CG Exp Codes

Exp 8:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<process.h>
 main()
{
float x1,y1,x2,y2,xmin,xmax,ymin,ymax,dx,dy;
float p[4],q[4],r[4];
float max,min,u1,u2;
float xi,xii,yi,yii;
int gd,gm,i;
gd=DETECT;
initgraph(&gd,&gm,(char*)"");
//clrscr();
printf("\n enter the line co-ordinates");
printf("\n enter 1st x=");
scanf("%f",&x1);
printf("\t 1st y=");
scanf("%f",&y1);
printf("\n enter 2nd x=");
scanf("%f",&x2);
printf("\t 2nd y=");
scanf("%f",&y2);
printf("\n enter window boundry");
printf("\n xmin=");
scanf("%f",&xmin);
printf("\n ymin=");
scanf("%f",&ymin);
printf("\n xmax=");
scanf("%f",&xmax);
printf("\n ymax=");
scanf("%f",&ymax);
dx=x2-x1;
dy=y2-y1;
//cleardevice();
line(x1,y1,x2,y2);
rectangle(xmin,ymin,xmax,ymax);
p[0]=-dx;
q[0]=x1-xmin;
p[1]=dx;
q[1]=xmax-x1;
p[2]=-dy;
q[2]=y1-ymin;
p[3]=dy;
q[3]=ymax-y1;
```

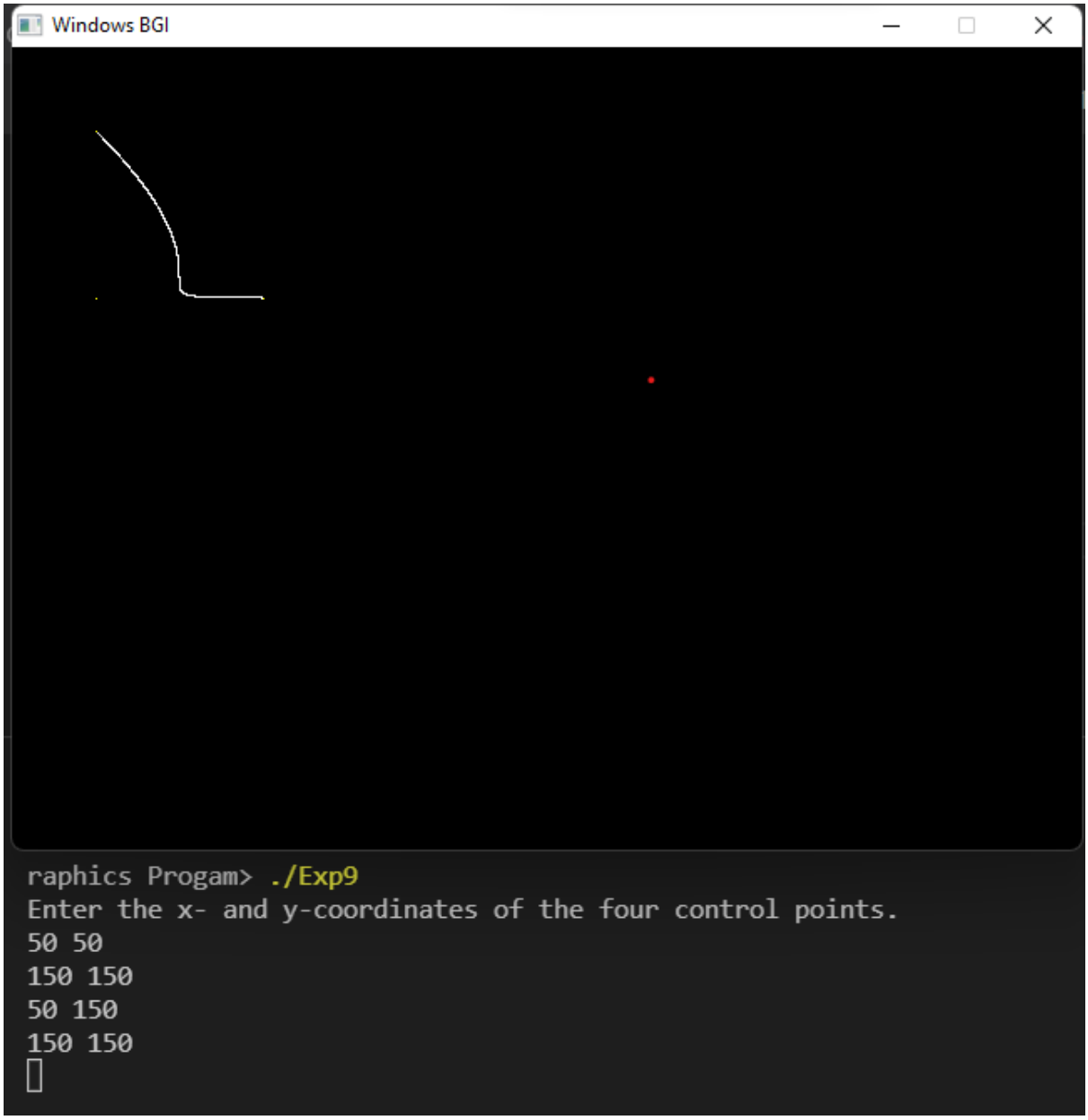
```
for(i=0;i<4;i++)
{
if(p[i]==0 && q[i]<0)
{
printf("Line is outside the boundry,it is not a clipping candidate\n");
//getch();
//exit(0);
}
}
for(i=0;i<4;i++)
{
r[i]=q[i]/p[i];
printf("\n r[%f]=%f",i,r[i]);
}
max=0;min=1;
for(i=0;i<4;i++)
{
if(p[i]<0)
{
if(r[i]>max)
max=r[i];
}
else
{
if(r[i]<min)
min=r[i];
}
}
u1=max;
u2=min;
printf("\n u1=%f",u1);
printf("\n u2=%f",u2);
if(u1>u2)
{
printf("\n line is completely outside");
//getch();
//exit(0);

}

else
{
xi=x1+(u1*dx);
yi=y1+(u1*dy);
xii=x1+(u2*dx);
yii=y1+(u2*dy);
rectangle(xmin,ymin,xmax,ymax);
setcolor(5);
```

```
line(xi,yi,xii,yii);
}
getch();
//closegraph();
}
```

Exp 9:

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

void bezier (int x[4], int y[4])
{
    int gd = DETECT, gm;
    int i;
    double t;

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

    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);

    getch();
    closegraph();
    return;
}

 main()
{
    int x[4], y[4];
    int i;

    printf ("Enter the x- and y-coordinates of the four control points.\n");
```

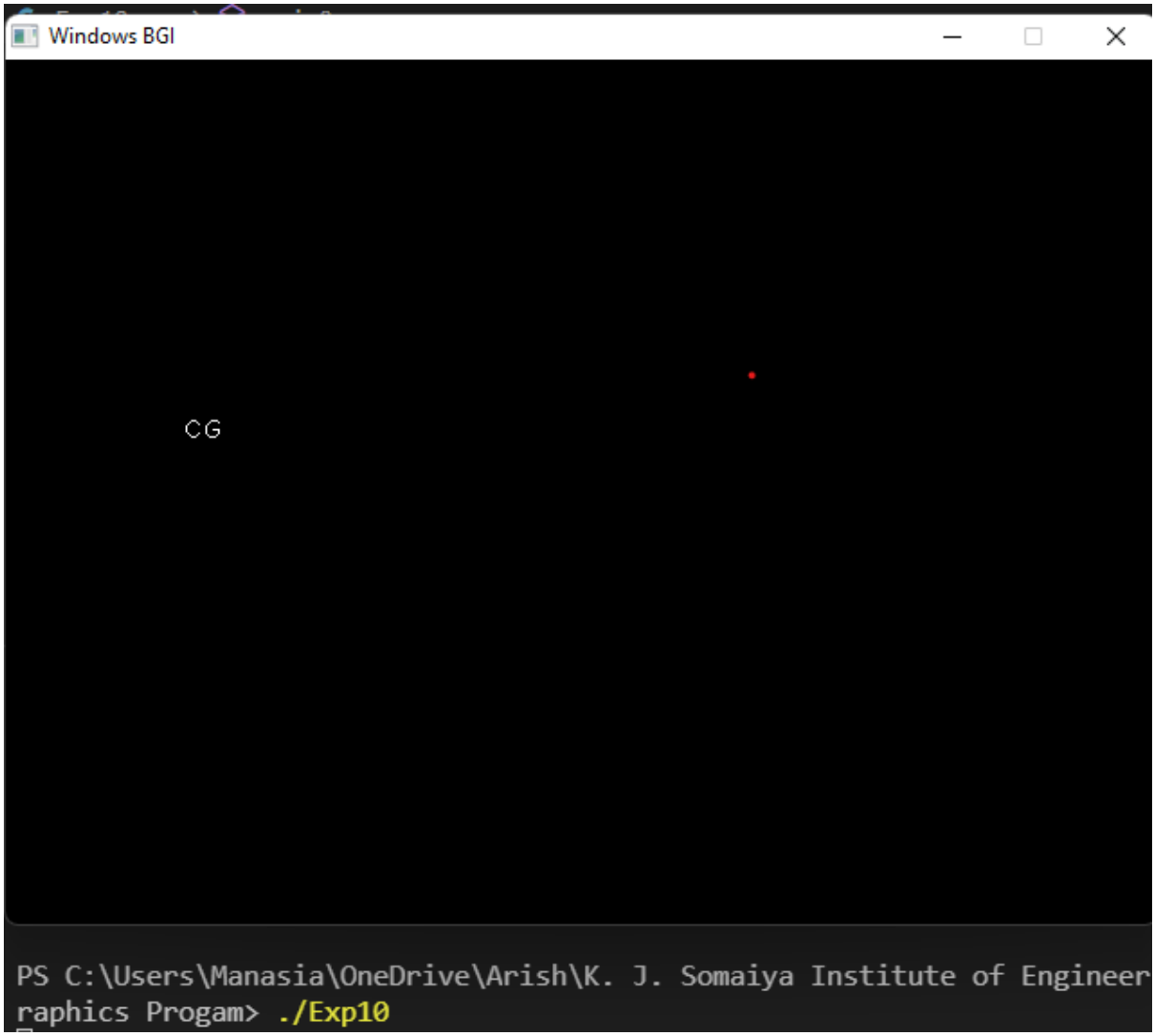
```
    for (i=0; i<4; i++)
        scanf ("%d%d", &x[i], &y[i]);

    bezier (x, y);
}
```



Exp 10:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
 main()
{
```

```
int gd=DETECT,gm,i,j;
int a[20][20]=
{{0,0,0,1,1,1,0,0,0,0,0,0,0,0,1,1,1,1,0,0},
{0,0,1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1,0},
{0,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,1},
{1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0},
{1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0},
{1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,1,1,1,0},
{1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0},
{0,1,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,0},
{0,0,1,0,0,0,1,0,0,0,0,0,1,0,0,0,0,1,0,0},
{0,0,0,1,1,1,0,0,0,0,0,0,0,1,1,1,1,0,0,0}};

 initgraph(&gd,&gm,(char*)"");
for(i=0;i<19;i++)
{
for(j=0;j<19;j++)
{
if(a[i][j]==1)
putpixel(100+j,200+i,WHITE);
}
}
getch();

}
```

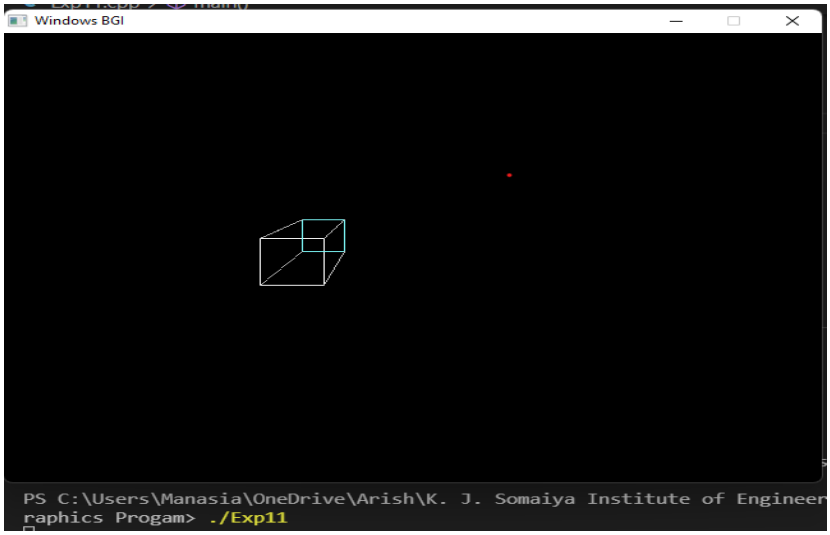


Exp 11:

```
#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, (char*)"");

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);
```

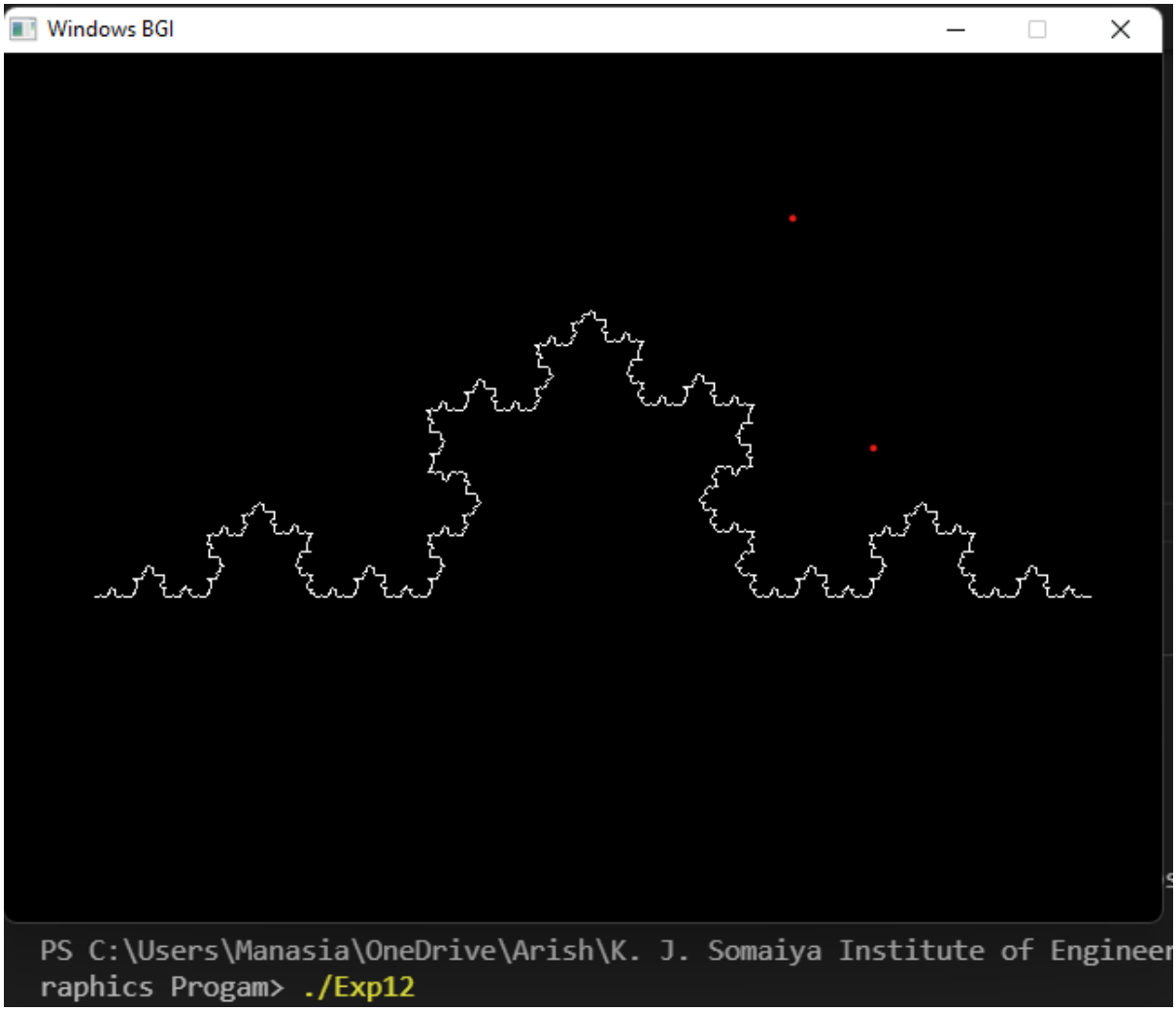
```
setcolor(7);
for(i=0;i<4;i++)
{
x1=(int) b[0][i]; y1=(int) b[1][i];
x2=(int) b[0][4+i]; y2=(int) b[1][4+i];
line( x1, ymax-y1, x2, ymax-y2);
}


getch(); getch();

}
```



Exp 12:

```
#include<stdio.h>
#include<graphics.h>
#include<stdlib.h>

#define SIN 0.866
void koch(int x1,int y1,int x2,int y2,int m)
{
    int xx,yy,x[5],y[5],lx,ly,offx=50,offy=300;
    lx=(x2-x1)/3;
    ly=(y2-y1)/3;
    x[0]=x1;
    y[0]=y1;
    x[4]=x2;
    y[4]=y2;
    x[1]=x[0]+lx;
    y[1]=y[0]+ly;
    x[3]=x[0]+2*lx;
    y[3]=y[0]+2*ly;
```

```
    xx=x[3]-x[1];
    yy=y[3]-y[1];
    x[2]=xx*(0.5)+yy*(SIN);
    y[2]=-xx*(SIN)+yy*(0.5);
    x[2]=x[2]+x[1];
    y[2]=y[2]+y[1];
if(m>0)
{
    koch(x[0],y[0],x[1],y[1],m-1);
    koch(x[1],y[1],x[2],y[2],m-1);
    koch(x[2],y[2],x[3],y[3],m-1);
    koch(x[3],y[3],x[4],y[4],m-1);
}
else
{
    line(offx+x[0],offy+y[0],offx+x[1],offy+y[1]);
    line(offx+x[1],offy+y[1],offx+x[2],offy+y[2]);
    line(offx+x[2],offy+y[2],offx+x[3],offy+y[3]);
    line(offx+x[3],offy+y[3],offx+x[4],offy+y[4]);
}
}
int main()
{
    int n,gd=DETECT, gm;
    int x1=0,x2=550,y1=0,y2=0;
    printf("\n Enter the level of curve generation:");
    scanf("%d",&n);
    //detectgraph(&gd,&gm);
    initgraph(&gd, &gm, (char*)"");
    koch(x1,y1,x2,y2,n);
//while(!kbhit);
//return 0;

    getch();
    //closegraph();
    return 0;
}
```


Windows BGI
PS C:\Users\Manasia\OneDrive\Arish\K. J. Somaiya Institute of Engineer
raphics Progam> ./Exp12