PRACTICAL 5

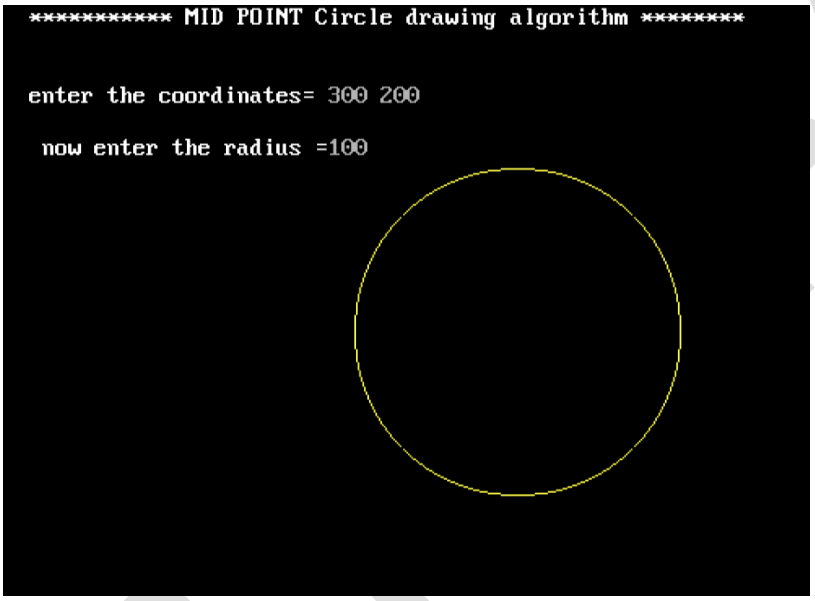
a. Develop the program for the mid-point circle drawing algorithm.

CODE:

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
void main()
{
int x,y,x_mid,y_mid,radius,dp;
int g_mode,g_driver=DETECT;
clrscr();
initgraph(&g_driver,&g_mode,"C:\\TURBOC3\\BGI");
printf("********** MID POINT Circle drawing algorithm ********\n\n");
printf("\nenter the coordinates= ");
scanf("%d %d",&x_mid,&y_mid);
printf("\n now enter the radius =");
scanf("%d",&radius);
x=0;
y=radius;
dp=1-radius;
do
{
putpixel(x_mid+x,y_mid+y,YELLOW);
putpixel(x_mid+y,y_mid+x,YELLOW);

putpixel(x_mid-y,y_mid+x,YELLOW);
putpixel(x_mid-x,y_mid+y,YELLOW);

putpixel(x_mid-x,y_mid-y,YELLOW);
putpixel(x_mid-y,y_mid-x,YELLOW);

putpixel(x_mid+y,y_mid-x,YELLOW);
putpixel(x_mid+x,y_mid-y,YELLOW);
if(dp<0) {
```

```
dp+=(2*x)+1;
}
else{
y=y-1;
dp+=(2*x)-(2*y)+1;
}
x=x+1;
}while(y>x);
getch();
}
```

OUTPUT:



PRACTICAL 7

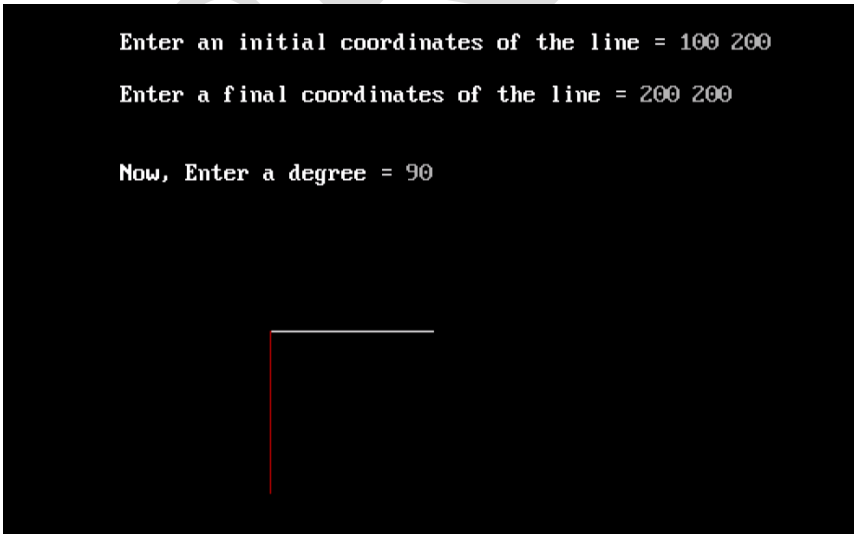
a. Perform 2D Rotation on a given object.

CODE:

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

```
int gd=DETECT,gm;
int x1,y1,x2,y2;
double degree,radian;
int rx,ry;
initgraph(&gd,&gm,"C://TURBOC3//BGI");
printf("\n Enter an initial coordinates of the line = ");
scanf("%d %d",&x1,&y1);
printf("\n Enter a final coordinates of the line = ");
scanf("%d %d",&x2,&y2);
line(x1,y1,x2,y2);
printf("\n\n Now, Enter a degree = ");
scanf("%lf",&degree);
radian=degree*0.01745;
rx=(int)(x1 +((x2-x1)*cos(radian)-(y2-y1)*sin(radian)));
ry=(int)(y1 +((x2-x1)*sin(radian)+(y2-y1)*cos(radian)));
setcolor(RED);
line(x1,y1,rx,ry);
getch();
closegraph();
}
```

OUTPUT:



b. Program to create a house like figure and perform the following operations.

i.Scaling about the origin followed by translation.
ii. Scaling with reference to an arbitrary point.
iii. Reflect about the line y = mx + c.

CODE:

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

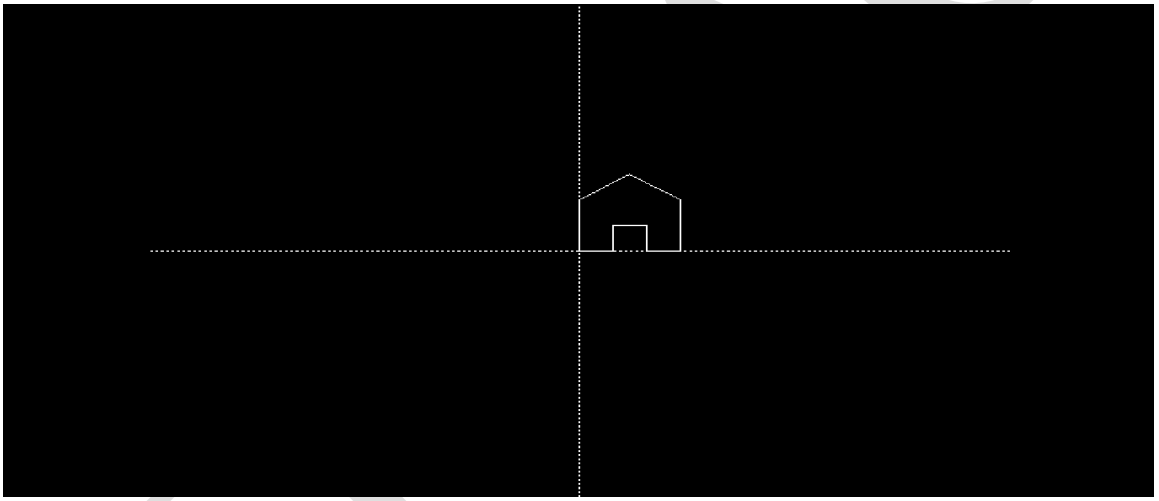
```
 for (i=0; i<9; i++)
 {
int xnew, ynew;
xnew = h[i][0] * cos (angle) - h[i][1] * sin (angle);
ynew = h[i][0] * sin (angle) + h[i][1] * cos (angle);
h[i][0] = xnew; h[i][1] = ynew;
 }
}
void scale (int h[][2], int sx, int sy)
{
 int i;
 for (i=0; i<9; i++)
 {
h[i][0] *= sx;
h[i][1] *= sy;
 }
}
void 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;
}
```

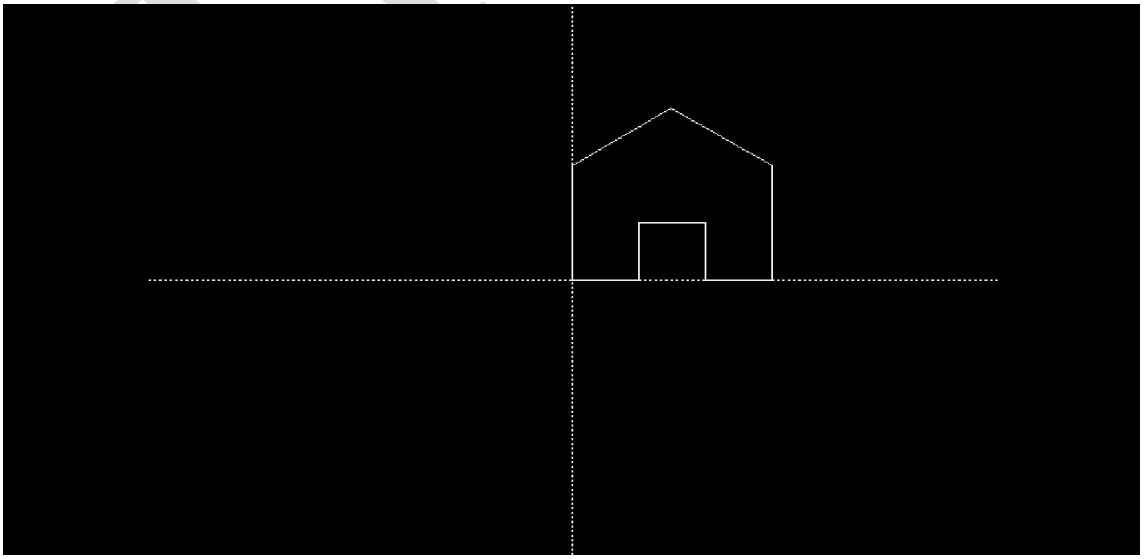
```
void ini()
{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TC\\bgi");
}
void dini()
{
getch();
closegraph();
}
void main()
{
int h[9][2],sx,sy,x,y,m,c,choice;
do
{
clrscr();
printf("\n1. Scaling about the origin.\n");
printf("2. Scaling about an arbitrary point.\n");
printf("3. Reflection about the line y = mx + c.\n");
printf("4. Exit\n");
printf("Enter the choice: ");
scanf("%d",&choice);
switch(choice)
{
case 1: printf ("\nEnter sx and sy: ");
scanf ("%d%d", &sx, &sy);
ini();
reset (h);
draw (h);getch();
scale (h, sx, sy);
cleardevice();
draw (h);
dini();
break;
case 2: printf ("Enter sx and sy: ");
scanf ("%d%d", &sx, &sy);
printf ("Enter the co-ordinates of point(x,y): ");
scanf ("%d%d", &x, &y);
ini();
reset (h);
translate (h, x, y);// Go to arbitrary point
```

```
draw(h); getch();//Show its arbitrary position
cleardevice();
translate(h,-x,-y);//Take it back to origin
draw(h);
getch();
cleardevice();
scale (h, sx, sy);//Now Scale it
draw(h);
getch();
translate (h, x, y);//Back to Arbitrary point
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);
}
```

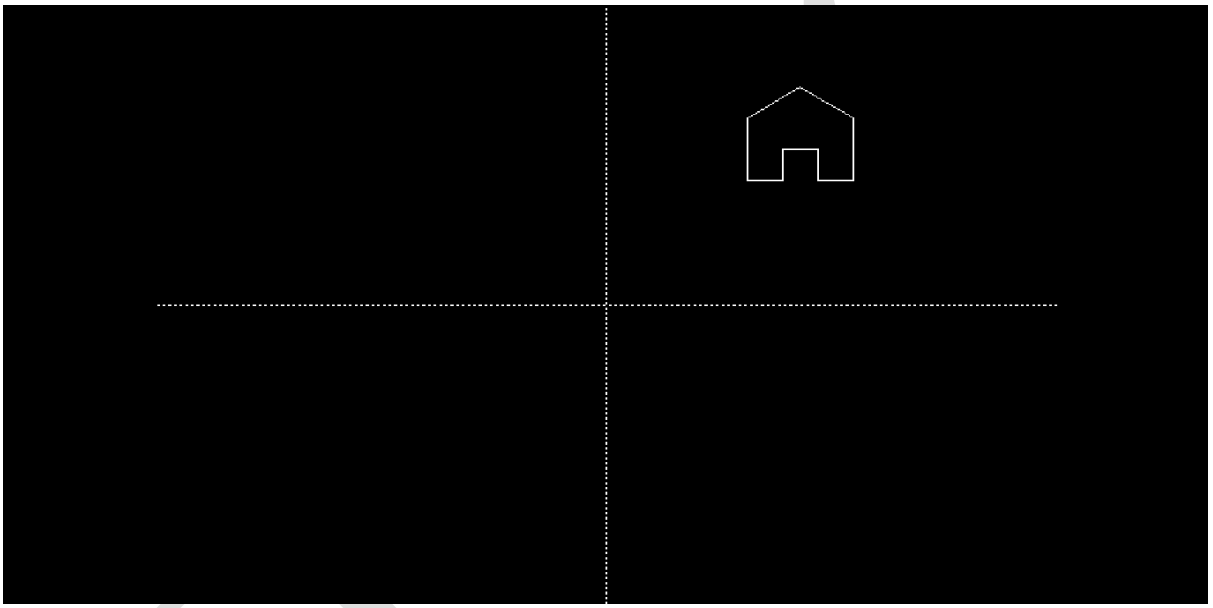
OUTPUT:

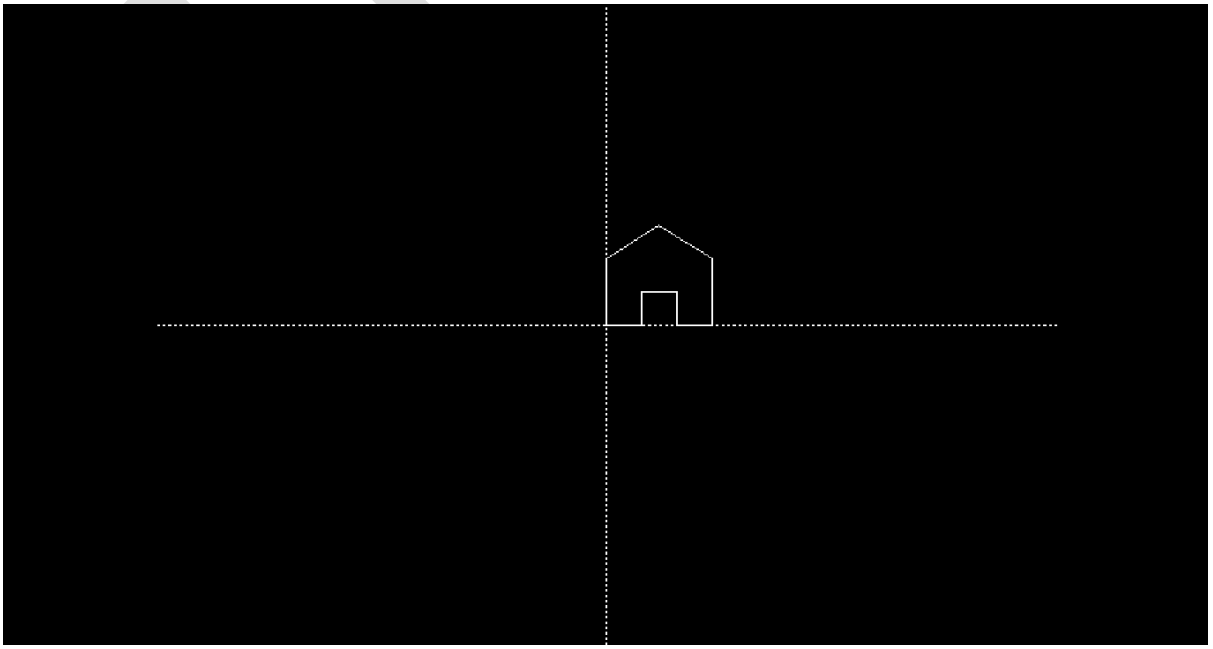
```
1. Scaling about the origin.
2. Scaling about an arbitrary point.
3. Reflection about the line y = mx + c.
4. Exit
Enter the choice: 1

Enter sx and sy: 2 2
```

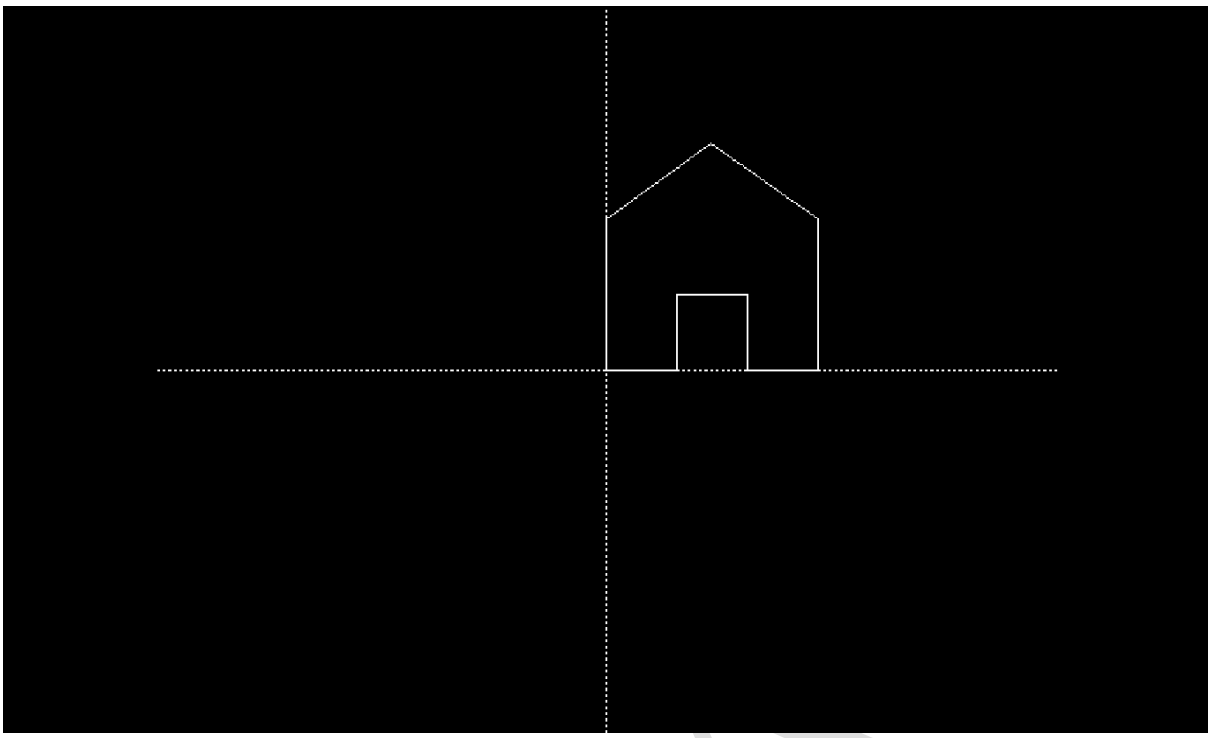


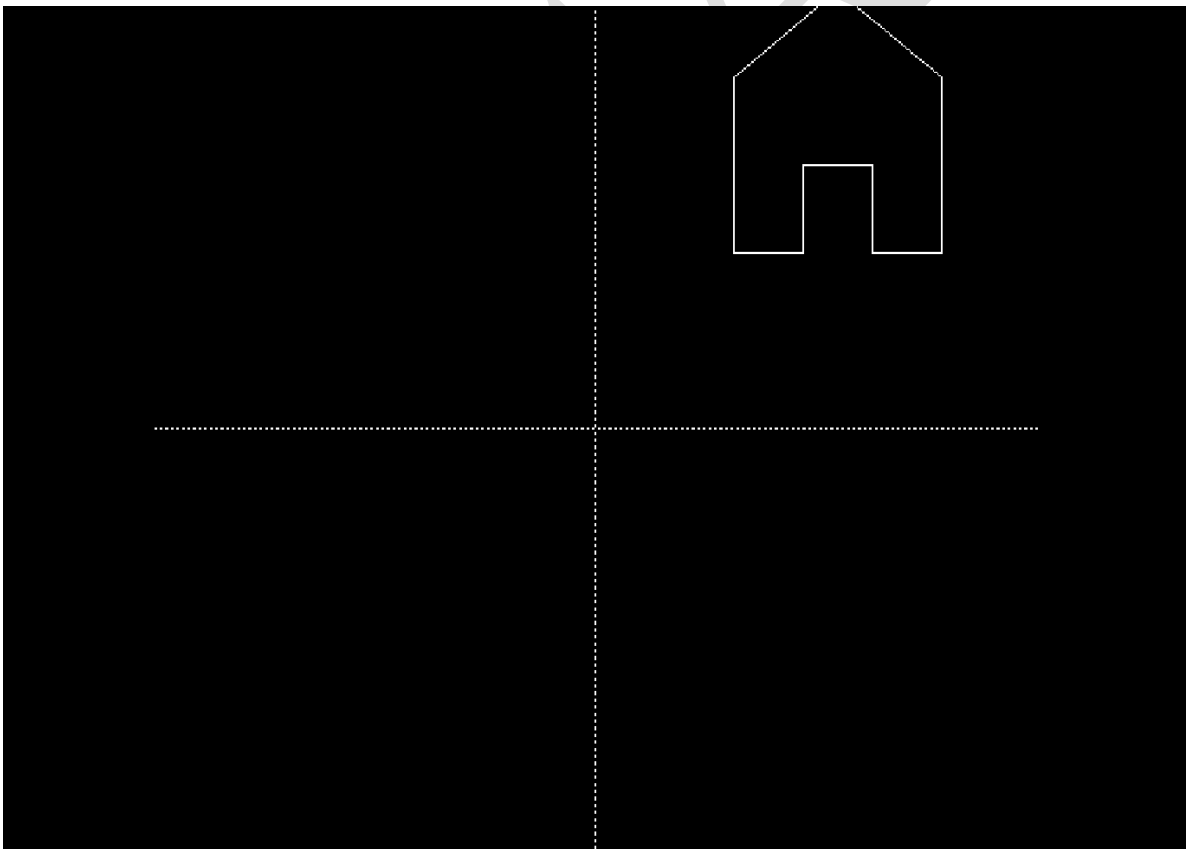


```
1. Scaling about the origin.
2. Scaling about an arbitrary point.
3. Reflection about the line y = mx + c.
4. Exit
Enter the choice: 2
Enter sx and sy: 2 2
Enter the co-ordinates of point(x,y): 100 100_
```









```
1. Scaling about the origin.
2. Scaling about an arbitrary point.
3. Reflection about the line y = mx + c.
4. Exit
Enter the choice: 3
Enter the values of m and c: 46 50
```