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

struct Image
{
    unsigned long sizeX;
    unsigned long sizeY;
    char *data;
};

// Function to load textures


GLuint texture[3];




int ImageLoad(char *filename, struct Image *image)
{
    FILE *file;
    unsigned long size; //size of the image in bytes
    unsigned long i; //standard counter
    unsigned short int planes; //number of planes in image (must be 1)
    unsigned short int bpp; //number of bits per pixel (must be 24)
    char temp; //temporary color storage for bgr-rgb conversion

    //make sure the file is here
    file = fopen(filename,"rb");



    //seek through the bmp header, up to width/height
    fseek(file, 18, SEEK_CUR);

    //read the width
    i = fread(&image->sizeX, 4, 1, file);

    //read the height
    i = fread(&image->sizeY, 4, 1, file);
```

```
    //calculate the size of the image in bytes (assuming 24 bits or 3 bytes per pixel)
    size = image->sizeX * image->sizeY * 3;

    //read the planes
    fread(&planes, 2, 1, file);

    //read the bits per pixel
    i = fread(&bpp, 2, 1, file);

    //seek past the rest of the bitmap header
    fseek(file, 24, SEEK_CUR);

    //read the data
    image->data = (char*)malloc(size);

    i = fread(image->data, size, 1, file);

    for(i=0;i<size;i+=3) //reverse all of colors (bgr -> rbg)
    {
        temp = image->data[i];
        image->data[i] = image->data[i+2];
        image->data[i+2] = temp;
    }

    return 1;
}

struct Image *loadTexture(char *filename)
{


    // Allocate space for texture
     struct Image *image = (struct Image*)malloc(sizeof(struct Image));


    // Image not loaded
    if (!ImageLoad(filename, image))
    {
        exit(1);
    }

    return image;
}
```

```
void drawDoor(){

/*


  //texre door==================================================


    //door
    glEnable(GL_TEXTURE_2D);
    glBindTexture(GL_TEXTURE_2D, texture[1]); // Bind the door texture

    glBegin(GL_POLYGON);
    glColor3f(1.0f, 1.0f, 1.0f); // Set a white tint for the door

    glTexCoord2f(0.0f, 0.0f); // Texture coordinate for the bottom-left corner
    glVertex2f(-0.05f, -0.2f);

    glTexCoord2f(1.0f, 0.0f); // Texture coordinate for the bottom-right corner
    glVertex2f(0.07f, -0.2f);

    glTexCoord2f(1.0f, 1.0f); // Texture coordinate for the top-right corner
    glVertex2f(0.07f, 0.1f);

    glTexCoord2f(0.0f, 1.0f); // Texture coordinate for the top-left corner
    glVertex2f(-0.05f, 0.1f);

    glEnd();

glDisable(GL_TEXTURE_2D); // Disable texture mapping



//texre door==================================================}
```

```
void drawLeftHouse()

{


/*================================rooof tex============================*/

    glEnable(GL_TEXTURE_2D);

    glBindTexture(GL_TEXTURE_2D, texture[0]);

    glBegin(GL_POLYGON);

    glTexCoord2f(1.0f, 0.0f); // Top-right corner of the texture
    glVertex2f(0.2f, 0.2f);

    glTexCoord2f(0.0f, 0.0f); // Top-left corner of the texture
    glVertex2f(-0.22f, 0.2f);

    glTexCoord2f(0.0f, 1.0f); // Bottom-left corner of the texture
    glVertex2f(-0.29f, 0.5f);

    glTexCoord2f(1.0f, 1.0f); // Bottom-right corner of the texture
    glVertex2f(0.13f, 0.5f);

    glEnd();

    glDisable(GL_TEXTURE_2D);

/*================================rooof tex============================*/

drawDoor();


}
```

```
void drawVillage() {

    //houseRight
    glPushMatrix();
    glTranslatef(-0.27f,0.0f,0.0f);
    drawLeftHouse();
    glPopMatrix();

}

void display(){
    glLoadIdentity();

    glEnable(GL_BLEND);
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    glClearColor(0.0,0.0,0.0,0.0);
    glClear(GL_COLOR_BUFFER_BIT);
    drawVillage();
    glutSwapBuffers();

}


void init()
{
// Load textures
    struct    Image *roofTexture = loadTexture("roof.bmp");
    struct  Image *doorTexture = loadTexture("door.bmp");

    // Generate OpenGL texture IDs
    glGenTextures(3, texture); // Increment the size to 3

    // Bind and set up the first texture for roof
    glBindTexture(GL_TEXTURE_2D, texture[0]);
    glTexImage2D(GL_TEXTURE_2D, 0, 3, roofTexture->sizeX, roofTexture->sizeY, 0, GL_RGB,
GL_UNSIGNED_BYTE, roofTexture->data);
    glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_LINEAR);
    glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);

    // Bind and set up the second texture for door
    glBindTexture(GL_TEXTURE_2D, texture[1]);
    glTexImage2D(GL_TEXTURE_2D, 0, 3, doorTexture->sizeX, doorTexture->sizeY, 0,
GL_RGB, GL_UNSIGNED_BYTE, doorTexture->data);
```

```
    glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_LINEAR);
    glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);



    // You can add more textures here in the same way

    // Clean up loaded images
    free(roofTexture->data);
    free(roofTexture);
    free(doorTexture->data);
    free(doorTexture);
}



int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGBA);
    glutInitWindowSize(779, 468);
    glutCreateWindow("Assignment-2");
    init();//new
    glutDisplayFunc(display);
    glutKeyboardFunc(keyboard);
    glutMainLoop();

    return 0;
}
```