

PROJECT SUBMISSION:- COMPUTER GRAPHICS:

TOPIC:- SCENERY GENERATOR

SUBMITTED BY:-

Chetanya Agarwal (211141)

Samriti Thakur (211150)

SUBMITTED TO:-

Mr. Prateek Thakral

SOURCE CODE:

```
#include <GL/glut.h>
#include <cmath>
#include <cstdlib>
#include <iostream>
#include <vector>
struct Hut {
    float x;
    float y;
    float width;
    float height;

    Hut(float x, float y, float width, float height)
        : x(x), y(y), width(width), height(height) {}
};
std::vector<Hut> huts;
bool isOverlap(float x, float y, float width, float height) {
    for (size_t i = 0; i < huts.size(); ++i) {
        const Hut& hut = huts[i];
        if (x < hut.x + hut.width + 5 &&
            x + width + 5 > hut.x &&
            y < hut.y + hut.height + 5 &&
            y + height + 5 > hut.y) {
            return true;
        }
    }
}
```

```

    }
}
return false;
}
bool isNearRiver(float x, float y, float width, float height) {
    float riverX1 = -100;
    float riverX2 = 100;
    float riverY1 = -500;
    float riverY2 = 150;

    return x < riverX2 && x + width > riverX1 && y < riverY2 && y + height > riverY1;
}
void drawMountains() {
    glColor3f(0.8f, 0.4f, 0.2f);

    // Mountain 1
    glBegin(GL_TRIANGLES);
    glVertex2f(-600, 150);
    glVertex2f(-500, 350);
    glVertex2f(-400, 150);
    glEnd();

    // Mountain 2
    glBegin(GL_TRIANGLES);
    glVertex2f(-400, 150);
    glVertex2f(-300, 350);
    glVertex2f(-200, 150);
    glEnd();

    // Mountain 3
    glBegin(GL_TRIANGLES);
    glVertex2f(-200, 150);
    glVertex2f(-100, 350);
    glVertex2f(0, 150);
    glEnd();

    // Mountain 4
    glBegin(GL_TRIANGLES);
    glVertex2f(0, 150);
    glVertex2f(100, 350);
    glVertex2f(200, 150);
    glEnd();

    // Mountain 5

```

```
glBegin(GL_TRIANGLES);
glVertex2f(200, 150);
glVertex2f(300, 350);
glVertex2f(400, 150);
glEnd();
```

```
// Mountain 6
```

```
glBegin(GL_TRIANGLES);
glVertex2f(400, 150);
glVertex2f(500, 350);
glVertex2f(600, 150);
glEnd();
```

```
}
```

```
void drawRiver() {
```

```
    glColor3f(0.0f, 0.2f, 0.4f); // Darker blue color for the river during sunset
    glBegin(GL_POLYGON);
    glVertex2f(-100, -500);
    glVertex2f(0, 150);
    glVertex2f(100, -500);
    glEnd();
```

```
}
```

```
void drawHuts() {
```

```
    for (size_t i = 0; i < huts.size(); ++i) {
        const Hut& hut = huts[i];
        float hutX = hut.x;
        float hutY = hut.y - 150.0f;
```

```
        // Set hut body color
```

```
        glColor3f(0.5f, 0.35f, 0.05f); // Brown color for huts
```

```
        // Main body of the hut
```

```
        glBegin(GL_QUADS);
        glVertex2f(hutX, hutY);
        glVertex2f(hutX, hutY + hut.height);
        glVertex2f(hutX + hut.width, hutY + hut.height);
        glVertex2f(hutX + hut.width, hutY);
        glEnd();
```

```
        // Set door color
```

```
        glColor3f(0.3f, 0.2f, 0.0f); // Dark brown color for doors
```

```
        // Door
```

```

    glBegin(GL_QUADS);
    glVertex2f(hutX + 7, hutY);
    glVertex2f(hutX + 7, hutY + 15);
    glVertex2f(hutX + 13, hutY + 15);
    glVertex2f(hutX + 13, hutY);
    glEnd();

    // Set roof color
    glColor3f(0.4f, 0.2f, 0.0f); // Roof color

    // Roof of the hut
    glBegin(GL_TRIANGLES);
    glVertex2f(hutX - 3, hutY + hut.height);
    glVertex2f(hutX + hut.width / 2, hutY + hut.height + 10);
    glVertex2f(hutX + hut.width + 3, hutY + hut.height);
    glEnd();
}
}

void drawStars() {
    glColor3f(1.0f, 1.0f, 1.0f); // White color for stars
    glPointSize(2.0f);
    glBegin(GL_POINTS);
    for (int i = 0; i < 500; ++i) {
        float x = (rand() % 1200) - 600;
        float y = (rand() % 400) + 200; // Adjusted Y-coordinate range
        glVertex2f(x, y);
    }
    glEnd();
}

void drawSun() {
    glColor3f(1.0f, 1.0f, 0.8f); // Yellow color for the sun
    glBegin(GL_POLYGON);
    for (int i = 0; i < 360; i += 5) {
        float theta = i * 3.14159 / 180;
        float x = 50 * cos(theta);
        float y = 50 * sin(theta) + 380;
        glVertex2f(x, y);
    }
    glEnd();
}

void drawMorningScenery() {
    glClearColor(0.4f, 0.6f, 1.0f, 1.0f);
    glClear(GL_COLOR_BUFFER_BIT);

```

```

// Darker green grass below the mountains during sunset
glColor3f(0.2f, 0.4f, 0.2f); // Darker green color for grass during sunset
glBegin(GL_POLYGON);
glVertex2f(-600, -500);
glVertex2f(-600, 150);
glVertex2f(600, 150);
glVertex2f(600, -500);
glEnd();

// Draw orangish semi-circle on the endpoints of the 3rd and 4th mountain
glColor3f(0.8f, 0.8f, 0.2f); // Orangish color for the semi-circle
glBegin(GL_POLYGON);
for (int i = 90; i <= 270; i += 5) {
    float theta = (i - 90) * 3.14159 / 180; // Rotate by 90 degrees clockwise
    float x = 90 * cos(theta); // Adjusted position with increased radius
    float y = 90 * sin(theta) + 150; // Adjusted y-position
    glVertex2f(x, y);
}
glEnd();

drawMountains();
drawRiver();

// Draw birds in the sky using the greater-than sign (>)
glColor3f(0.0f, 0.0f, 0.0f); // Black color for birds
glRasterPos2f(-500, 400);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(-450, 420);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(-400, 400);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(-350, 420);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(400, 400);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(450, 420);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(500, 400);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');

drawHuts();

glFlush();

```

```

}
void drawDarkYellowSun()
{
    // Dark yellow color for the sun
    glColor3f(0.8f, 0.8f, 0.2f);
    glBegin(GL_POLYGON);
    for (int i = 0; i < 360; i += 5) {
        float theta = i * 3.14159 / 180;
        float x = 50 * cos(theta);
        float y = 50 * sin(theta) + 380;
        glVertex2f(x, y);
    }
    glEnd();
}

void drawAfternoonMountains() {
    // Darken the mountains
    glColor3f(0.5f, 0.25f, 0.0f); // Dark brown color for mountains

    // Left-side mountains
    // Mountain 1
    glBegin(GL_TRIANGLES);
    glVertex2f(-600, 150);
    glVertex2f(-500, 350);
    glVertex2f(-400, 150);
    glEnd();

    // Mountain 2
    glBegin(GL_TRIANGLES);
    glVertex2f(-400, 150);
    glVertex2f(-300, 350);
    glVertex2f(-200, 150);
    glEnd();

    // Right-side mountains
    // Mountain 3
    glBegin(GL_TRIANGLES);
    glVertex2f(-200, 150);
    glVertex2f(-100, 350);
    glVertex2f(0, 150);
    glEnd();

    // Mountain 4
    glBegin(GL_TRIANGLES);
    glVertex2f(0, 150);

```

```

glVertex2f(100, 350);
glVertex2f(200, 150);
glEnd();

// Additional mountains
// Mountain 5
glBegin(GL_TRIANGLES);
glVertex2f(200, 150);
glVertex2f(300, 350);
glVertex2f(400, 150);
glEnd();

// Mountain 6
glBegin(GL_TRIANGLES);
glVertex2f(400, 150);
glVertex2f(500, 350);
glVertex2f(600, 150);
glEnd();
}

void drawNoonScenery() {
    glClearColor(0.4f, 0.6f, 1.0f, 1.0f); // Light blue for the sky
    glClear(GL_COLOR_BUFFER_BIT);

    // Darker green grass below the mountains during noon
    glColor3f(0.2f, 0.4f, 0.2f); // Dark green color for grass during noon
    glBegin(GL_POLYGON);
    glVertex2f(-600, -500);
    glVertex2f(-600, 150);
    glVertex2f(600, 150);
    glVertex2f(600, -500);
    glEnd();

    drawDarkYellowSun();
    drawAfternoonMountains();
    drawRiver();
    drawHuts();

    glFlush();
}

void drawEveningScenery() {
    glClearColor(0.4f, 0.6f, 1.0f, 1.0f); // Bluish-orange for the sky
    glClear(GL_COLOR_BUFFER_BIT);

```

```

// Darker green grass below the mountains during sunset
glColor3f(0.2f, 0.4f, 0.2f); // Darker green color for grass during sunset
glBegin(GL_POLYGON);
glVertex2f(-600, -500);
glVertex2f(-600, 150);
glVertex2f(600, 150);
glVertex2f(600, -500);
glEnd();

// Draw orangish semi-circle on the endpoints of the 3rd and 4th mountain
glColor3f(1.0f, 0.5f, 0.0f); // Orangish color for the semi-circle
glBegin(GL_POLYGON);
for (int i = 90; i <= 270; i += 5) {
    float theta = (i - 90) * 3.14159 / 180; // Rotate by 90 degrees clockwise
    float x = 90 * cos(theta); // Adjusted position with increased radius
    float y = 90 * sin(theta) + 150; // Adjusted y-position
    glVertex2f(x, y);
}
glEnd();

drawMountains();
drawRiver();

// Draw birds in the sky using the greater-than sign (>)
glColor3f(0.0f, 0.0f, 0.0f); // Black color for birds
glRasterPos2f(-500, 400);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(-450, 420);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(-400, 400);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(-350, 420);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(400, 400);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(450, 420);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');
glRasterPos2f(500, 400);
glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '>');

drawHuts();

glFlush();
}

```



```

void drawMountainsNight() {
    glColor3f(0.3f, 0.15f, 0.0f); // Dark brown color for night mountains

    // Draw left-side mountains
    glBegin(GL_TRIANGLES);
    glVertex2f(-600, 150);
    glVertex2f(-500, 350);
    glVertex2f(-400, 150);
    glEnd();

    glBegin(GL_TRIANGLES);
    glVertex2f(-400, 150);
    glVertex2f(-300, 350);
    glVertex2f(-200, 150);
    glEnd();

    // Draw right-side mountains
    glBegin(GL_TRIANGLES);
    glVertex2f(-200, 150);
    glVertex2f(-100, 350);
    glVertex2f(0, 150);
    glEnd();

    glBegin(GL_TRIANGLES);
    glVertex2f(0, 150);
    glVertex2f(100, 350);
    glVertex2f(200, 150);
    glEnd();

    // Additional mountains
    glBegin(GL_TRIANGLES);
    glVertex2f(200, 150);
    glVertex2f(300, 350);
    glVertex2f(400, 150);
    glEnd();

    glBegin(GL_TRIANGLES);
    glVertex2f(400, 150);
    glVertex2f(500, 350);
    glVertex2f(600, 150);
    glEnd();
}

void drawNightScenery() {

```

```

glClearColor(0.0f, 0.0f, 0.2f, 1.0f);
glClear(GL_COLOR_BUFFER_BIT);

// Darker grass below the mountains
glColor3f(0.0f, 0.3f, 0.0f); // Dark green color for grass
glBegin(GL_POLYGON);
glVertex2f(-600, -500);
glVertex2f(-600, 150);
glVertex2f(600, 150);
glVertex2f(600, -500);
glEnd();

drawStars();

// Draw the moon without shine
glColor3f(1.0f, 1.0f, 1.0f);
glBegin(GL_POLYGON);
for (int i = 0; i < 360; i += 5) {
    float theta = i * 3.14159 / 180;
    float x = 50 * cos(theta);
    float y = 50 * sin(theta) + 380;
    glVertex2f(x, y);
}
glEnd();

drawMountainsNight();

drawRiver();
drawHuts();

glFlush();
}

void display() {
    int choice;

    while (true) {
        std::cout << "Enter 1 for Morning, 2 for Noon, 3 for Evening, 4 for Night or 0 to
exit: ";
        std::cin >> choice;

        switch (choice) {
            case 0:
                exit(0);

```

```

        break;

    case 1:
        drawMorningScenery();
        break;

    case 2:
        drawNoonScenery();
        break;

    case 3:
        drawEveningScenery();
        break;

    case 4:
        drawNightScenery();
        break;

    default:
        std::cout << "Invalid choice. Please enter a valid option.\n";
        break;
}

glutSwapBuffers();
}
}

int main()
{
    glutInitWindowSize(800, 600);
    glutCreateWindow("Village Scenery");

    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-600, 600, -500, 500);

    // Place 55 huts away from each other
    for (int i = 0; i < 55; i++)
    {
        float hutX, hutY, hutWidth, hutHeight;
        do {
            hutX = -580 + rand() % 1200; // Increased scattering range
            hutY = -250 + rand() % 400;  // Adjusted Y-coordinate range
            hutWidth = 20.0f;

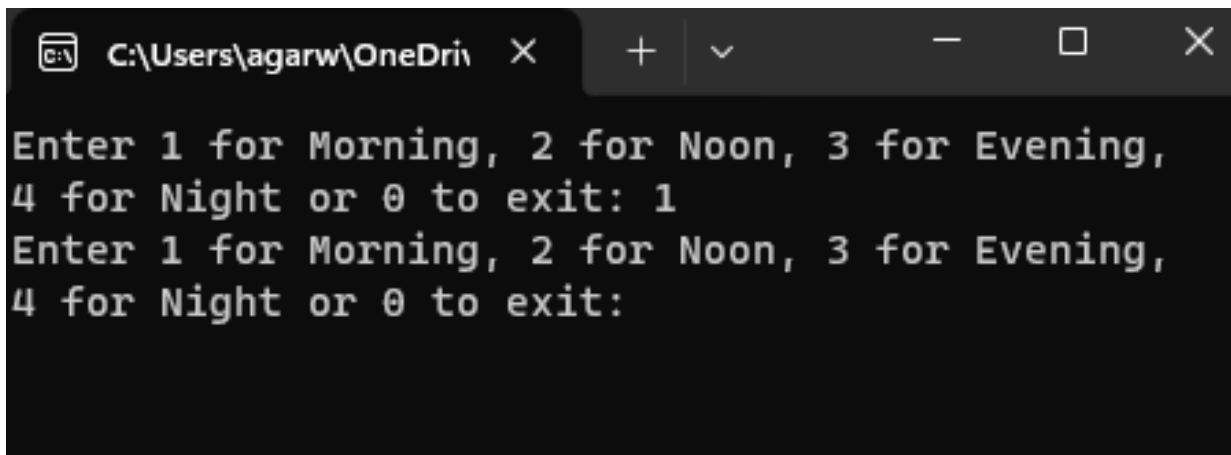
```

```
        hutHeight = 30.0f;
    } while (isOverlap(hutX, hutY, hutWidth, hutHeight) || isNearRiver(hutX, hutY,
hutWidth, hutHeight));

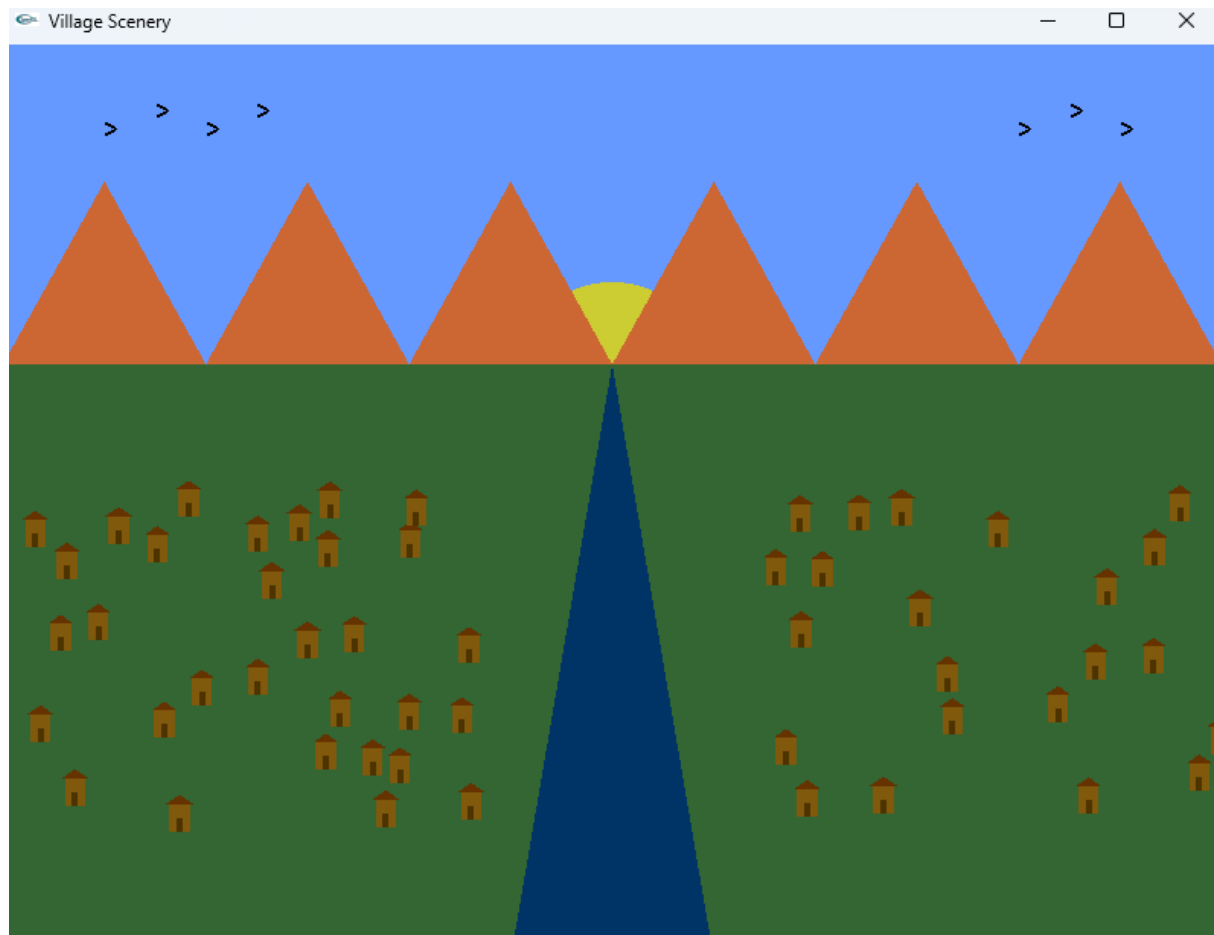
    huts.push_back(Hut(hutX, hutY, hutWidth, hutHeight));
}

glutDisplayFunc(display);
glutMainLoop();
return 0;
}
```

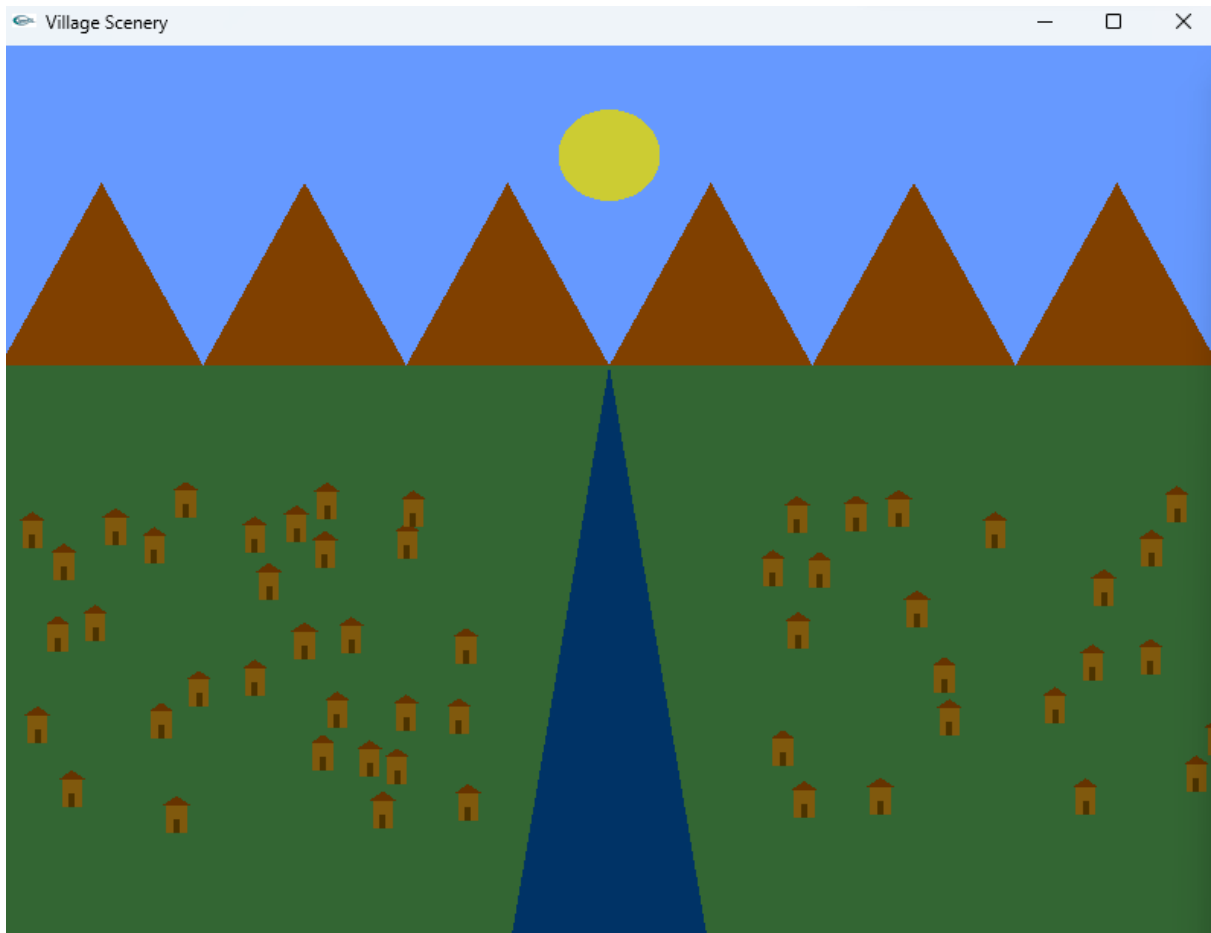
OUTPUT:

A screenshot of a Windows command prompt window. The title bar shows the file explorer icon, the path 'C:\Users\agarw\OneDri', and standard window controls (close, maximize, minimize). The command prompt has a black background with white text. It displays two identical prompts: 'Enter 1 for Morning, 2 for Noon, 3 for Evening, 4 for Night or 0 to exit:'. The first prompt has the number '1' entered, while the second prompt is empty.

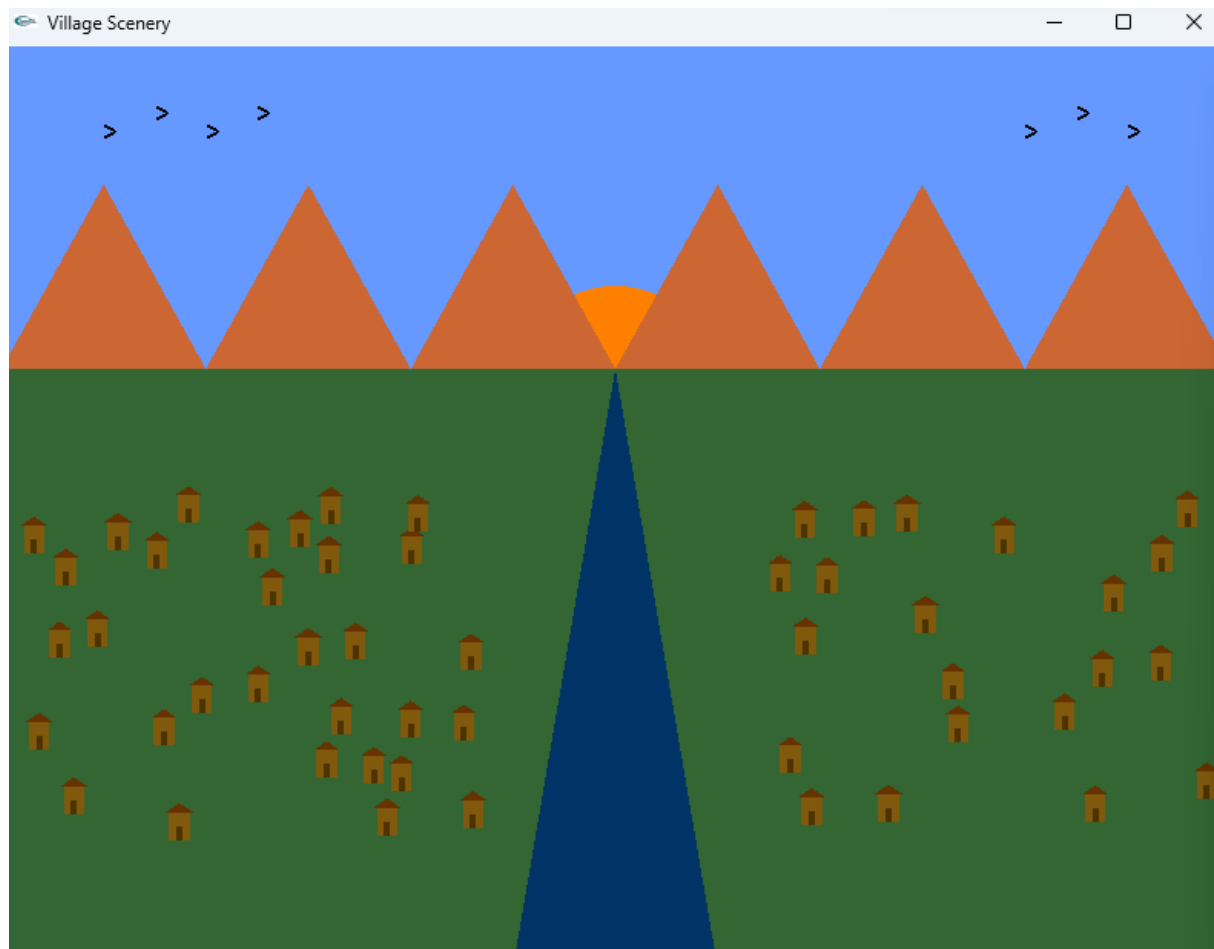
```
C:\Users\agarw\OneDri X + - □ X
Enter 1 for Morning, 2 for Noon, 3 for Evening,
4 for Night or 0 to exit: 1
Enter 1 for Morning, 2 for Noon, 3 for Evening,
4 for Night or 0 to exit:
```



MORNING



AFTERNOON



EVENING



NIGHT