b[][3],float a[][3],float c[][3]){
    int i,j,m;
```

```
    for(i=0;i<3;i++){
        for(j=0;j<3;j++){
            c[i][j]=0;
            }
        }
    for(i=0;i<3;i++){
        for(j=0;j<3;j++){
            for(m=0;m<3;m++){
                c[i][j]=c[i][j]+(a[i][m]*b[m][j]);
            }
        }
    }
}
void ref(float a[][3]){
        float b[10][3],c[10][3];
        int i=0,j;
        show(a);
        for(i=0;i<3;i++){
                for(j=0;j<3;j++){
                        b[i][j]=0;
                        if(i==j)
                        b[i][j]=1;
                }
        }
        b[1][1]=-1;
```

```
        matmul(b,a,c);
        setcolor(YELLOW);
        show(c);
}
int main(){
    int gdriver=DETECT,gmode,k;
    float a[10][3];
    printf("Enter the coordinates of triangle:\n");
    for(k=0;k<3;k++){
        printf("Enter the coordinates:\n",k+1);
        scanf("%f%f",&a[k][0],&a[k][1]);
        a[k][2]=1;
    }
    initgraph(&gdriver,&gmode," ");

    setcolor(WHITE);
    show(a);
    ref(a);
    getch();
    closegraph();
    return 0;
}
```

Output:


"C:\Users\Never Give Up\Documents\Graphics\ref.exe"
Enter the coordinates of triangle:
Enter the coordinates:
150 100
Enter the coordinates:
100 180
Enter the coordinates:
200 180


Windows BGI

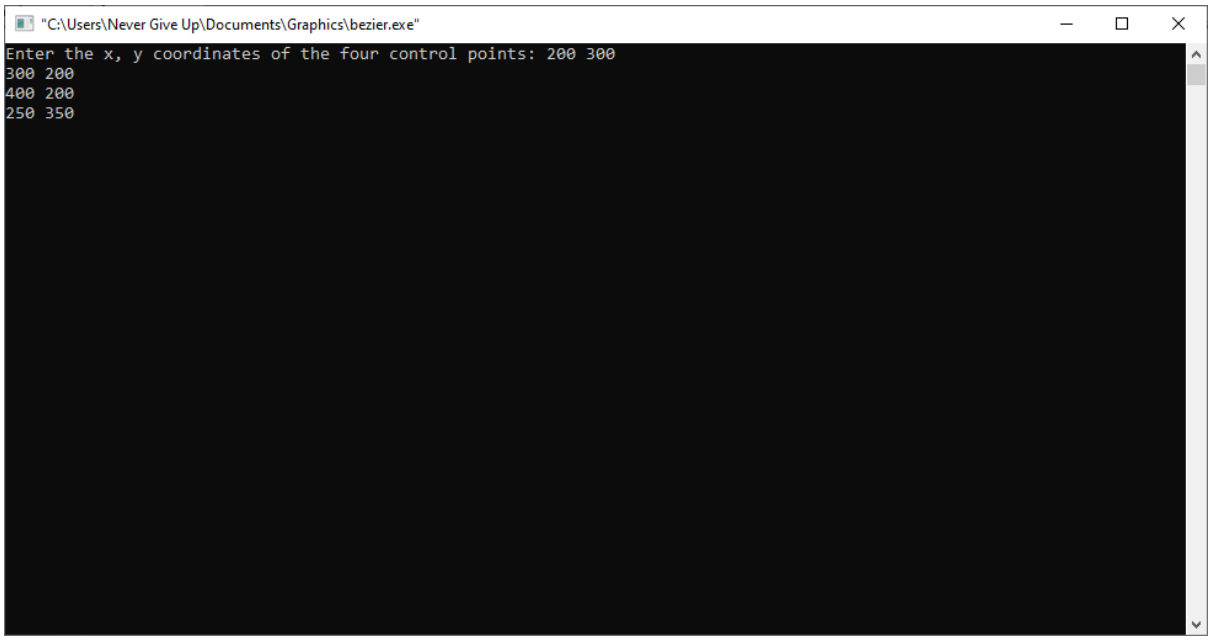
Write a c program to draw Bezier curve.

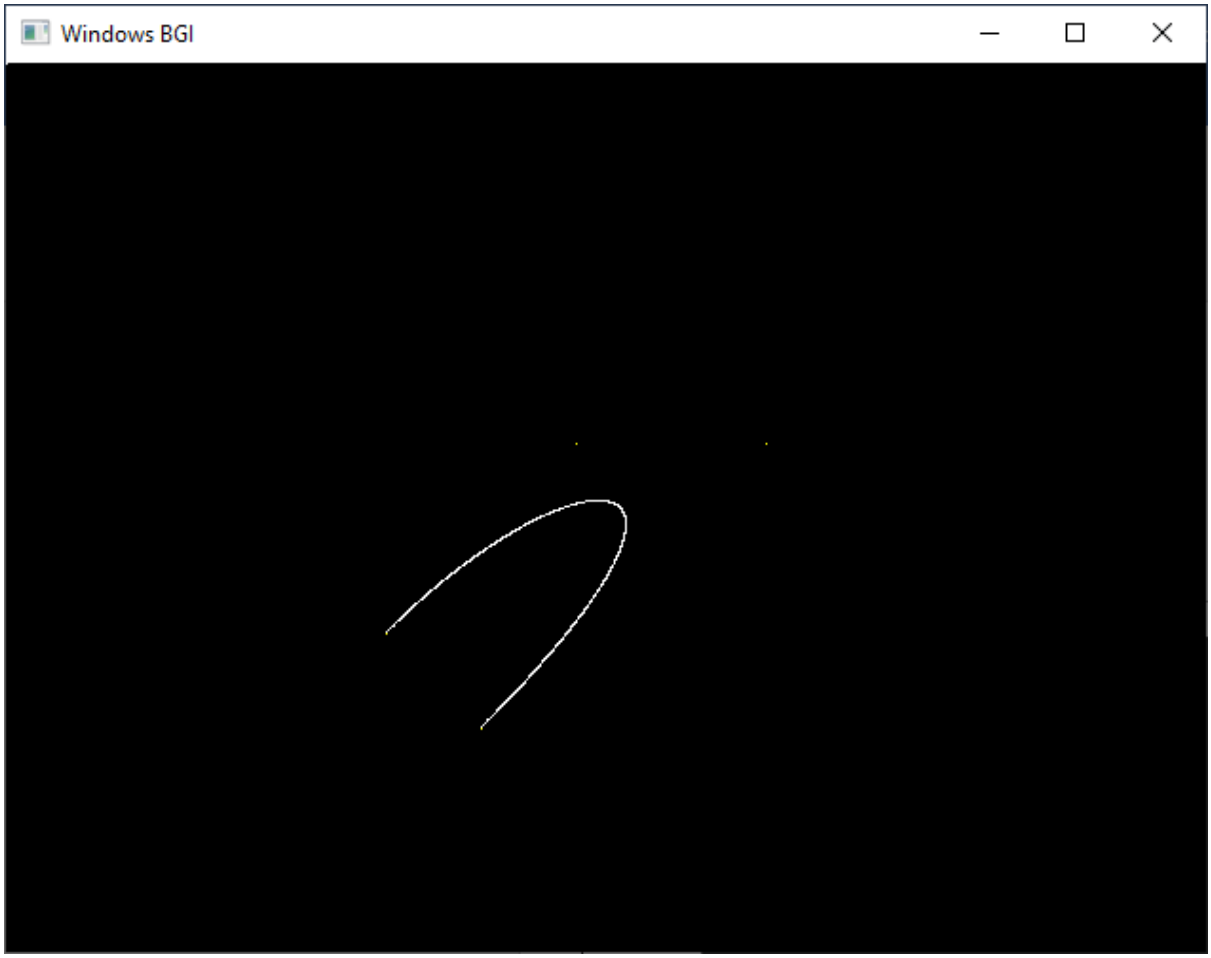
Code:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
void bezier(int x[4], int y[4]){
    int i;
    double t;
    for(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, WHITE);
    }
    for(i = 0; i<4; i++)
        putpixel(x[i], y[i], YELLOW);
    return;
}
int main(){
    int gd = DETECT, gm;
    int x[4], y[4];
    int i;
    printf("Enter the x, y coordinates of the four control points: ");
```

```
    for(i = 0; i<4; i++)
        scanf("%d %d", &x[i], &y[i]);
    initgraph(&gd, &gm, " ");
    bezier(x, y);
    getch();
    closegraph();
}
```

Output:




Windows BGI

THANKYOU