Lab Taks-4

Submission Guidelines-

- Rename the file to your id only. If your id is 18-XXXXX-1, then the file name must be 18-XXXXX-1.docx.
- Must submit within time that will be discussed in class VUES to the section named Lab Tak-4
- Must include resources for all the section in the table

Question- 1

Draw the scenario of a traffic signal

Graph Plot (Picture)-

(Not Needed)

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);

//Land 1
 glBegin(GL_QUADS);
glColor3ub(34,139,34);
glVertex2f(10,10);
glVertex2f(100,10);
glVertex2f(100,100);
glVertex2f(10,100);
glEnd();

//Land 2
 glBegin(GL_QUADS);
glColor3ub(34,139,34);
glVertex2f(-100,10);
```

```
glVertex2f(-10,10);
glVertex2f(-10,100);
glVertex2f(-100,100);
glEnd();

//Land 3
 glBegin(GL_QUADS);
glColor3ub(229,201,164);
glVertex2f(-100,-10);
glVertex2f(-100,-100);
glVertex2f(-10,-100);
glVertex2f(-10,-10);
glEnd();

//Land 4
 glBegin(GL_QUADS);
glColor3ub(229,201,164);
glVertex2f(10,-100);
glVertex2f(100,-100);
glVertex2f(100,-10);
glVertex2f(10,-10);
glEnd();

 //Road 1
 glBegin(GL_QUADS);
glColor3ub(159,159,157);
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();

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

```
glVertex2f(100,-2);
glVertex2f(100,2);
glVertex2f(84,2);
glEnd();

//Road 2
 glBegin(GL_QUADS);
glColor3ub(159,159,157);
glVertex2f(-10,-100);
glVertex2f(10,-100);
glVertex2f(10,100);
glVertex2f(-10,100);
glEnd();

//Cross2
 glBegin(GL_QUADS);
glColor3ub(255,250,250);
glVertex2f(-20,-2);
glVertex2f(30,-2);
glVertex2f(30,2);
glVertex2f(-20,2);
glEnd();

//Cross1
 glBegin(GL_QUADS);
glColor3ub(255,250,250);
glVertex2f(-2,-100);
glVertex2f(2,-100);
glVertex2f(2,-84);
glVertex2f(-2,-84);
glEnd();

//Cross2
 glBegin(GL_QUADS);
glColor3ub(255,250,250);
glVertex2f(-2,-60);
glVertex2f(2,-60);
glVertex2f(2,-25);
glVertex2f(-2,-25);
glEnd();

//Cross3
 glBegin(GL_QUADS);
glColor3ub(255,250,250);
```

```
glVertex2f(-2,20);
glVertex2f(2,20);
glVertex2f(2,55);
glVertex2f(-2,55);
glEnd();

//Cross4
 glBegin(GL_QUADS);
glColor3ub(255,250,250);
glVertex2f(-2,80);
glVertex2f(2,80);
glVertex2f(2,100);
glVertex2f(-2,100);
glEnd();

//Light ground
 glBegin(GL_QUADS);
glColor3ub(29,29,31);
glVertex2f(-34,-24);
glVertex2f(-42,-24);
glVertex2f(-44,-36);
glVertex2f(-31,-36);
glEnd();

//Traffic Light
 glBegin(GL_QUADS);
glColor3ub(105,105,105);
glVertex2f(-40,20);
glVertex2f(-35,20);
glVertex2f(-35,-30);
glVertex2f(-40,-30);
glEnd();

//Traffic up
 glBegin(GL_QUADS);
glColor3ub(105,105,105);
glVertex2f(-40,20);
glVertex2f(5,20);
glVertex2f(5,25);
glVertex2f(-40,25);
glEnd();

//Light box
 glBegin(GL_QUADS);
```

```
glColor3ub(75,0,130);
glVertex2f(-30,8);
glVertex2f(5,8);
glVertex2f(5,20);
glVertex2f(-30,20);
glEnd();

//Red light
glLineWidth(7.5);
          glBegin(GL_POLYGON);
          for(int i=0;i<200;i++)
     {
        glColor3ub(246, 51, 9);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=5;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x+0,y+15);
     }
          glEnd();
          //Yellow light
glLineWidth(7.5);
          glBegin(GL_POLYGON);
          for(int i=0;i<200;i++)
     {
        glColor3ub(255,255,0);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=5;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x-12,y+15);
     }
          glEnd();
   //Green light
glLineWidth(7.5);
          glBegin(GL_POLYGON);
          for(int i=0;i<200;i++)
     {
        glColor3ub(0,128,0);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=5;
```

```
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x-24,y+15);
    }
        glEnd();

// Building 1
glBegin(GL_QUADS);
glColor3ub(162,143,165);
glVertex2f(22,22);
glVertex2f(42,22);
glVertex2f(42,94);
glVertex2f(22,94);
glEnd();
//Door
glBegin(GL_QUADS);
glColor3ub(215,93,52);
glVertex2f(30,22);
glVertex2f(34,22);
glVertex2f(34,34);
glVertex2f(30,34);
glEnd();
//Window ground left
glBegin(GL_QUADS);
glColor3ub(80,86,134);
glVertex2f(43,25);
glVertex2f(45,25);
glVertex2f(45,29);
glVertex2f(43,29);
glEnd();


// Building 2
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();

// Building 3
glBegin(GL_QUADS);
glColor3ub(191,134,140);
```

```
glVertex2f(64,23);
glVertex2f(84,23);
glVertex2f(84,93);
glVertex2f(64,93);
glEnd();

// Building 4
glBegin(GL_QUADS);
glColor3ub(172,102,152);
glVertex2f(-45,30);
glVertex2f(-20,30);
glVertex2f(-20,95);
glVertex2f(-45,95);
glEnd();

// Building 5
glBegin(GL_QUADS);
glColor3ub(133,172,201);
glVertex2f(-75,30);
glVertex2f(-55,30);
glVertex2f(-55,93);
glVertex2f(-75,93);
glEnd();
// Building 5
glBegin(GL_QUADS);
glColor3ub(244,170,135);
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();

//Car
glBegin(GL_QUADS);
glColor3ub(220,20,60);
glVertex2f(45,-1);
glVertex2f(55,-1);
glVertex2f(58,4);
glVertex2f(43,4);
glEnd();

//Back Side
glBegin(GL_QUADS);
glColor3ub(220,20,60);
```

```
glVertex2f(55,-1);
glVertex2f(65,-1);
glVertex2f(62,4);
glVertex2f(53,4);
glEnd();

//Back window
glBegin(GL_TRIANGLES);
glColor3ub(118,122,121);
glVertex2f(58,4);
glVertex2f(60,4);
glVertex2f(53,13);
glEnd();

//Font Side
glBegin(GL_QUADS);
glColor3ub(220,20,60);
glVertex2f(35,-1);
glVertex2f(45,-1);
glVertex2f(46.5,4);
glVertex2f(37,4);
glEnd();

//Font window
glBegin(GL_TRIANGLES);
glColor3ub(121,121,121);
glVertex2f(40,4);
glVertex2f(43,4);
glVertex2f(48,13);
glEnd();

//Upper Side
glBegin(GL_QUADS);
glColor3ub(220,20,60);
glVertex2f(43,4);
glVertex2f(58,4);
glVertex2f(53,13);
glVertex2f(48,13);
glEnd();

//Right Font Window
glBegin(GL_QUADS);
glColor3ub(121,121,121);
glVertex2f(47,6);
```

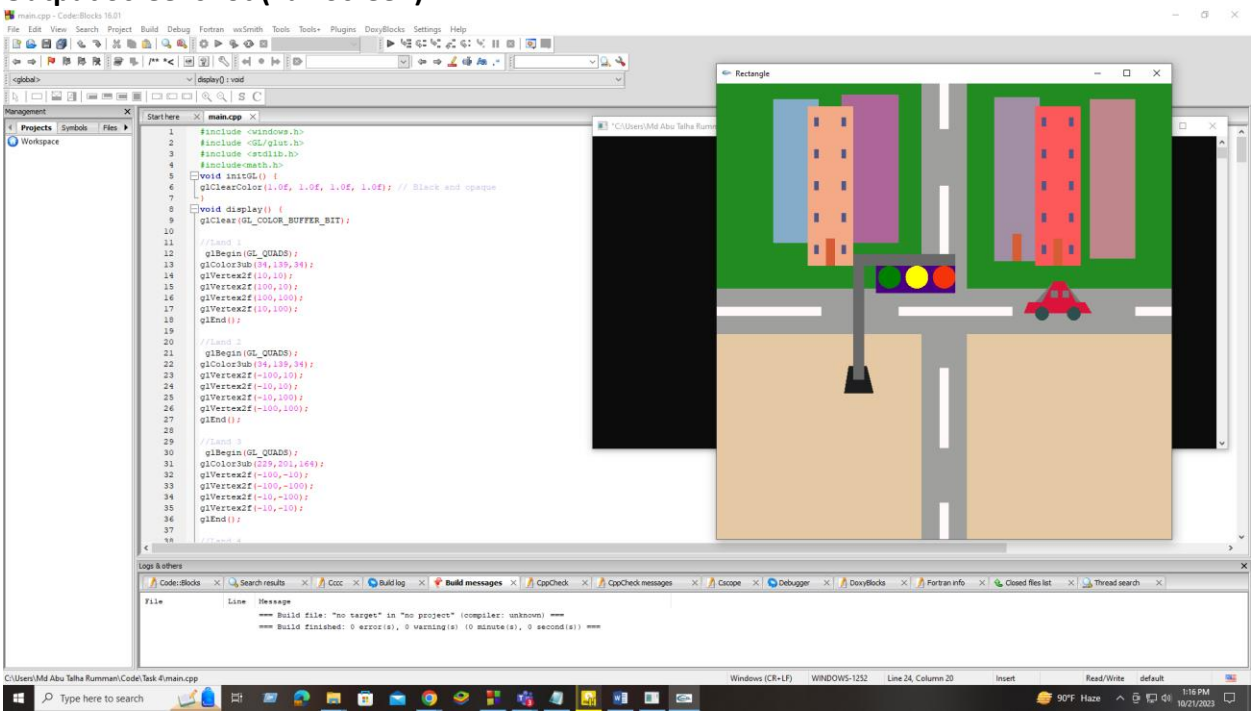
```
glVertex2f(50,6);
glVertex2f(50,9);
glVertex2f(47,9);
glEnd();

//Right back Window
glBegin(GL_QUADS);
glColor3ub(121,121,121);
glVertex2f(52,6);
glVertex2f(55,6);
glVertex2f(55,9);
glVertex2f(52,9);
glEnd();
//Back Wheel
glLineWidth(7.5);
          glBegin(GL_POLYGON);
          for(int i=0;i<200;i++)
     {
        glColor3ub(47,79,79);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=3;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x+57,y-1);
     }
          glEnd();

   //Font Wheel
glLineWidth(7.5);
          glBegin(GL_POLYGON);
          for(int i=0;i<200;i++)
     {
        glColor3ub(47,79,79);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=3;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x+43,y-1);
     }
          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 two village scenarios for day and night

Graph Plot (Picture)-

(Not Needed)

```
Code-(Day)
#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);


//sky
glBegin(GL_QUADS);
glColor3ub(135,206,250);
glVertex2f(-100,60);
glVertex2f(100,60);
glVertex2f(100,100);
glVertex2f(-100,100);
glEnd();

//Far tree1
glBegin(GL_QUADS);
glColor3ub(98,50,2);
glVertex2f(21,60);
glVertex2f(26,60);
glVertex2f(26,75);
glVertex2f(21,75);
glEnd();

//Far tree leaf
glLineWidth(7.5);
          glBegin(GL_POLYGON);
          for(int i=0;i<200;i++)
      {
         glColor3ub(32,140,31);
         float pi=3.1416;
         float A=(i*2*pi)/200;
         float r=7;
         float x = r * cos(A);
         float y = r * sin(A);
         glVertex2f(x+23.5,y+80);
```

```
    }
        glEnd();

//Far tree2
glBegin(GL_QUADS);
glColor3ub(98,50,2);
glVertex2f(-92,60);
glVertex2f(-87,60);
glVertex2f(-87,75);
glVertex2f(-92,75);
glEnd();

//Far tree leaf
glLineWidth(7.5);
         glBegin(GL_POLYGON);
         for(int i=0;i<200;i++)
    {
       glColor3ub(32,140,31);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=7;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x-89,y+80);
    }
         glEnd();

//Front forest
glLineWidth(7.5);
         glBegin(GL_POLYGON);
         for(int i=0;i<200;i++)
    {
       glColor3ub(32,140,31);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=12;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x-100,y+60);
    }
         glEnd();

         glLineWidth(7.5);
         glBegin(GL_POLYGON);
```

```
        for(int i=0;i<200;i++)
    {
      glColor3ub(32,140,31);
      float pi=3.1416;
      float A=(i*2*pi)/200;
      float r=12;
      float x = r * cos(A);
      float y = r * sin(A);
      glVertex2f(x-80,y+60);
    }
        glEnd();

                glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
      glColor3ub(34,139,34);
      float pi=3.1416;
      float A=(i*2*pi)/200;
      float r=20;
      float x = r * cos(A);
      float y = r * sin(A);
      glVertex2f(x-60,y+57);
    }

        glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
      glColor3ub(34,139,34);
      float pi=3.1416;
      float A=(i*2*pi)/200;
      float r=12;
      float x = r * cos(A);
      float y = r * sin(A);
      glVertex2f(x-35,y+60);
    }
        glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
      glColor3ub(34,139,34);
      float pi=3.1416;
      float A=(i*2*pi)/200;
```

```
        float r=12;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x-10,y+60);
    }
         glLineWidth(7.5);
         glBegin(GL_POLYGON);
         for(int i=0;i<200;i++)
    {
        glColor3ub(34,139,34);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=12;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x+10,y+60);
    }
         glEnd();

         glLineWidth(7.5);
         glBegin(GL_POLYGON);
         for(int i=0;i<200;i++)
    {
        glColor3ub(34,139,34);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=20;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x+70,y+60);
    }
         glEnd();

                  glLineWidth(7.5);
         glBegin(GL_POLYGON);
         for(int i=0;i<200;i++)
    {
        glColor3ub(34,139,34);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=5;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x+22,y+60);
```

```
    }
        glEnd();
        glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
       glColor3ub(34,139,34);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=15;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x+40,y+60);
    }
        glEnd();
        glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
       glColor3ub(34,139,34);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=12;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x+100,y+60);
    }
        glEnd();

        //Sun
glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
       glColor3ub(255,121,0);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=12;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x,y+100);
    }
        glEnd();
```

```
//Village
glBegin(GL_QUADS);
glColor3ub(99,255,49);
glVertex2f(-100,-100);
glVertex2f(100,-100);
glVertex2f(100,60);
glVertex2f(-100,60);
glEnd();

//River Bank1
glBegin(GL_QUADS);
glColor3ub(152,154,107);
glVertex2f(-100,25);
glVertex2f(100,25);
glVertex2f(100,30);
glVertex2f(-100,30);
glEnd();

//River Bank2
glBegin(GL_QUADS);
glColor3ub(152,154,107);
glVertex2f(27,-100);
glVertex2f(100,-100);
glVertex2f(100,30);
glVertex2f(20,30);
glEnd();

//River
glBegin(GL_POLYGON);
glColor3ub(58,176,204);
glVertex2f(35,60);
glVertex2f(100,60);
glVertex2f(100,-100);
glVertex2f(50,-100);
glEnd();

glBegin(GL_QUADS);
glColor3ub(58,176,204);
glVertex2f(-100,30);
glVertex2f(100,30);
glVertex2f(100,60);
glVertex2f(-100,60);
glEnd();
```

```
//Wave 1
glBegin(GL_TRIANGLES);
glColor3ub(58,176,204);
glVertex2f(25,20);
glVertex2f(55,10);
glVertex2f(30,40);
glEnd();

//Wave 2
glBegin(GL_TRIANGLES);
glColor3ub(58,176,204);
glVertex2f(30,0);
glVertex2f(50,-10);
glVertex2f(50,30);
glEnd();

//Wave 2.5
glBegin(GL_TRIANGLES);
glColor3ub(58,176,204);
glVertex2f(35,-10);
glVertex2f(55,-40);
glVertex2f(50,0);
glEnd();

//Wave 3
glBegin(GL_TRIANGLES);
glColor3ub(58,176,204);
glVertex2f(35,-40);
glVertex2f(55,-60);
glVertex2f(50,-10);
glEnd();

//Wave 4
glBegin(GL_QUADS);
glColor3ub(58,176,204);
glVertex2f(30,-70);
glVertex2f(60,-90);
glVertex2f(60,-60);
glVertex2f(50,-40);
glEnd();

//Wave 5
glBegin(GL_QUADS);
glColor3ub(58,176,204);
```

```
glVertex2f(40,-90);
glVertex2f(50,-90);
glVertex2f(50,-80);
glVertex2f(40,-80);
glEnd();

//Wave 6
glBegin(GL_TRIANGLES);
glColor3ub(58,176,204);
glVertex2f(30,-100);
glVertex2f(50,-100);
glVertex2f(50,-80);
glEnd();
//House
glBegin(GL_TRIANGLES);
glColor3ub(163,90,45);
glVertex2f(-90,-20);
glVertex2f(-60,-20);
glVertex2f(-75,0);
glEnd();

glBegin(GL_QUADS);
glColor3ub(146,4,3);
glVertex2f(-60,-20);
glVertex2f(0,-20);
glVertex2f(-15,0);
glVertex2f(-75,0);
glEnd();

glBegin(GL_QUADS);
glColor3ub(163,90,45);
glVertex2f(-60,-50);
glVertex2f(0,-50);
glVertex2f(0,-20);
glVertex2f(-60,-20);
glEnd();

glBegin(GL_QUADS);
glColor3ub(163,90,45);
glVertex2f(-90,-50);
glVertex2f(-60,-50);
glVertex2f(-60,-20);
glVertex2f(-90,-20);
glEnd();
```

```
//Border
glBegin(GL_QUADS);
glColor3ub(0,0,0);
glVertex2f(-60.5,-50);
glVertex2f(-60,-50);
glVertex2f(-60,-20);
glVertex2f(-60.5,-20);
glEnd();

//Door
glBegin(GL_QUADS);
glColor3ub(184,117,72);
glVertex2f(-34,-50);
glVertex2f(-26,-50);
glVertex2f(-26,-35);
glVertex2f(-34,-35);
glEnd();

//Lock
glLineWidth(7.5);
glBegin(GL_POLYGON);
for(int i=0;i<200;i++)
   {
      glColor3ub(169,169,169);
      float pi=3.1416;
      float A=(i*2*pi)/200;
      float r=1.5;
      float x = r * cos(A);
      float y = r * sin(A);
      glVertex2f(x-28.5,y-43);
   }
glEnd();

//Window left
glBegin(GL_QUADS);
glColor3ub(0,129,127);
glVertex2f(-50,-42);
glVertex2f(-44,-42);
glVertex2f(-44,-36);
glVertex2f(-50,-36);
glEnd();

//Window Right
```

```
glBegin(GL_QUADS);
glColor3ub(0,129,127);
glVertex2f(-16,-42);
glVertex2f(-10,-42);
glVertex2f(-10,-36);
glVertex2f(-16,-36);
glEnd();

//Window 3rd
glBegin(GL_QUADS);
glColor3ub(0,129,127);
glVertex2f(-78.25,-42);
glVertex2f(-72.25,-42);
glVertex2f(-72.25,-36);
glVertex2f(-78.25,-36);
glEnd();

//Ground border
glBegin(GL_QUADS);
glColor3ub(0,0,0);
glVertex2f(-90,-58);
glVertex2f(0,-58);
glVertex2f(0,-50);
glVertex2f(-90,-50);
glEnd();

//Tree 1
glBegin(GL_QUADS);
glColor3ub(98,50,2);
glVertex2f(10,-55);
glVertex2f(18,-55);
glVertex2f(18,0);
glVertex2f(10,0);
glEnd();

//Leaf 1
glLineWidth(7.5);
glBegin(GL_POLYGON);
for(int i=0;i<200;i++)
   {
      glColor3ub(47,205,48);
      float pi=3.1416;
      float A=(i*2*pi)/200;
      float r=20;
```

```
      float x = r * cos(A);
      float y = r * sin(A);
      glVertex2f(x+13,y+0);
   }
glEnd();

//Leaf 2
glLineWidth(7.5);
glBegin(GL_POLYGON);
for(int i=0;i<200;i++)
   {
      glColor3ub(47,205,48);
      float pi=3.1416;
      float A=(i*2*pi)/200;
      float r=15;
      float x = r * cos(A);
      float y = r * sin(A);
      glVertex2f(x+13,y+24);
   }
glEnd();

//Tree 2
glBegin(GL_QUADS);
glColor3ub(98,50,2);
glVertex2f(-60,0);
glVertex2f(-55,0);
glVertex2f(-55,10);
glVertex2f(-60,10);
glEnd();

//Leaf 1
glBegin(GL_TRIANGLES);
glColor3ub(0,100,0);
glVertex2f(-70,10);
glVertex2f(-45,10);
glVertex2f(-57.5,25);
glEnd();

//Leaf 2
glBegin(GL_TRIANGLES);
glColor3ub(0,128,0);
glVertex2f(-70,17);
glVertex2f(-45,17);
glVertex2f(-57.5,35);
```

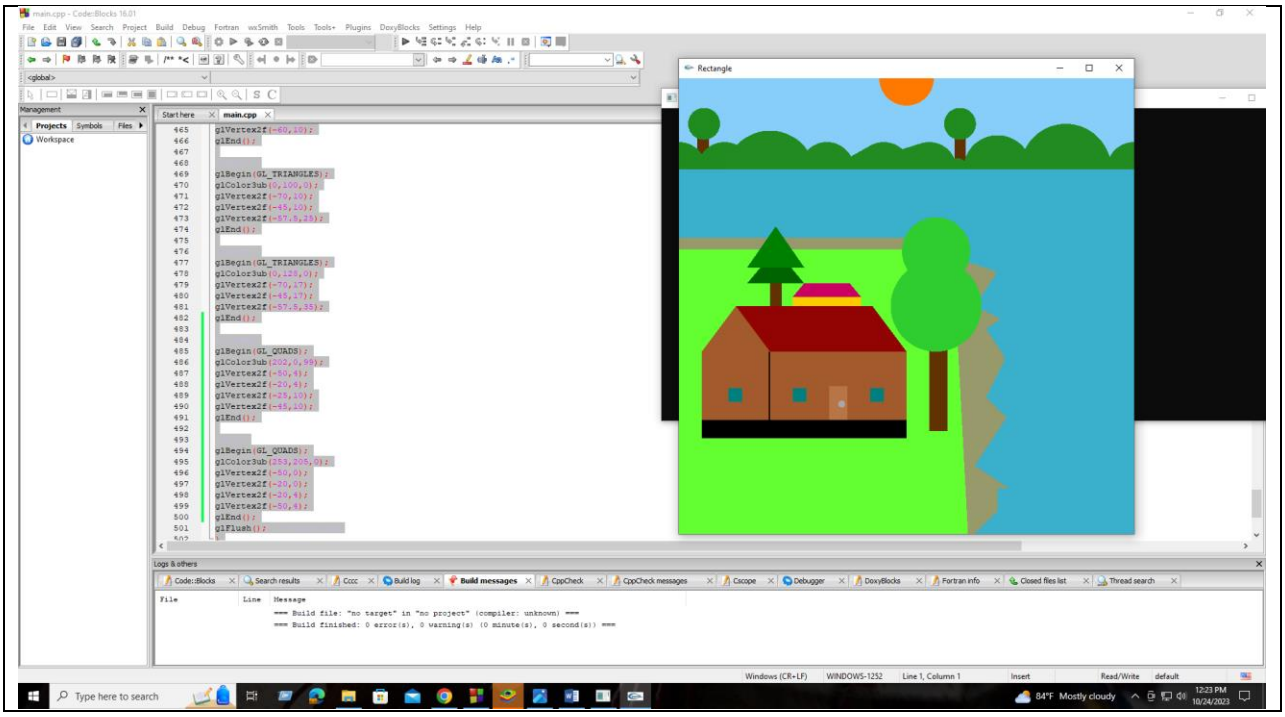
```
glEnd();

//House2
glBegin(GL_QUADS);
glColor3ub(202,0,99);
glVertex2f(-50,4);
glVertex2f(-20,4);
glVertex2f(-25,10);
glVertex2f(-45,10);
glEnd();

//Body
glBegin(GL_QUADS);
glColor3ub(253,205,0);
glVertex2f(-50,0);
glVertex2f(-20,0);
glVertex2f(-20,4);
glVertex2f(-50,4);
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)-

(Day)-



Code-(Night)

```
#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);


//sky
glBegin(GL_QUADS);
glColor3ub(0,0,0);
glVertex2f(-100,60);
glVertex2f(100,60);
glVertex2f(100,100);
glVertex2f(-100,100);
glEnd();

//Far tree1
glBegin(GL_QUADS);
```

```
glColor3ub(98,50,2);
glVertex2f(21,60);
glVertex2f(26,60);
glVertex2f(26,75);
glVertex2f(21,75);
glEnd();

//Far tree leaf
glLineWidth(7.5);
          glBegin(GL_POLYGON);
          for(int i=0;i<200;i++)
     {
        glColor3ub(32,140,31);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=7;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x+23.5,y+80);
     }
          glEnd();

//Far tree2
glBegin(GL_QUADS);
glColor3ub(98,50,2);
glVertex2f(-92,60);
glVertex2f(-87,60);
glVertex2f(-87,75);
glVertex2f(-92,75);
glEnd();

//Far tree leaf
glLineWidth(7.5);
          glBegin(GL_POLYGON);
          for(int i=0;i<200;i++)
     {
        glColor3ub(32,140,31);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=7;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x-89,y+80);
     }
```

```
        glEnd();

//Front forest
glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
       glColor3ub(32,140,31);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=12;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x-100,y+60);
    }
        glEnd();

        glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
       glColor3ub(32,140,31);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=12;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x-80,y+60);
    }
        glEnd();

                glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
       glColor3ub(34,139,34);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=20;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x-60,y+57);
    }
```

```
        glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
       glColor3ub(34,139,34);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=12;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x-35,y+60);
    }
        glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
       glColor3ub(34,139,34);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=12;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x-10,y+60);
    }
        glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
       glColor3ub(34,139,34);
       float pi=3.1416;
       float A=(i*2*pi)/200;
       float r=12;
       float x = r * cos(A);
       float y = r * sin(A);
       glVertex2f(x+10,y+60);
    }
        glEnd();

        glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
       glColor3ub(34,139,34);
       float pi=3.1416;
```

```
    float A=(i*2*pi)/200;
    float r=20;
    float x = r * cos(A);
    float y = r * sin(A);
    glVertex2f(x+70,y+60);
  }
     glEnd();

              glLineWidth(7.5);
     glBegin(GL_POLYGON);
     for(int i=0;i<200;i++)
  {
    glColor3ub(34,139,34);
    float pi=3.1416;
    float A=(i*2*pi)/200;
    float r=5;
    float x = r * cos(A);
    float y = r * sin(A);
    glVertex2f(x+22,y+60);
  }
     glEnd();
     glLineWidth(7.5);
     glBegin(GL_POLYGON);
     for(int i=0;i<200;i++)
  {
    glColor3ub(34,139,34);
    float pi=3.1416;
    float A=(i*2*pi)/200;
    float r=15;
    float x = r * cos(A);
    float y = r * sin(A);
    glVertex2f(x+40,y+60);
  }
     glEnd();
     glLineWidth(7.5);
     glBegin(GL_POLYGON);
     for(int i=0;i<200;i++)
  {
    glColor3ub(34,139,34);
    float pi=3.1416;
    float A=(i*2*pi)/200;
    float r=12;
    float x = r * cos(A);
    float y = r * sin(A);
```

```
        glVertex2f(x+100,y+60);
    }
        glEnd();

        //Sun
  glLineWidth(7.5);
        glBegin(GL_POLYGON);
        for(int i=0;i<200;i++)
    {
        glColor3ub(189,183,107);
        float pi=3.1416;
        float A=(i*2*pi)/200;
        float r=12;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y+100);
    }
        glEnd();

//Village
glBegin(GL_QUADS);
glColor3ub(32,140,31);
glVertex2f(-100,-100);
glVertex2f(100,-100);
glVertex2f(100,60);
glVertex2f(-100,60);
glEnd();

//River Bank1
glBegin(GL_QUADS);
glColor3ub(152,154,107);
glVertex2f(-100,25);
glVertex2f(100,25);
glVertex2f(100,30);
glVertex2f(-100,30);
glEnd();

//River Bank2
glBegin(GL_QUADS);
glColor3ub(152,154,107);
glVertex2f(27,-100);
glVertex2f(100,-100);
glVertex2f(100,30);
glVertex2f(20,30);
```

```
glEnd();

//River
glBegin(GL_POLYGON);
glColor3ub(25,25,112);
glVertex2f(35,60);
glVertex2f(100,60);
glVertex2f(100,-100);
glVertex2f(50,-100);
glEnd();

glBegin(GL_QUADS);
glColor3ub(25,25,112);
glVertex2f(-100,30);
glVertex2f(100,30);
glVertex2f(100,60);
glVertex2f(-100,60);
glEnd();

//Wave 1
glBegin(GL_TRIANGLES);
glColor3ub(25,25,112);
glVertex2f(25,20);
glVertex2f(55,10);
glVertex2f(30,40);
glEnd();

//Wave 2
glBegin(GL_TRIANGLES);
glColor3ub(25,25,112);
glVertex2f(30,0);
glVertex2f(50,-10);
glVertex2f(50,30);
glEnd();

//Wave 2.5
glBegin(GL_TRIANGLES);
glColor3ub(25,25,112);
glVertex2f(35,-10);
glVertex2f(55,-40);
glVertex2f(50,0);
glEnd();

//Wave 3
```

```
glBegin(GL_TRIANGLES);
glColor3ub(25,25,112);
glVertex2f(35,-40);
glVertex2f(55,-60);
glVertex2f(50,-10);
glEnd();

//Wave 4
glBegin(GL_QUADS);
glColor3ub(25,25,112);
glVertex2f(30,-70);
glVertex2f(60,-90);
glVertex2f(60,-60);
glVertex2f(50,-40);
glEnd();

//Wave 5
glBegin(GL_QUADS);
glColor3ub(25,25,112);
glVertex2f(40,-90);
glVertex2f(50,-90);
glVertex2f(50,-80);
glVertex2f(40,-80);
glEnd();

//Wave 6
glBegin(GL_TRIANGLES);
glColor3ub(25,25,112);
glVertex2f(30,-100);
glVertex2f(50,-100);
glVertex2f(50,-80);
glEnd();
//House
glBegin(GL_TRIANGLES);
glColor3ub(163,90,45);
glVertex2f(-90,-20);
glVertex2f(-60,-20);
glVertex2f(-75,0);
glEnd();

glBegin(GL_QUADS);
glColor3ub(146,4,3);
glVertex2f(-60,-20);
glVertex2f(0,-20);
```

```
glVertex2f(-15,0);
glVertex2f(-75,0);
glEnd();

glBegin(GL_QUADS);
glColor3ub(163,90,45);
glVertex2f(-60,-50);
glVertex2f(0,-50);
glVertex2f(0,-20);
glVertex2f(-60,-20);
glEnd();

glBegin(GL_QUADS);
glColor3ub(163,90,45);
glVertex2f(-90,-50);
glVertex2f(-60,-50);
glVertex2f(-60,-20);
glVertex2f(-90,-20);
glEnd();

//Border
glBegin(GL_QUADS);
glColor3ub(0,0,0);
glVertex2f(-60.5,-50);
glVertex2f(-60,-50);
glVertex2f(-60,-20);
glVertex2f(-60.5,-20);
glEnd();

//Door
glBegin(GL_QUADS);
glColor3ub(184,117,72);
glVertex2f(-34,-50);
glVertex2f(-26,-50);
glVertex2f(-26,-35);
glVertex2f(-34,-35);
glEnd();

//Lock
glLineWidth(7.5);
glBegin(GL_POLYGON);
for(int i=0;i<200;i++)
   {
      glColor3ub(169,169,169);
```

```
      float pi=3.1416;
      float A=(i*2*pi)/200;
      float r=1.5;
      float x = r * cos(A);
      float y = r * sin(A);
      glVertex2f(x-28.5,y-43);
   }
glEnd();

//Window left
glBegin(GL_QUADS);
glColor3ub(0,129,127);
glVertex2f(-50,-42);
glVertex2f(-44,-42);
glVertex2f(-44,-36);
glVertex2f(-50,-36);
glEnd();

//Window Right
glBegin(GL_QUADS);
glColor3ub(0,129,127);
glVertex2f(-16,-42);
glVertex2f(-10,-42);
glVertex2f(-10,-36);
glVertex2f(-16,-36);
glEnd();

//Window 3rd
glBegin(GL_QUADS);
glColor3ub(0,129,127);
glVertex2f(-78.25,-42);
glVertex2f(-72.25,-42);
glVertex2f(-72.25,-36);
glVertex2f(-78.25,-36);
glEnd();

//Ground border
glBegin(GL_QUADS);
glColor3ub(0,0,0);
glVertex2f(-90,-58);
glVertex2f(0,-58);
glVertex2f(0,-50);
glVertex2f(-90,-50);
glEnd();
```

```
//Tree 1
glBegin(GL_QUADS);
glColor3ub(98,50,2);
glVertex2f(10,-55);
glVertex2f(18,-55);
glVertex2f(18,0);
glVertex2f(10,0);
glEnd();

//Leaf 1
glLineWidth(7.5);
glBegin(GL_POLYGON);
for(int i=0;i<200;i++)
   {
      glColor3ub(0,100,0);
      float pi=3.1416;
      float A=(i*2*pi)/200;
      float r=20;
      float x = r * cos(A);
      float y = r * sin(A);
      glVertex2f(x+13,y+0);
   }
glEnd();

//Leaf 2
glLineWidth(7.5);
glBegin(GL_POLYGON);
for(int i=0;i<200;i++)
   {
      glColor3ub(0,100,0);
      float pi=3.1416;
      float A=(i*2*pi)/200;
      float r=15;
      float x = r * cos(A);
      float y = r * sin(A);
      glVertex2f(x+13,y+24);
   }
glEnd();

//Tree 2
glBegin(GL_QUADS);
glColor3ub(98,50,2);
glVertex2f(-60,0);
```

```
glVertex2f(-55,0);
glVertex2f(-55,10);
glVertex2f(-60,10);
glEnd();

//Leaf 1
glBegin(GL_TRIANGLES);
glColor3ub(0,100,0);
glVertex2f(-70,10);
glVertex2f(-45,10);
glVertex2f(-57.5,25);
glEnd();

//Leaf 2
glBegin(GL_TRIANGLES);
glColor3ub(0,128,0);
glVertex2f(-70,17);
glVertex2f(-45,17);
glVertex2f(-57.5,35);
glEnd();

//House2
glBegin(GL_QUADS);
glColor3ub(202,0,99);
glVertex2f(-50,4);
glVertex2f(-20,4);
glVertex2f(-25,10);
glVertex2f(-45,10);
glEnd();

//Body
glBegin(GL_QUADS);
glColor3ub(253,205,0);
glVertex2f(-50,0);
glVertex2f(-20,0);
glVertex2f(-20,4);
glVertex2f(-50,4);
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)-

(Night)-