1) Compiling and running draw.cpp and draw_main.cpp was completed successfully. Below are the screen shots of the compiling and testing of the program.

Illustration 1: compiling draw.cpp
Illustration 3: Running draw.cpp
2) Compiling and running mouse_key.cpp and mouse_key_main.cpp was completed successfully. Below are the screen shots of the compiling and testing of the program.

```
[izbib@jb359-0 lab1]$ vi Makefile
[izbib@jb359-0 lab1]$ make
[izbib@jb359-0 lab1]$ g++ -c mouse_key_main.cpp -I/usr/X11R/include -I/share/mesa/include
[izbib@jb359-0 lab1]$ g++ -c mouse_key.cpp -I/usr/X11R/include -I/share/mesa/include
[izbib@jb359-0 lab1]$ g++ -o mouse_key mouse_key_main.o mouse_key.o -L/usr/X11/lib -L/usr/X11R6/lib -lglut -lGLU -lGL -lX11 -lXext -lXmu -lXext -lXmu -lXt -lXi -lSM -lICE
```

Illustration 4: Compiling mouse_key.cpp

Illustration 5: Running mouse_key.cpp
2) This program draws a point at the position of every mouse click and draws a line connecting the points. If 'o' is pressed this feature is toggled on and off. The program halts if 'esc' is pressed.

The program was completed successfully and below are the snapshots of the compiling and testing followed by the source code.

Illustration 6: Compiling draw_lines.cpp
//draw_lines.cpp
#include <GL/glut.h>
#include <stdlib.h>
#define screenHeight 500
#include <iostream>
using namespace std;

bool connect_dots = true;
GLint x1 = -1, y1 = -1;

//initialization
void init(void) {
    glClearColor(1.0, 1.0, 1.0, 0.0); //get white background color
    glColor3f(0.0f, 0.0f, 0.0f); //set drawing color
glPointSize(4.0); // a dot is 4x4
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0.0, 500.0, 0.0, 500.0);
} // init

void display() {
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
  glFlush();
}

void drawLine(int x1, int y1, int x2, int y2) {
  glBegin(GL_LINE);
  glVertex2i(x1, y1); // first point
  glVertex2i(x2, y2); // second point
  glEnd();
} // drawLine

void drawDot(int x, int y) {
  glBegin(GL_POINTS);
  glVertex2i(x, y); // draw a point
  glEnd();
} // drawDot

void myMouse(int button, int state, int mouseX, int mouseY) {
  if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN) {
    GLint x = mouseX;
    GLint y = screenHeight - mouseY;
    drawDot(x, y);
    if (connect_dots && x1 > -1 && y1 > -1) { // Feature is on and first point already set
      drawLine(x1, y1, x, y);
    }
    x1 = x;
    y1 = y;
    glFlush(); // send all output to screen
  }
}

void myKeyboard(unsigned char key, int x, int y) {
  switch (key) {
  case 'o':
  }
}
connect_dots = !connect_dots; //toggles feature
break;
case 27:
exit(-1);
default:
break;
}
}

//draw_lines_main.cpp: main loop of draw_lines program

#include <GL/glut.h>
#include <math.h>
#include <stdlib.h>
#include <stdio.h>

//initialization
#include <GL/glut.h>
#include <math.h>
#include <stdlib.h>
#include <stdio.h>

//initialization
void init(void);
void myMouse( int button, int state, int x, int y);
void myKeyboard ( unsigned char key, int x, int y );
void display( void );

/*  Main Loop
*  Open window with initial window size, title bar,
*  RGBA display mode, depth buffer.
*/
int main(int argc, char** argv)
{
    glutInit(&argc, argv);  //initialize toolkit
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB );  //set display mode
    glutInitWindowSize(500, 500);  //set window size on screen
    glutInitWindowPosition( 100, 150 );  //set window position on screen
    glutCreateWindow(argv[0]);  //open screen widow
    init();
    glutMouseFunc( myMouse );
    glutKeyboardFunc( myKeyboard );
    glutDisplayFunc( display );
    glutMainLoop();  //go into perpetual loop

    return 0;
}