Alexandria University

Faculty of Engineering

Computer Graphics

Lab #1 OpenGL

Name: Toka Ashraf Abo Elwafa Ahmed

ID: 19015539

Problem Statement:

You are required to create OpenGL project using project template. You should implement an application that handles user input at runtime. Input handling should be as follows:

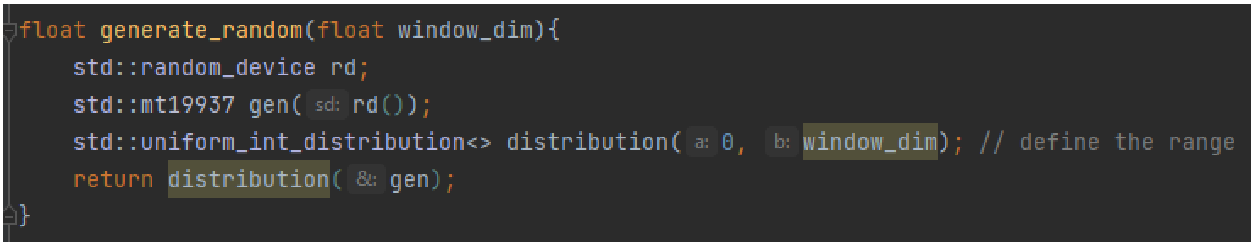
• When user presses '+' button, a new point should be drawn at random location within application window.

• When user presses '-' button, the last point drawn should be erased.

• For every two successive points, a line should be drawn connecting them. A point is allowed to be part of only one line so that number of lines is the half number of points. (Note: if number of points is odd, the last point will not be part of any line until user adds a new point).

Flow of The Code:

1- Coding the function that generate random number to use it to random the position on the screen within the range of the window(window width or window height) :



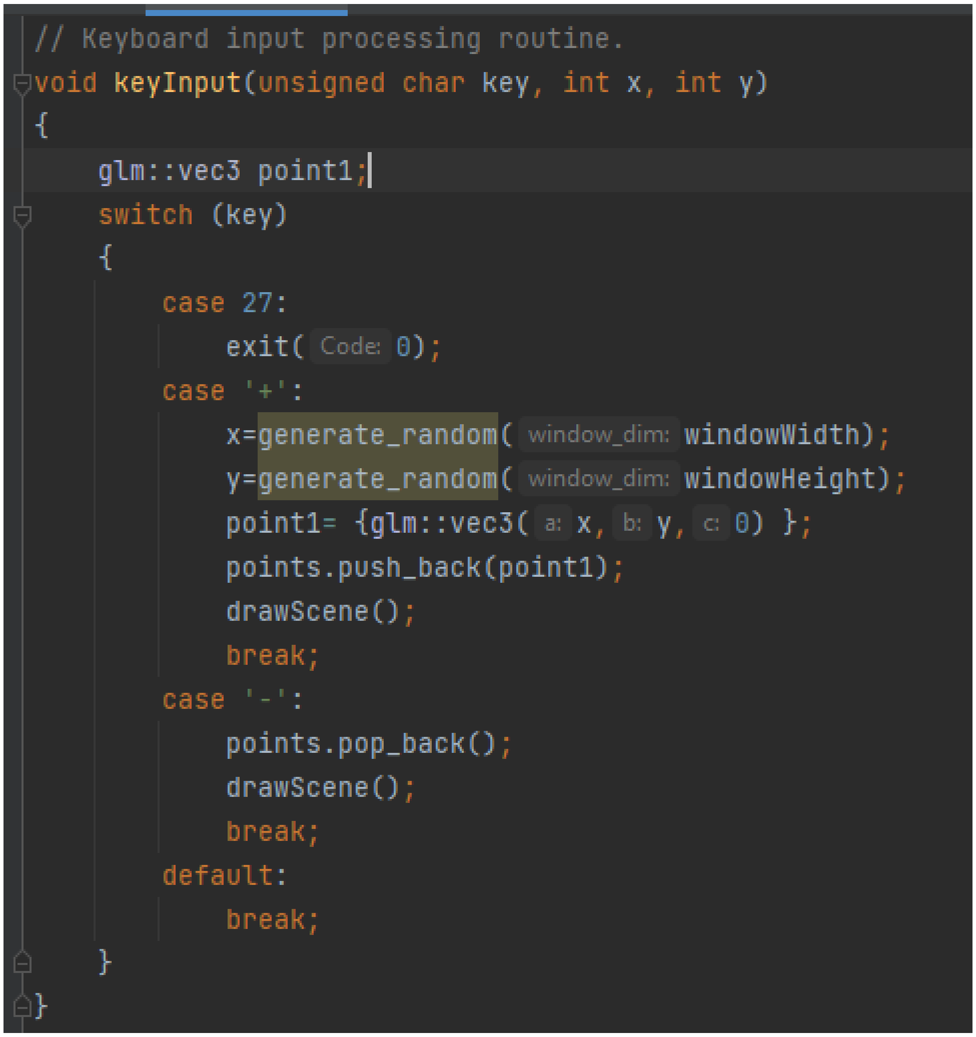
```
float generate_random(float window_dim){
    std::random_device rd;
    std::mt19937 gen( sd: rd());
    std::uniform_int_distribution<> distribution( a: 0,  b: window_dim); // define the range
    return distribution( &: gen);
}
```

2- Completing the function of (keyInput) by adding two cases:

➢ In case of pressing "+" button:

- Function that generates random number is used to randomize the point position then put this point into vec3 then inserting it at the end of the vector and finally call function of (drawScene) to draw the points on the window.

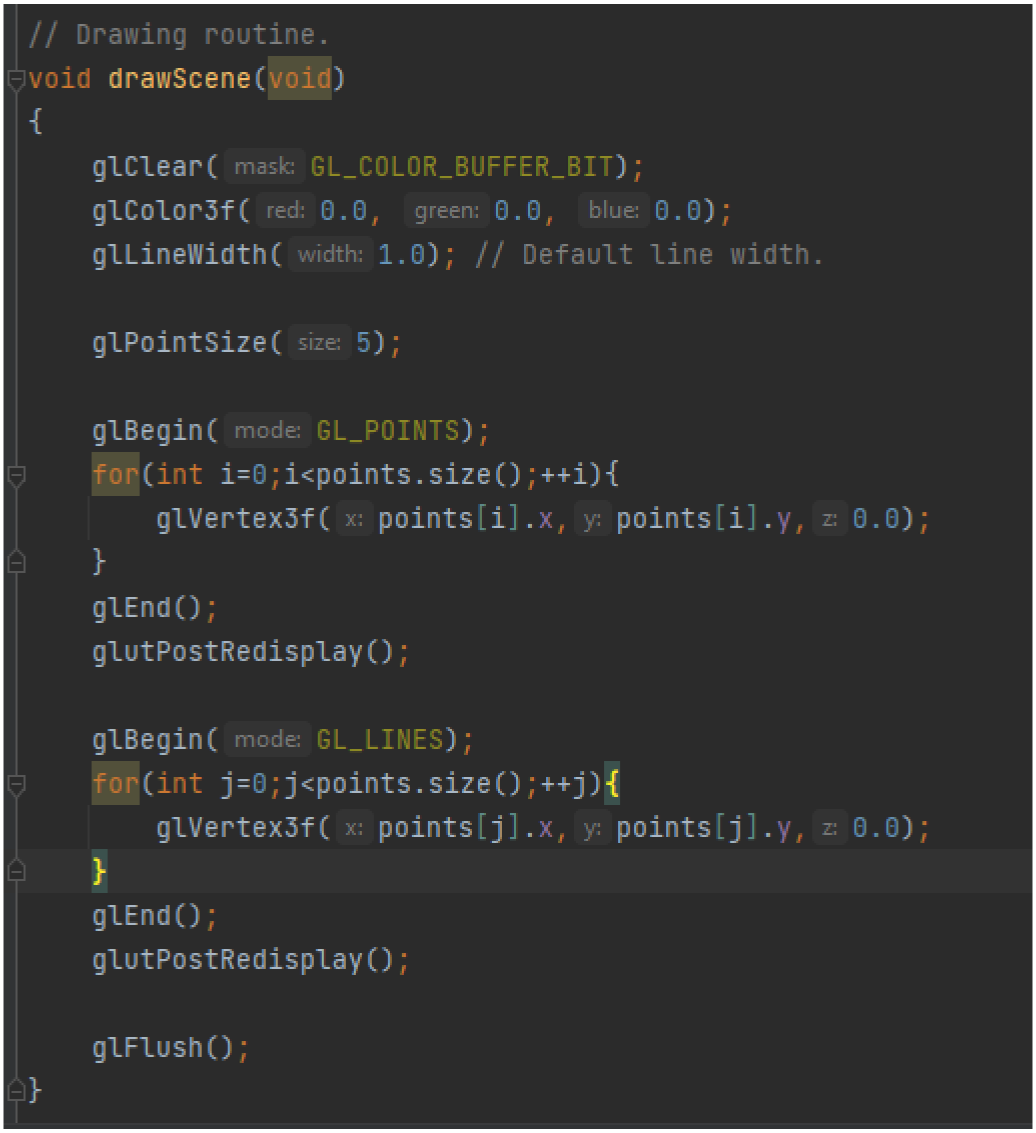
- In case of pressing “-“ button:
  - Deleting the last point added to the vector of the points then call function of (drawScene) to draw the points on the window.



```
// Keyboard input processing routine.
void keyInput(unsigned char key, int x, int y)
{
    glm::vec3 point1;
    switch (key)
    {
        case 27:
            exit( Code: 0);
        case '+':
            x=generate_random( window_dim: windowWidth);
            y=generate_random( window_dim: windowHeight);
            point1= {glm::vec3( a: x, b: y, c: 0) };
            points.push_back(point1);
            drawScene();
            break;
        case '-':
            points.pop_back();
            drawScene();
            break;
        default:
            break;
    }
}
```

3- Drawing the points from the vector after storing all points and all changes made on it:

- ➢ Drawing points by looping on the vector and draw each point using (glVertex3f) then drawing lines between the points in the same way.
- ➢ After drawing both points and lines, function of (glutPostRedisplay) must be called to make refresh to the window to make all points appear frequently.



```
// Drawing routine.
void drawScene(void)
{
    glClear( mask: GL_COLOR_BUFFER_BIT);
    glColor3f( red: 0.0,  green: 0.0,  blue: 0.0);
    glLineWidth( width: 1.0); // Default line width.

    glPointSize( size: 5);

    glBegin( mode: GL_POINTS);
    for(int i=0;i<points.size();++i){
        glVertex3f( x: points[i].x,  y: points[i].y,  z: 0.0);
    }
    glEnd();
    glutPostRedisplay();

    glBegin( mode: GL_LINES);
    for(int j=0;j<points.size();++j){
        glVertex3f( x: points[j].x,  y: points[j].y,  z: 0.0);
    }
    glEnd();
    glutPostRedisplay();

    glFlush();
}
```