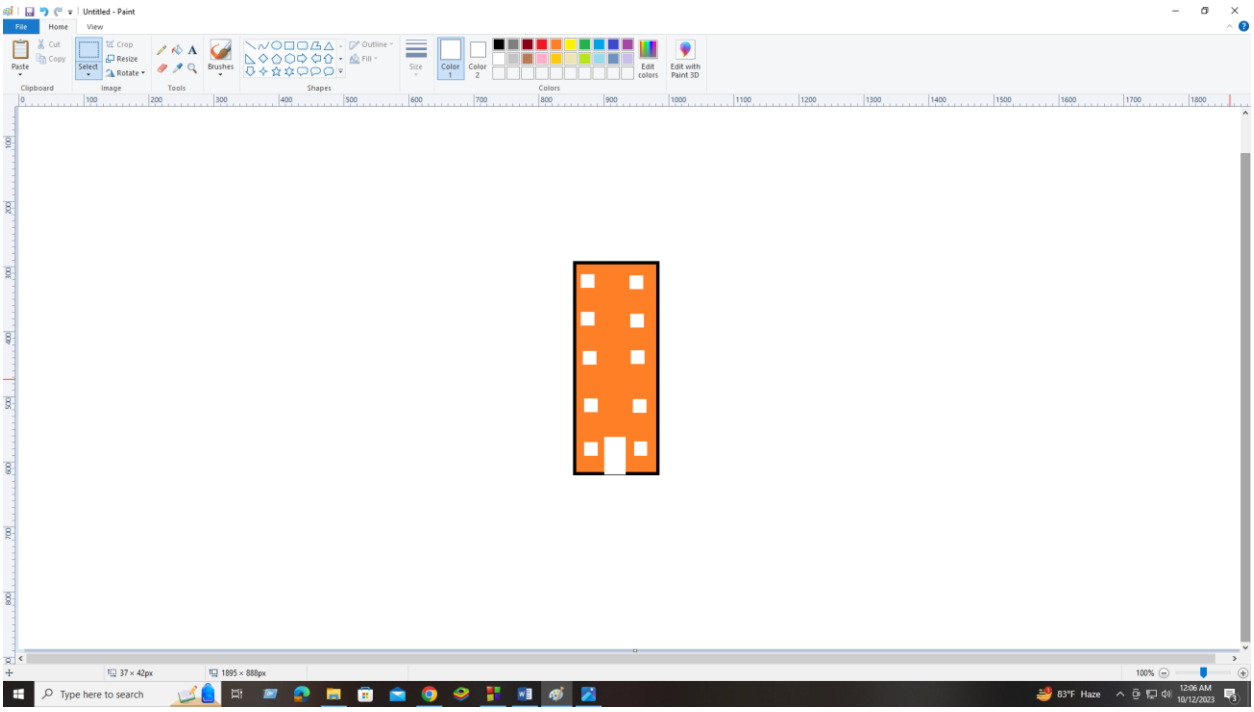
COMPUTER GRAPHICS [B]

Lab Taks-3

Submission Guidelines-17.10.2023

Question- 1
Draw five storied building with windows and a front door

Graph Plot (Picture)-



Code-

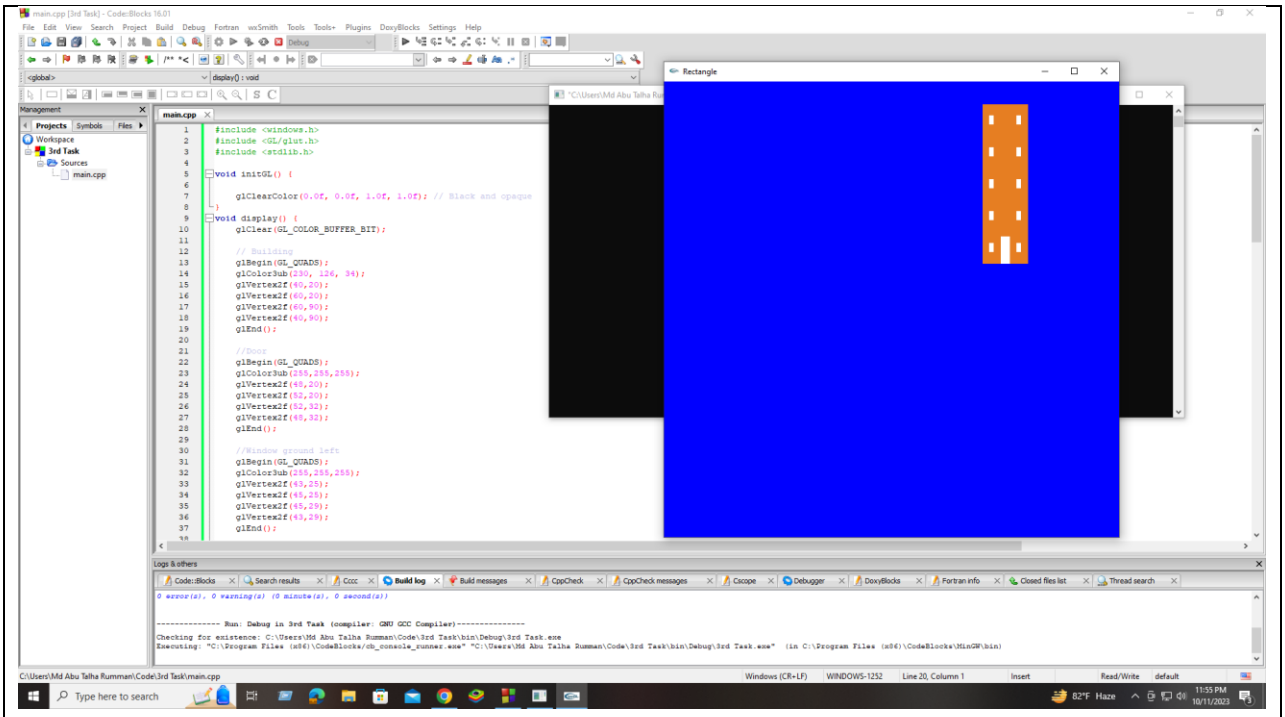
```
#include <windows.h>
#include <GL/glut.h>
#include <stdlib.h>

void initGL() {

        glClearColor(0.0f, 0.0f, 1.0f, 1.0f); // Black and opaque
}
void display() {
```

```
        glClear(GL_COLOR_BUFFER_BIT);

        // Building
        glBegin(GL_QUADS);
        glColor3ub(230, 126, 34);
        glVertex2f(40,20);
        glVertex2f(60,20);
        glVertex2f(60,90);
        glVertex2f(40,90);
        glEnd();

        //Door
        glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(48,20);
        glVertex2f(52,20);
        glVertex2f(52,32);
        glVertex2f(48,32);
        glEnd();

        //Window ground left
        glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(43,25);
        glVertex2f(45,25);
        glVertex2f(45,29);
        glVertex2f(43,29);
        glEnd();

    //Window ground right
        glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(55,25);
        glVertex2f(57,25);
        glVertex2f(57,29);
        glVertex2f(55,29);
        glEnd();

        //Window 1st left
        glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(43,39);
        glVertex2f(45,39);
        glVertex2f(45,43);
```

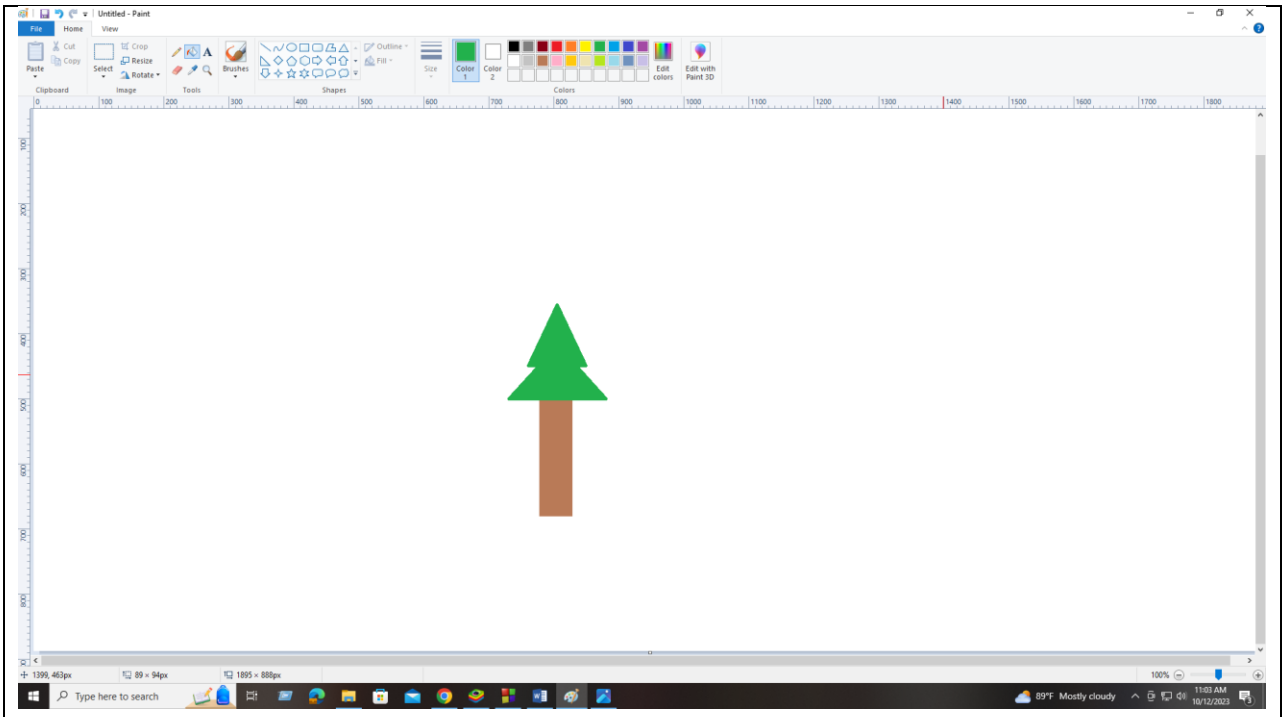
```
        glVertex2f(43,43);
        glEnd();

        //Window 1st right
    glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(55,39);
        glVertex2f(57,39);
        glVertex2f(57,43);
        glVertex2f(55,43);
        glEnd();

    //Window 2nd left
        glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(43,53);
        glVertex2f(45,53);
        glVertex2f(45,57);
        glVertex2f(43,57);
        glEnd();

    //Window 2nd right
        glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(55,53);
        glVertex2f(57,53);
        glVertex2f(57,57);
        glVertex2f(55,57);
        glEnd();

    //Window 3rd left
        glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(43,67);
        glVertex2f(45,67);
        glVertex2f(45,71);
        glVertex2f(43,71);
        glEnd();

    //Window 3rd right
        glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(55,67);
        glVertex2f(57,67);
```

```
        glVertex2f(57,71);
        glVertex2f(55,71);
        glEnd();

  //Window 4th left
        glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(43,81);
        glVertex2f(45,81);
        glVertex2f(45,85);
        glVertex2f(43,85);
        glEnd();

  //Window 4th left
        glBegin(GL_QUADS);
        glColor3ub(255,255,255);
        glVertex2f(55,81);
        glVertex2f(57,81);
        glVertex2f(57,85);
        glVertex2f(55,85);
        glEnd();

        glFlush();  // Render now
}

/* Main function: GLUT runs as a console application starting at main()  */
int main(int argc, char** argv) {
        glutInit(&argc, argv);
        glutInitWindowSize(700, 700);       // Initialize GLUT
        glutCreateWindow("Rectangle");  // Create window with the given title
        //glutInitWindowSize(320, 320);   // Set the window's initial width & height
        glutInitWindowPosition(50, 50); // Position the window's initial top-left corner
        glutDisplayFunc(display);      // Register callback handler for window re-paint event
        initGL();                  // Our own OpenGL initialization
        gluOrtho2D(-100, 100, -100, 100);
        glutMainLoop();             // Enter the event-processing loop
        return 0;
}
```

Output Screenshot (Full Screen)-



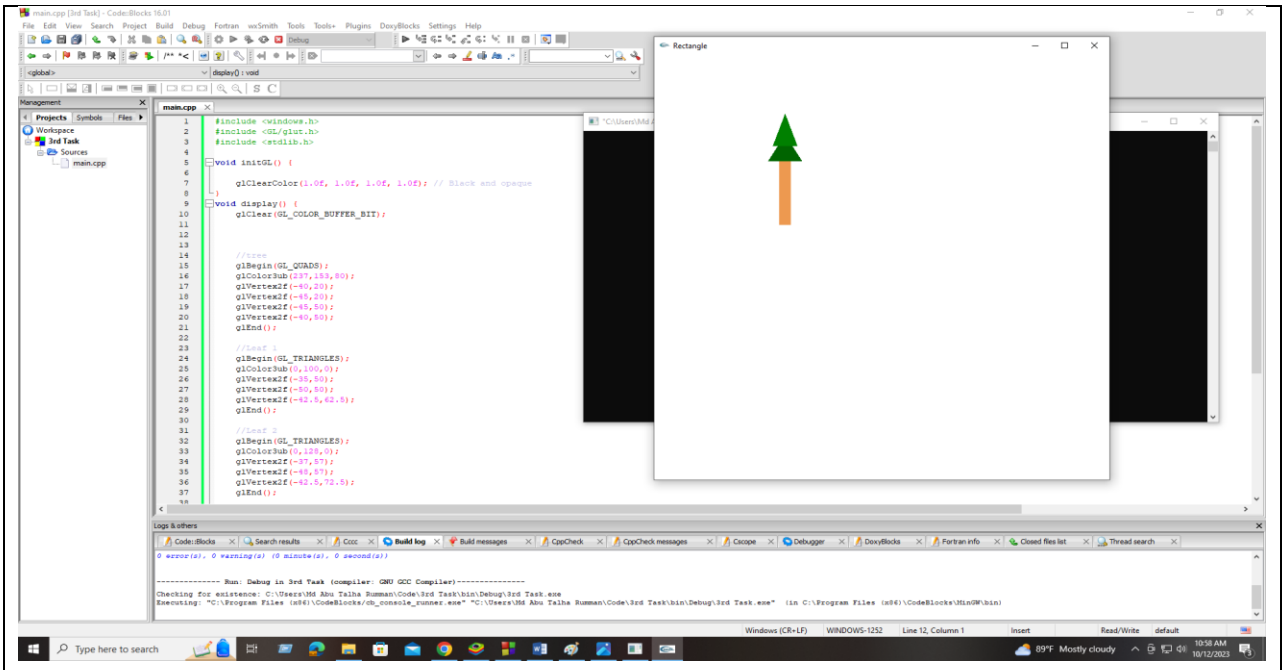
Question- 2 Draw a tree
Graph Plot (Picture)-




```
Code-
#include <windows.h>
#include <GL/glut.h>
#include <stdlib.h>

void initGL() {

        glClearColor(1.0f, 1.0f, 1.0f, 1.0f); // Black and opaque
}
void display() {
        glClear(GL_COLOR_BUFFER_BIT);

   //tree
        glBegin(GL_QUADS);
        glColor3ub(237,153,80);
        glVertex2f(-40,20);
        glVertex2f(-45,20);
        glVertex2f(-45,50);
        glVertex2f(-40,50);
        glEnd();

        //Leaf 1
   glBegin(GL_TRIANGLES);
        glColor3ub(0,100,0);
        glVertex2f(-35,50);
```

```
        glVertex2f(-50,50);
        glVertex2f(-42.5,62.5);
        glEnd();

        //Leaf 2
   glBegin(GL_TRIANGLES);
        glColor3ub(0,128,0);
        glVertex2f(-37,57);
        glVertex2f(-48,57);
        glVertex2f(-42.5,72.5);
        glEnd();

        glFlush();  // Render now
}

/* Main function: GLUT runs as a console application starting at main()  */
int main(int argc, char** argv) {
        glutInit(&argc, argv);
        glutInitWindowSize(700, 700);       // Initialize GLUT
        glutCreateWindow("Rectangle");  // Create window with the given title
        //glutInitWindowSize(320, 320);   // Set the window's initial width & height
        glutInitWindowPosition(50, 50); // Position the window's initial top-left corner
        glutDisplayFunc(display);      // Register callback handler for window re-paint event
        initGL();                  // Our own OpenGL initialization
        gluOrtho2D(-100, 100, -100, 100);
        glutMainLoop();             // Enter the event-processing loop
        return 0;
}
```

Output Screenshot (Full Screen)-

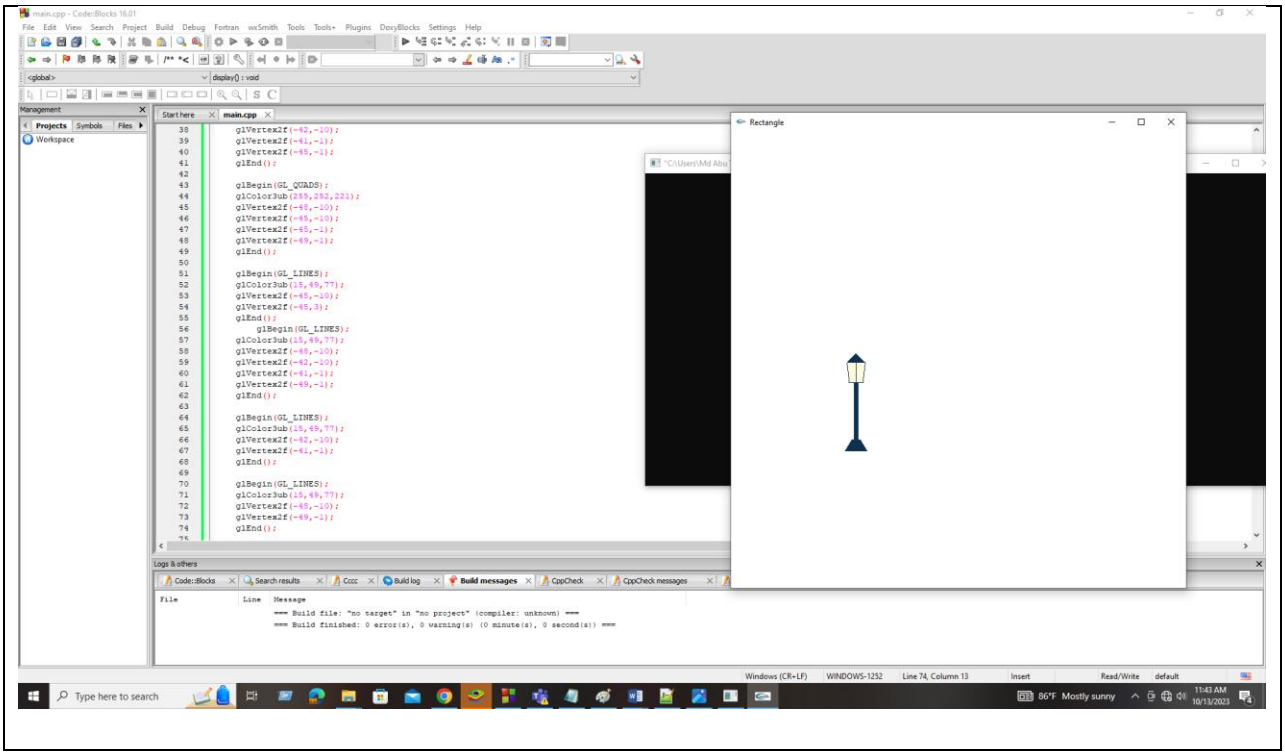


Question- 3 Draw a lamppost with black background
Graph Plot (Picture)-
Code- **#include <windows.h>** **#include <GL/glut.h>** **#include <stdlib.h>** **#include<math.h>** **void initGL() {** **glClearColor(1.0f, 1.0f, 1.0f, 1.0f); // Black and opaque** **}** **void display() {** **glClear(GL_COLOR_BUFFER_BIT);** **//lamppost** **glBegin(GL_QUADS);** **glColor3ub(15,49,77);** **glVertex2f(-50,-40);**

```
        glVertex2f(-40,-40);
        glVertex2f(-43,-35);
        glVertex2f(-47,-35);
        glEnd();

        //light
  glBegin(GL_QUADS);
        glColor3ub(15,49,77);
        glVertex2f(-46,-35);
        glVertex2f(-44,-35);
        glVertex2f(-44,-10);
        glVertex2f(-46,-10);
        glEnd();



  glBegin(GL_QUADS);
        glColor3ub(255,252,221);
        glVertex2f(-45,-10);
        glVertex2f(-42,-10);
        glVertex2f(-41,-1);
        glVertex2f(-45,-1);
        glEnd();

        glBegin(GL_QUADS);
        glColor3ub(255,252,221);
        glVertex2f(-48,-10);
        glVertex2f(-45,-10);
        glVertex2f(-45,-1);
        glVertex2f(-49,-1);
        glEnd();

        glBegin(GL_LINES);
        glColor3ub(15,49,77);
        glVertex2f(-45,-10);
        glVertex2f(-45,3);
        glEnd();
                glBegin(GL_LINES);
        glColor3ub(15,49,77);
        glVertex2f(-48,-10);
        glVertex2f(-42,-10);
        glVertex2f(-41,-1);
        glVertex2f(-49,-1);
```

```
        glEnd();

   glBegin(GL_LINES);
        glColor3ub(15,49,77);
        glVertex2f(-42,-10);
        glVertex2f(-41,-1);
        glEnd();

        glBegin(GL_LINES);
        glColor3ub(15,49,77);
        glVertex2f(-48,-10);
        glVertex2f(-49,-1);
        glEnd();

        glBegin(GL_TRIANGLES);
        glColor3ub(15,49,77);
        glVertex2f(-49,-1);
        glVertex2f(-41,-1);
        glVertex2f(-45,3);
        glEnd();
        glFlush();  // Render now


}


/* Main function: GLUT runs as a console application starting at main()  */
int main(int argc, char** argv) {
        glutInit(&argc, argv);
        glutInitWindowSize(700, 700);       // Initialize GLUT
        glutCreateWindow("Rectangle");  // Create window with the given title
        //glutInitWindowSize(320, 320);   // Set the window's initial width & height
        glutInitWindowPosition(50, 50); // Position the window's initial top-left corner
        glutDisplayFunc(display);      // Register callback handler for window re-paint event
        initGL();                  // Our own OpenGL initialization
        gluOrtho2D(-100, 100, -100, 100);
        glutMainLoop();             // Enter the event-processing loop
        return 0;
}
```

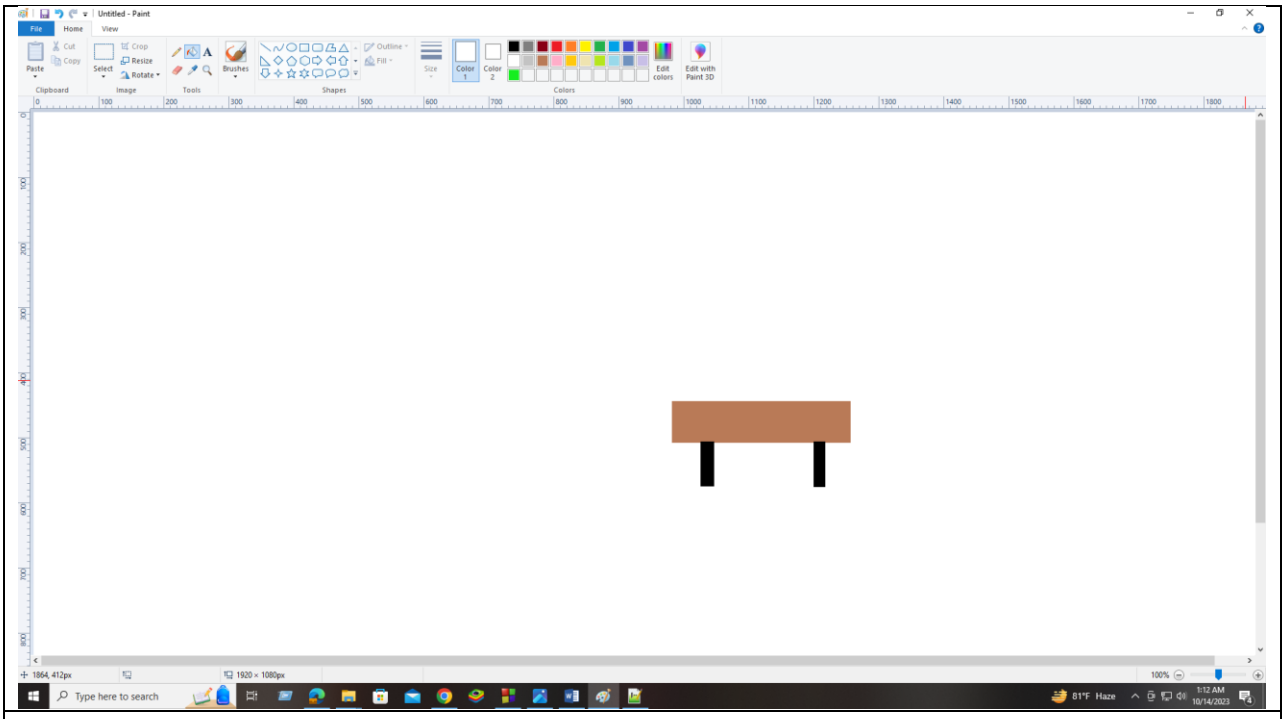
Output Screenshot (Full Screen)-



Question- 4

Draw a bench

Graph Plot (Picture)-

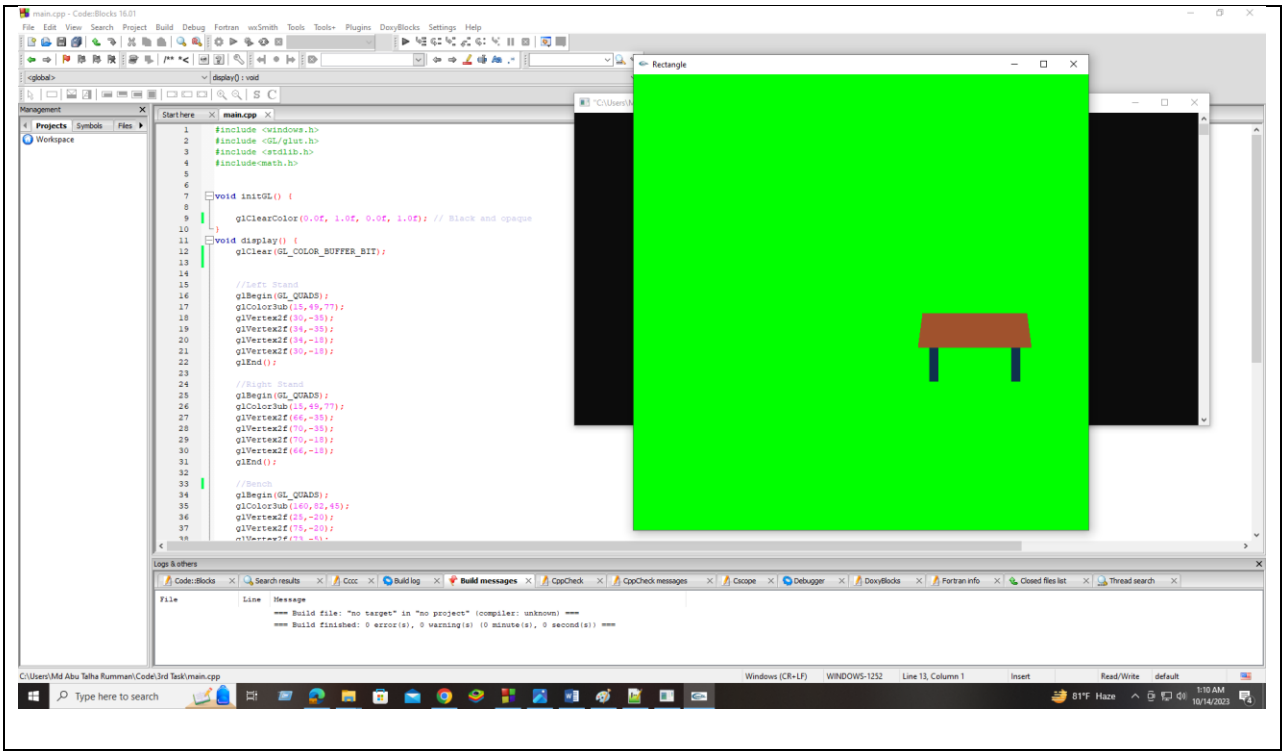


Code-

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


void initGL() {

        glClearColor(0.0f, 1.0f, 0.0f, 1.0f); // Black and opaque
}
void display() {
        glClear(GL_COLOR_BUFFER_BIT);


   //Left Stand
        glBegin(GL_QUADS);
        glColor3ub(15,49,77);
        glVertex2f(30,-35);
        glVertex2f(34,-35);
        glVertex2f(34,-18);
        glVertex2f(30,-18);
        glEnd();

        //Right Stand
        glBegin(GL_QUADS);
```

```
        glColor3ub(15,49,77);
        glVertex2f(66,-35);
        glVertex2f(70,-35);
        glVertex2f(70,-18);
        glVertex2f(66,-18);
        glEnd();

   //Bench
   glBegin(GL_QUADS);
        glColor3ub(160,82,45);
        glVertex2f(25,-20);
        glVertex2f(75,-20);
        glVertex2f(73,-5);
        glVertex2f(27,-5);
        glEnd();




        glFlush();  // Render now


}


/* Main function: GLUT runs as a console application starting at main()  */
int main(int argc, char** argv) {
        glutInit(&argc, argv);
        glutInitWindowSize(700, 700);       // Initialize GLUT
        glutCreateWindow("Rectangle");  // Create window with the given title
        //glutInitWindowSize(320, 320);   // Set the window's initial width & height
        glutInitWindowPosition(50, 50); // Position the window's initial top-left corner
        glutDisplayFunc(display);      // Register callback handler for window re-paint event
        initGL();                  // Our own OpenGL initialization
        gluOrtho2D(-100, 100, -100, 100);
        glutMainLoop();             // Enter the event-processing loop
        return 0;
}
```

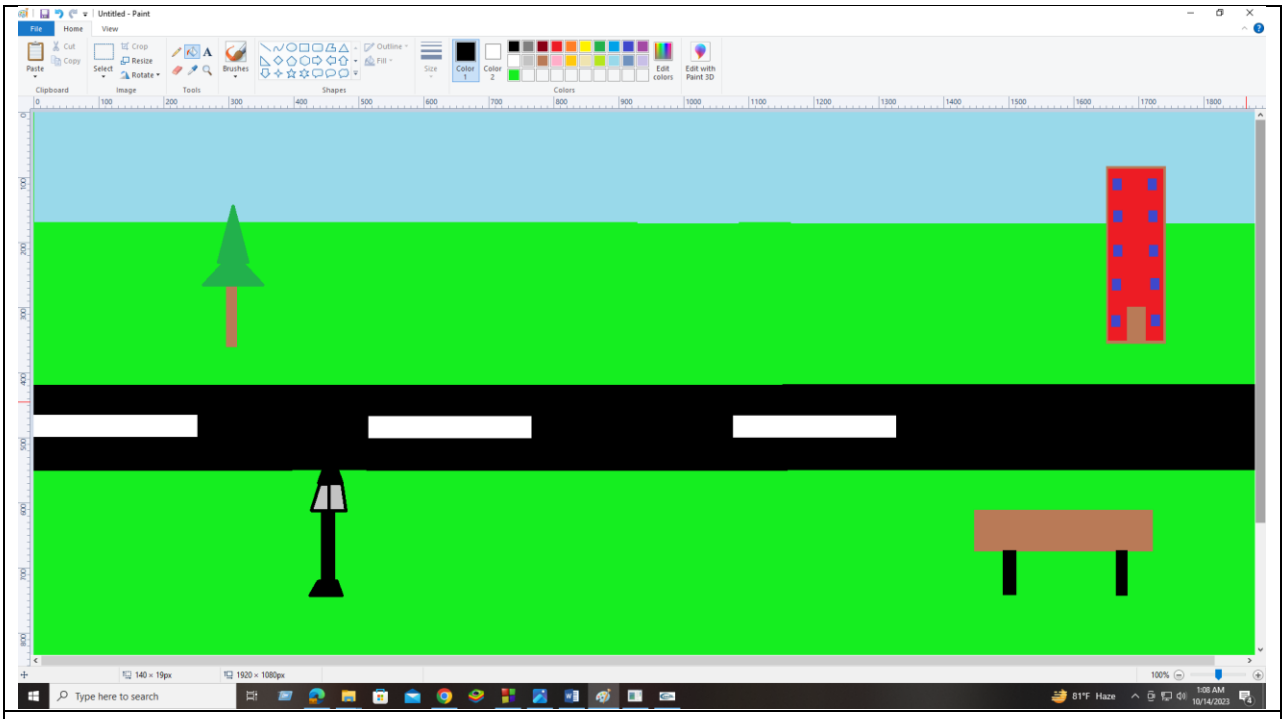
Output Screenshot (Full Screen)-



Question- 5

Use the building, tree, lamppost and bench to create a scenario

Graph Plot (Picture)-



Code-

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


void initGL() {

        glClearColor(0.0f, 1.0f, 0.0f, 1.0f); // Black and opaque
}
void display() {
        glClear(GL_COLOR_BUFFER_BIT);

   //Road
   glBegin(GL_QUADS);
        glColor3ub(29,29,31);
        glVertex2f(-100,-10);
        glVertex2f(100,-10);
        glVertex2f(100,10);
        glVertex2f(-100,10);
        glEnd();

        //Cross1
   glBegin(GL_QUADS);
        glColor3ub(255,250,250);
```

```
        glVertex2f(-100,-2);
        glVertex2f(-54,-2);
        glVertex2f(-54,2);
        glVertex2f(-100,2);
        glEnd();

        //Cross2
    glBegin(GL_QUADS);
        glColor3ub(255,250,250);
        glVertex2f(-8,-2);
        glVertex2f(38,-2);
        glVertex2f(38,2);
        glVertex2f(-8,2);
        glEnd();

        //Cross3
    glBegin(GL_QUADS);
        glColor3ub(255,250,250);
        glVertex2f(84,-2);
        glVertex2f(100,-2);
        glVertex2f(100,2);
        glVertex2f(84,2);
        glEnd();

    //Sky
    glBegin(GL_QUADS);
        glColor3ub(135,206,250);
        glVertex2f(-100,67);
        glVertex2f(100,67);
        glVertex2f(100,100);
        glVertex2f(-100,100);
        glEnd();

    // Building
        glBegin(GL_QUADS);
        glColor3ub(253, 89, 90);
        glVertex2f(40,20);
        glVertex2f(60,20);
        glVertex2f(60,90);
        glVertex2f(40,90);
        glEnd();

        //Door
        glBegin(GL_QUADS);
```

```
        glColor3ub(215,93,52);
        glVertex2f(48,20);
        glVertex2f(52,20);
        glVertex2f(52,32);
        glVertex2f(48,32);
        glEnd();

        //Window ground left
        glBegin(GL_QUADS);
        glColor3ub(80,86,134);
        glVertex2f(43,25);
        glVertex2f(45,25);
        glVertex2f(45,29);
        glVertex2f(43,29);
        glEnd();

  //Window ground right
        glBegin(GL_QUADS);
        glColor3ub(80,86,134);
        glVertex2f(55,25);
        glVertex2f(57,25);
        glVertex2f(57,29);
        glVertex2f(55,29);
        glEnd();

        //Window 1st left
        glBegin(GL_QUADS);
        glColor3ub(80,86,134);
        glVertex2f(43,39);
        glVertex2f(45,39);
        glVertex2f(45,43);
        glVertex2f(43,43);
        glEnd();

        //Window 1st right
  glBegin(GL_QUADS);
        glColor3ub(80,86,134);
        glVertex2f(55,39);
        glVertex2f(57,39);
        glVertex2f(57,43);
        glVertex2f(55,43);
        glEnd();

  //Window 2nd left
```

```
        glBegin(GL_QUADS);
        glColor3ub(80,86,134);
        glVertex2f(43,53);
        glVertex2f(45,53);
        glVertex2f(45,57);
        glVertex2f(43,57);
        glEnd();

    //Window 2nd right
        glBegin(GL_QUADS);
        glColor3ub(80,86,134);
        glVertex2f(55,53);
        glVertex2f(57,53);
        glVertex2f(57,57);
        glVertex2f(55,57);
        glEnd();

    //Window 3rd left
        glBegin(GL_QUADS);
        glColor3ub(80,86,134);
        glVertex2f(43,67);
        glVertex2f(45,67);
        glVertex2f(45,71);
        glVertex2f(43,71);
        glEnd();

    //Window 3rd right
        glBegin(GL_QUADS);
        glColor3ub(80,86,134);
        glVertex2f(55,67);
        glVertex2f(57,67);
        glVertex2f(57,71);
        glVertex2f(55,71);
        glEnd();

    //Window 4th left
        glBegin(GL_QUADS);
        glColor3ub(80,86,134);
        glVertex2f(43,81);
        glVertex2f(45,81);
        glVertex2f(45,85);
        glVertex2f(43,85);
        glEnd();
```

```
//Window 4th left
        glBegin(GL_QUADS);
        glColor3ub(80,86,134);
        glVertex2f(55,81);
        glVertex2f(57,81);
        glVertex2f(57,85);
        glVertex2f(55,85);
        glEnd();

//tree
        glBegin(GL_QUADS);
        glColor3ub(237,153,80);
        glVertex2f(-40,20);
        glVertex2f(-45,20);
        glVertex2f(-45,50);
        glVertex2f(-40,50);
        glEnd();

        //Leaf 1
glBegin(GL_TRIANGLES);
        glColor3ub(0,100,0);
        glVertex2f(-35,50);
        glVertex2f(-50,50);
        glVertex2f(-42.5,62.5);
        glEnd();

        //Leaf 2
glBegin(GL_TRIANGLES);
        glColor3ub(0,128,0);
        glVertex2f(-37,57);
        glVertex2f(-48,57);
        glVertex2f(-42.5,72.5);
        glEnd();

        //lamppost
        glBegin(GL_QUADS);
        glColor3ub(15,49,77);
        glVertex2f(-50,-40);
        glVertex2f(-40,-40);
        glVertex2f(-43,-35);
        glVertex2f(-47,-35);
        glEnd();

        //light
```

```
glBegin(GL_QUADS);
        glColor3ub(15,49,77);
        glVertex2f(-46,-35);
        glVertex2f(-44,-35);
        glVertex2f(-44,-10);
        glVertex2f(-46,-10);
        glEnd();

glBegin(GL_QUADS);
        glColor3ub(255,252,221);
        glVertex2f(-45,-10);
        glVertex2f(-42,-10);
        glVertex2f(-41,-1);
        glVertex2f(-45,-1);
        glEnd();

        glBegin(GL_QUADS);
        glColor3ub(255,252,221);
        glVertex2f(-48,-10);
        glVertex2f(-45,-10);
        glVertex2f(-45,-1);
        glVertex2f(-49,-1);
        glEnd();

        glBegin(GL_LINES);
        glColor3ub(15,49,77);
        glVertex2f(-45,-10);
        glVertex2f(-45,3);
        glEnd();

glBegin(GL_LINES);
        glColor3ub(15,49,77);
        glVertex2f(-48,-10);
        glVertex2f(-42,-10);
        glVertex2f(-41,-1);
        glVertex2f(-49,-1);
        glEnd();

glBegin(GL_LINES);
        glColor3ub(15,49,77);
        glVertex2f(-42,-10);
        glVertex2f(-41,-1);
        glEnd();
```

```
        glBegin(GL_LINES);
        glColor3ub(15,49,77);
        glVertex2f(-48,-10);
        glVertex2f(-49,-1);
        glEnd();

        glBegin(GL_TRIANGLES);
        glColor3ub(15,49,77);
        glVertex2f(-49,-1);
        glVertex2f(-41,-1);
        glVertex2f(-45,3);
        glEnd();

    //Left Stand
        glBegin(GL_QUADS);
        glColor3ub(15,49,77);
        glVertex2f(30,-35);
        glVertex2f(34,-35);
        glVertex2f(34,-18);
        glVertex2f(30,-18);
        glEnd();

        //Right Stand
        glBegin(GL_QUADS);
        glColor3ub(15,49,77);
        glVertex2f(66,-35);
        glVertex2f(70,-35);
        glVertex2f(70,-18);
        glVertex2f(66,-18);
        glEnd();

    //Bench
    glBegin(GL_QUADS);
        glColor3ub(160,82,45);
        glVertex2f(25,-20);
        glVertex2f(75,-20);
        glVertex2f(73,-5);
        glVertex2f(27,-5);
        glEnd();




        glFlush();  // Render now
```

```
}


/* Main function: GLUT runs as a console application starting at main()  */
int main(int argc, char** argv) {
        glutInit(&argc, argv);
        glutInitWindowSize(700, 700);       // Initialize GLUT
        glutCreateWindow("Rectangle");  // Create window with the given title
        //glutInitWindowSize(320, 320);   // Set the window's initial width & height
        glutInitWindowPosition(50, 50); // Position the window's initial top-left corner
        glutDisplayFunc(display);      // Register callback handler for window re-paint event
        initGL();                       // Our own OpenGL initialization
        gluOrtho2D(-100, 100, -100, 100);
        glutMainLoop();                 // Enter the event-processing loop
        return 0;
}
```

Output Screenshot (Full Screen)-