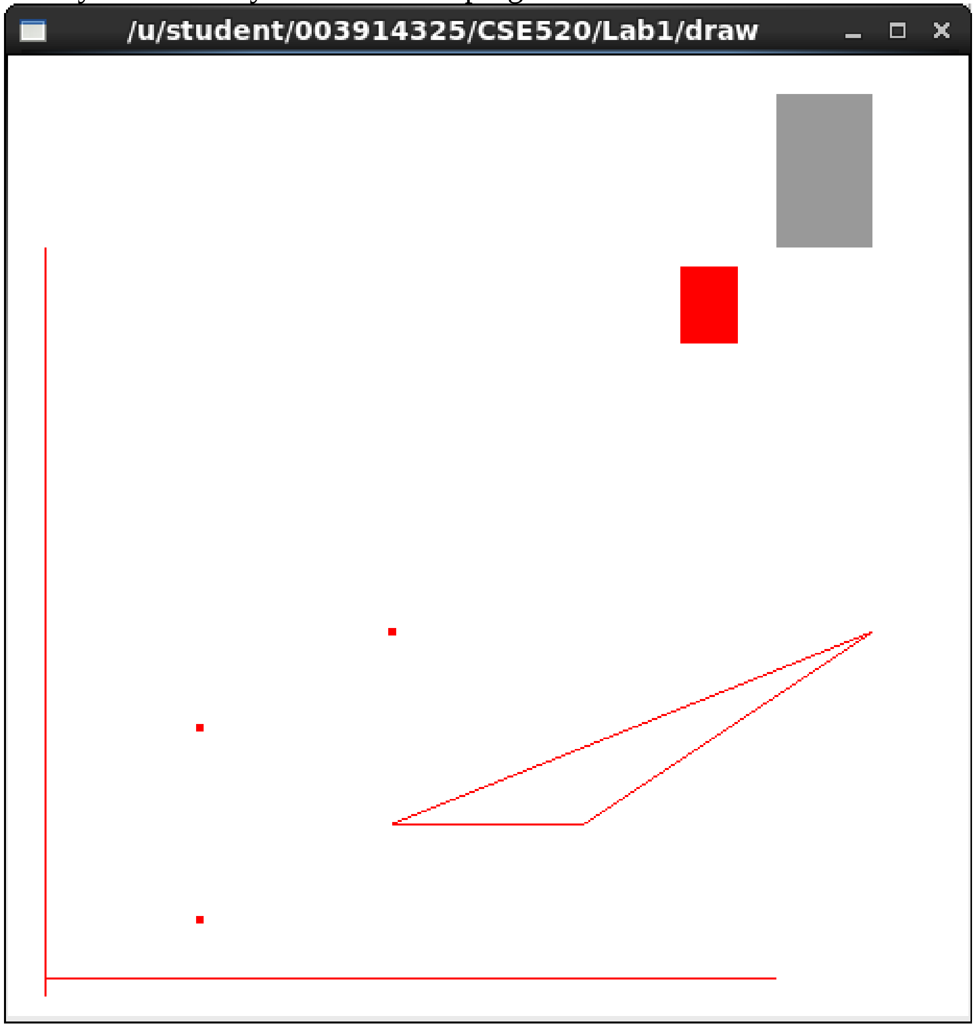
CSE520
Samuel Marrujo
Professor Yu
Lab 01

Draw01

In this part of the lab, it was a simple compilation of two programs. I was able to accomplish this task successfully. Here are my results for this program:

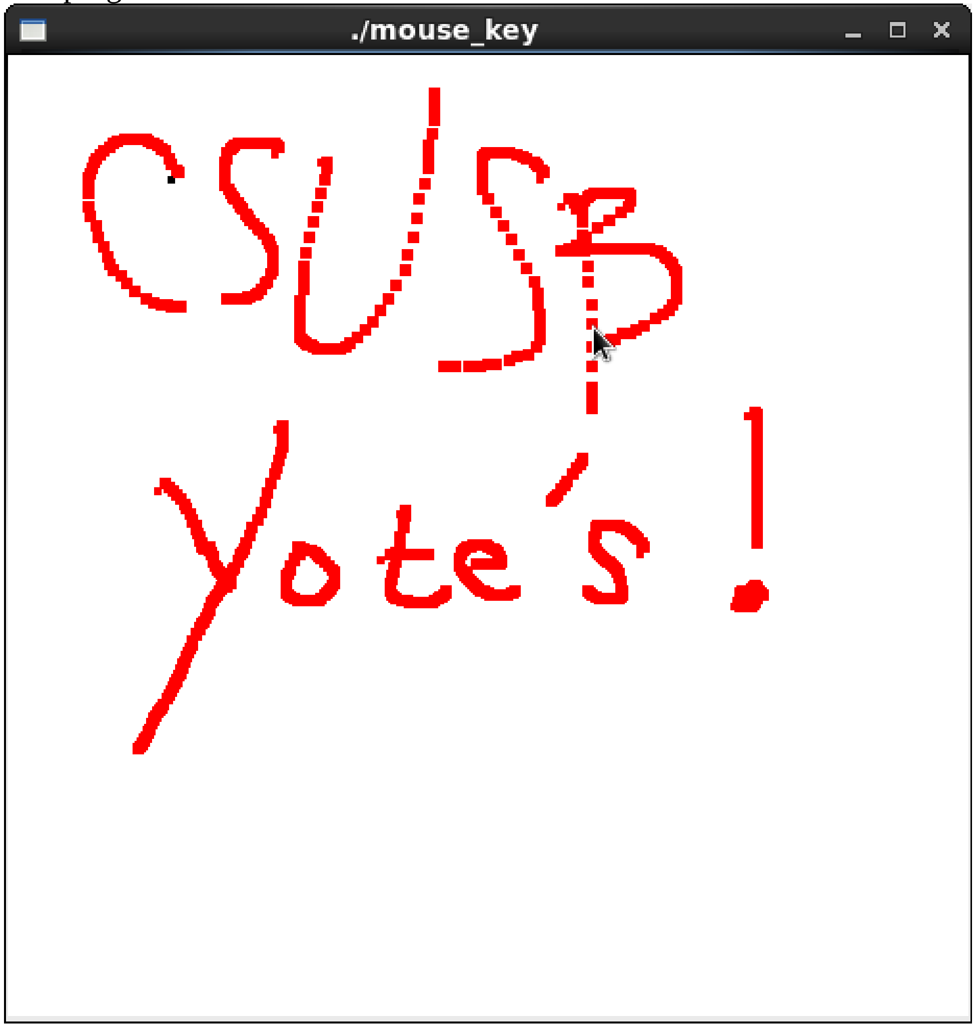


```
//draw.cpp : demo program for drawing 3 dots, two lines, ploylines, rectangles
#include <GL/glut.h>

//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();                         //replace current matrix with identity matrix
  gluOrtho2D( 0.0, 500.0, 0.0, 500.0 );
}

void display( void )
{
  glClear( GL_COLOR_BUFFER_BIT );     //clear screen
  glBegin( GL_POINTS );               //draw points
    glVertex2i( 100, 50 );            //draw a point
    glVertex2i( 100, 150 );           //draw a point
    glVertex2i( 200, 200 );           //draw a point
  glEnd();
  glBegin( GL_LINES );                //draw lines
    glVertex2i( 20, 20 );             //horizontal line
    glVertex2i( 400, 20 );
    glVertex2i( 20, 10 );             //vertical line
    glVertex2i( 20, 400 );
  glEnd();
  glBegin( GL_LINE_STRIP );           //draw polyline
    glVertex2i( 200, 100 );
    glVertex2i( 300, 100 );
    glVertex2i( 450, 200 );
    glVertex2i( 200, 100 );
  glEnd();
  glColor3f( 0.6, 0.6, 0.6 );         //bright grey
  glRecti( 400, 400, 450, 480 );
  glColor3f( 1.0, 0.0, 0.0 );         //red
  glRecti( 350, 350, 380, 390 );

  glFlush();                          //send all output to screen
}
//---------------------------------------------------------------------------

//draw_main.cpp: main loop of drawing program

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

```
#include <stdio.h>

//initialization
void init(void);
//does the drawing
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: single bufferring, RGBA
model
  glutInitWindowSize(500, 500);                  //set window size on screen
  glutInitWindowPosition( 100, 150 );            //set window position on screen
  glutCreateWindow(argv[0]);                     //open screen window
  init();
  glutDisplayFunc (display);            //points to display function
  glutMainLoop();                       //go into perpetual loop
  return 0;
}
```

Mouse_key

Lastly, in the mouse_key program, we are to modify the program to how we want it. So, I decided to change the brush size and the color, then decided to draw a happy face afterward! Here are my results for this program:

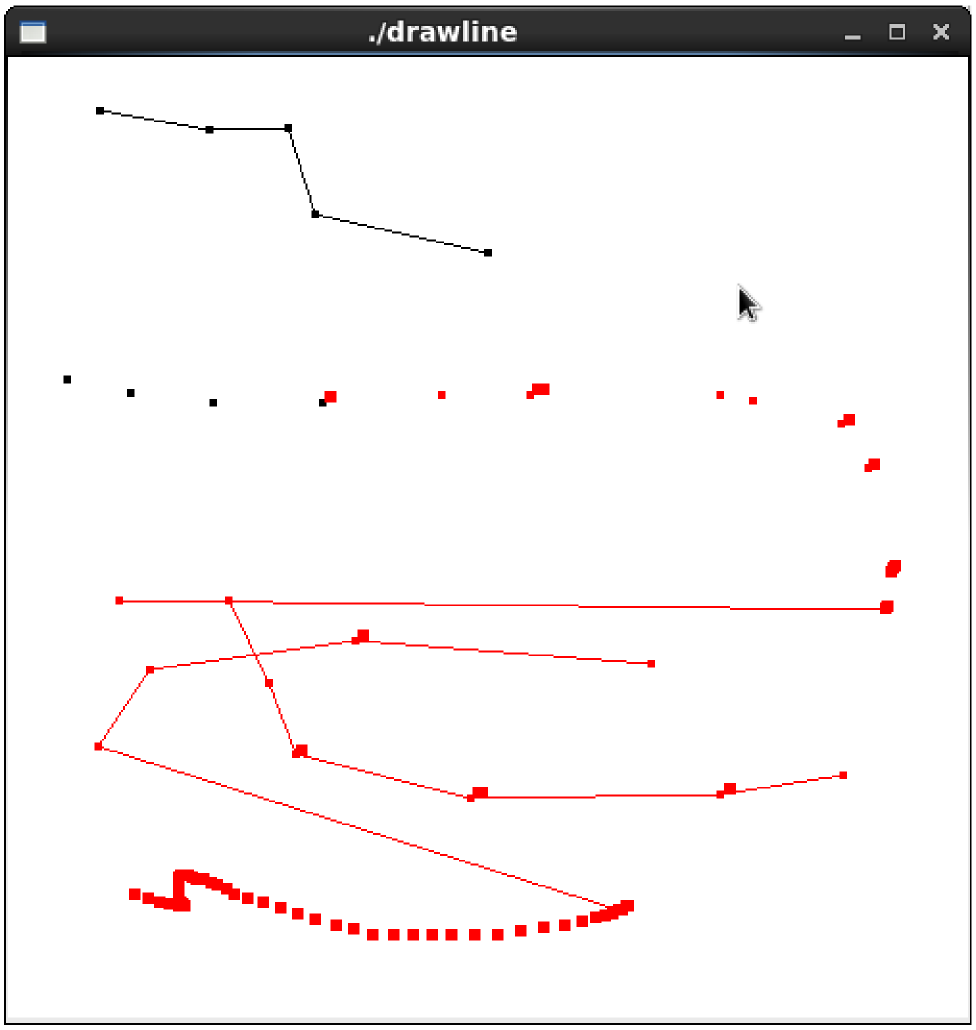


```
//mouse_key.cpp
#include <GL/glut.h>
#include <stdlib.h>

#define   screenHeight  500

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

void myMouse( int button, int state, int x, int y )
{
  if ( button == GLUT_LEFT_BUTTON && state == GLUT_DOWN )
    drawDot( x, screenHeight - y );
  glFlush();                             //send all output to screen
}

void myMovedMouse(  int mouseX, int mouseY)
{
  GLint x = mouseX;
  GLint y = screenHeight - mouseY;
  GLint brushsize = 6;
  glColor3f( 1.0, 0.0, 0.0 );
  glRecti ( x, y, x + brushsize, y + brushsize );
  glFlush();
} //myMovedMouse

void myKeyboard ( unsigned char key, int mouseX, int mouseY )
```

```
{
  GLint x = mouseX;
  GLint y = screenHeight - mouseY;
  switch( key )
  {
    case 'p':
            drawDot ( x, y );
            break;
    case 'e':
            exit ( -1 );
    default :
            break;
  }
}

//mouse_key_main.cpp: main loop of drawing program

#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 myMovedMouse(  int mouseX, int mouseY);
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 );
  glutMotionFunc( myMovedMouse );
  glutKeyboardFunc( myKeyboard );
  glutDisplayFunc( display );
  glutMainLoop();                          //go into perpetual loop
  return 0;
}
```

Draw lines

In this part of the lab, I constructed a program that creates a point from a click, and can draw a line from a left-button click. This program also can be exited by pushing the esc button. Furthermore, the 'o' button toggles whether you can draw lines, or draw points. I have accomplished all that is needed for this lab. Here are the results:



```
//mouse_key.cpp
#include <GL/glut.h>
#include <stdlib.h>

#define   screenHeight  500

bool connect = true;
GLint x_1 = -1, y_1 = -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 drawDot( int x, int y )
{
  glBegin( GL_POINTS );
    glVertex2i( x, y );               //draw a point
  glEnd();
} //drawDot

void drawLine(int x_1, int y_1, int x_2, int y_2) {
    glBegin(GL_LINES);
        glVertex2i(x_1, y_1);
        glVertex2i(x_2, y_2);
    glEnd();
}//drawLine

void myMouse( int button, int state, int msx, int msy )
{
  if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN) {
    GLint x = msx;
    GLint y = screenHeight - msy;
    drawDot(x, y);
    if (x_1 > -1 && y_1 > -1 && connect)
        drawLine(x_1,y_1,x,y);
    x_1=x;
    y_1=y;
  glFlush();                                    //send all output to screen
  }
}

void myMovedMouse(  int mouseX, int mouseY)
{
  GLint x = mouseX;
  GLint y = screenHeight - mouseY;
  GLint brushsize = 6;
  glColor3f( 1.0, 0.0, 0.0 );
  glRecti ( x, y, x + brushsize, y + brushsize );
  glFlush();
} //myMovedMouse

void myKeyboard ( unsigned char key, int mouseX, int mouseY )
{
  GLint x = mouseX;
  GLint y = screenHeight - mouseY;
  switch( key )
  {
    case 'o':
        connect = !connect; //Turn on/off switch
        break;
    case 'p':
            drawDot(x,y);
            break;
    case 27:
            exit(-1);
    default:
            break;
  }
}

//mouse_key_main.cpp: main loop of drawing program

#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 myMovedMouse(  int mouseX, int mouseY);
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 );
  glutMotionFunc( myMovedMouse );
  glutKeyboardFunc( myKeyboard );
  glutDisplayFunc( display );
  glutMainLoop();                        //go into perpetual loop
  return 0;
}
```