```
// Aditya Agre TYCOA6
//  Transformation Matrix


#include <graphics.h>
void
cw (float mat[][2], int angle)
{
  double c = cos (angle * 3.14 / 180), s = sin (angle * 3.14 / 180);
  double t[2][2] = { c, (-1) * s, s, c };
  float res[2][2] = { 0 };
  for (int i = 0; i < 2; ++i)
    for (int j = 0; j < 2; ++j)
      for (int k = 0; k < 2; ++k)
	res[i][j] += mat[i][k] * t[k][j];
  rectangle (res[0][0], res[0][1], res[1][0], res[1][1]);
}
void
acw (float mat[][2], int angle)
{
  double c = cos (angle * 3.14 / 180), s = sin (angle * 3.14 / 180);
  double t[2][2] = { c, s, (-1) * s, c };
  float res[2][2] = { 0 };
  for (int i = 0; i < 2; ++i)
    for (int j = 0; j < 2; ++j)
      for (int k = 0; k < 2; ++k)
	res[i][j] += mat[i][k] * t[k][j];
  rectangle (res[0][0], res[0][1], res[1][0], res[1][1]);
}


int
```

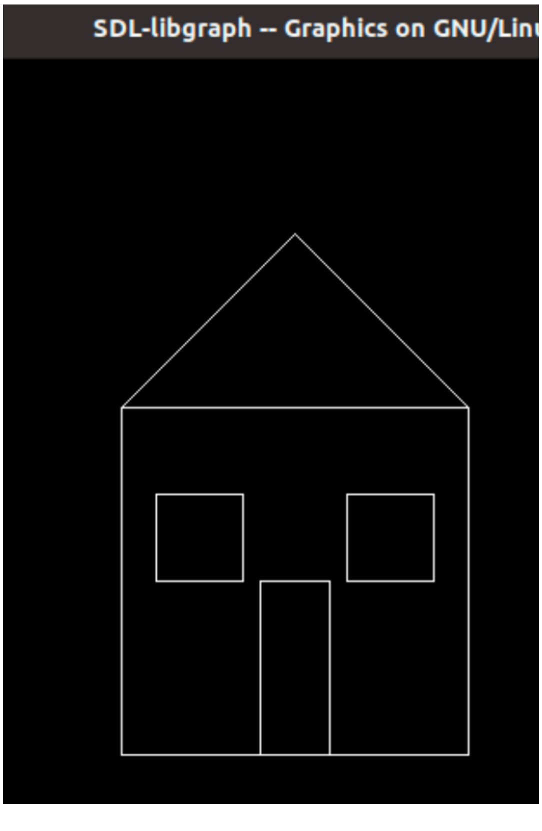
```
main ()
{
  int gd = DETECT, gm;
  initgraph (&gd, &gm, NULL);
  rectangle (200, 200, 400, 400);
  line (200, 200, 300, 100);
  line (300, 100, 400, 200);
  rectangle (280, 300, 320, 400);
  rectangle (220, 250, 270, 300);
  rectangle (330, 250, 380, 300);
  delay (2000);
  cleardevice ();
//translation int x=50;
  int y = 50;
  rectangle (200 + x, 200 + y, 400 + x, 400 + y);
  line (200 + x, 200 + y, 300 + x, 100 + y);
  line (300 + x, 100 + y, 400 + x, 200 + y);
  rectangle (280 + x, 300 + y, 320 + x, 400 + y);
  rectangle (220 + x, 250 + y, 270 + x, 300 + y);
  rectangle (330 + x, 250 + y, 380 + x, 300 + y);
  delay (2000);
  cleardevice ();
//scaling
  float a = 1.1;
  float b = 1.3;
  rectangle (200 * a, 200 * b, 400 * a, 400 * b);
  line (200 * a, 200 * b, 300 * a, 100 * b);
  line (300 * a, 100 * b, 400 * a, 200 * b);
  rectangle (280 * a, 300 * b, 320 * a, 400 * b);
  rectangle (220 * a, 250 * b, 270 * a, 300 * b);
```

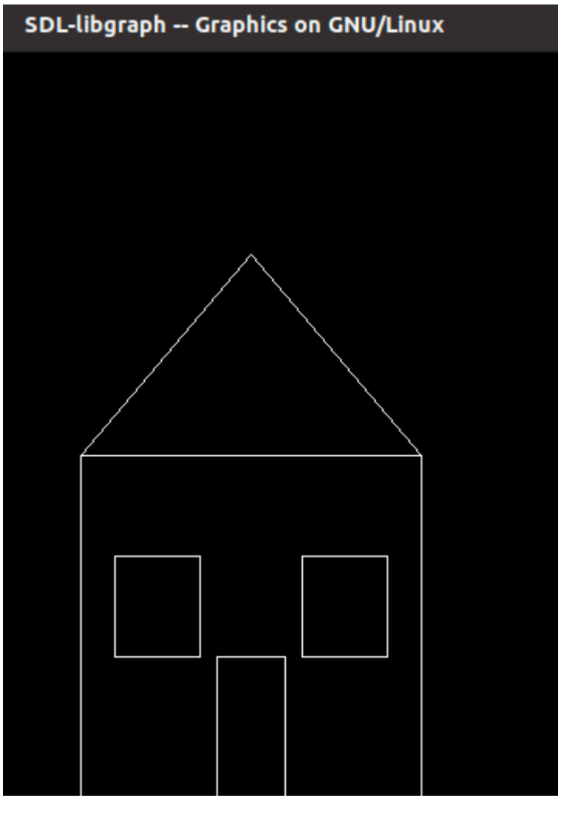
```
 rectangle (330 * a, 250 * b, 380 * a, 300 * b);
 delay (2000);
 cleardevice ();
//Rotation //clockwise
 int angle = 15;
 cw (rectangle, angle);
 cw (line, angle);
 cw (line, angle);
 delay (2000);
 cleardevice ();
//anticlockwise acw(rectangle, angle); acw(line, angle); acw(line, angle); delay(2000); cleardevice();
 getch ();
 closegraph ();
 return 0;
}
```

Output:

1. Diagram:

2. Translation:

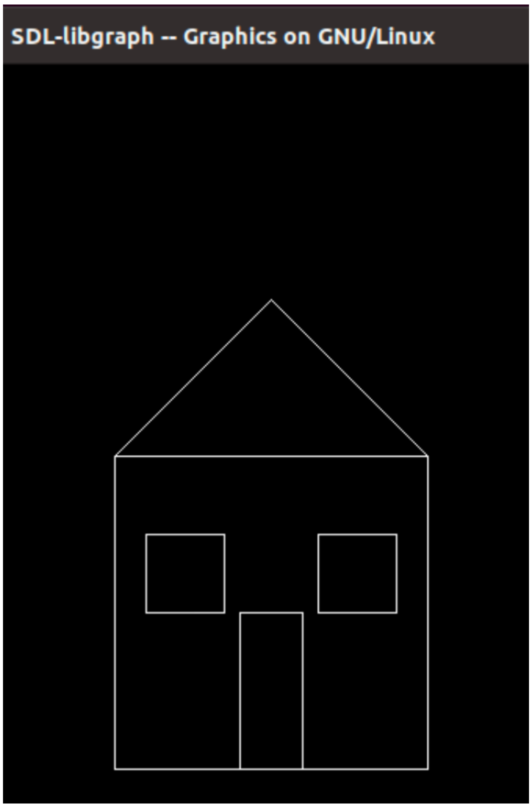


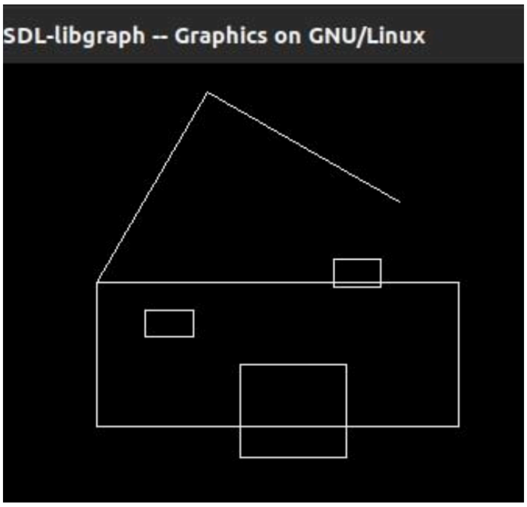


3. Scaling:

4. Clockwise Rotation:

5. Anti-clockwise rotation:


SDL-libgraph -- Graphics on GNU/Linux


SDL-libgraph -- Graphics on GNU/Linux