Can BEYAZNAR

161044038

CSE461 ASSIGNMENT 1

Bu ödevde, **"GL/glut.h"** kütüphanesi kullanılarak 3 boyutlu bir ev tasarlandı.Uygulama, Windows işletim sisteminde yazıldı ve CodeBlocks IDE'si kullanıldı. Windows için OpenGL alt kısımda bulunan bağlantıdaki gibi kurulum yapılmıştır:

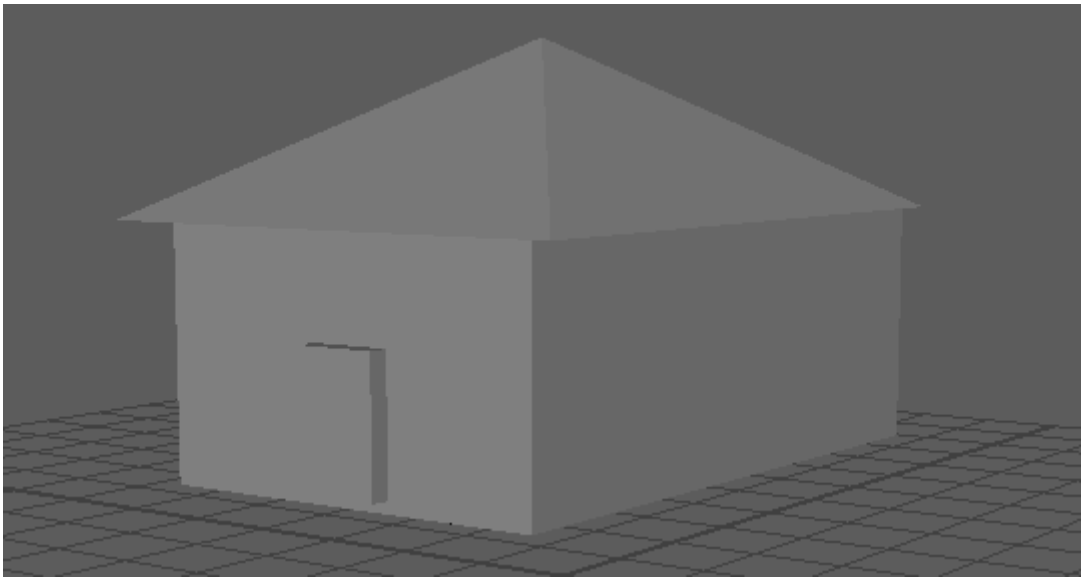
https://www.youtube.com/watch?v=pak-Z6uQjAQ

OpenGL ile yapılan ev iki katlıdır. Birinci katın ön duvarında kapı, yan duvarlarında ise ikişer tane pencere bulunmaktadır. İkinci katında ise balkon bulunmaktadır.

DEMO VIDEOSU : https://youtu.be/8_p1x22XPFY

Neler yapıldı ve nasıl yapıldı?

Ev çiziminin tamamında **glVertex3f()** fonksiyonundan yardım alınmıştır. Öncelikle, OpenGL ile ev yaparken yardım almak adına Maya üzerinden basit bir ev çizildi.



Ardından bu evden yardım alarak OpenGL üzerinden ev yapımı işlemleri başladı.

```
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(1000, 500);
    glutInitWindowPosition(50, 50);
    glutCreateWindow("161044038_HW1");
    glutDisplayFunc(display_screen);
    glutReshapeFunc(reshape);
    glutKeyboardFunc(get_user_input);
    initialize_env();
    glutMainLoop();
    return 0;
}
```

Öncelikle main kısmında, yapılan evin ekranda gözükebilmesi ve kullanıcıdan gelen girdileri klavyeden alıp uygulamaya yansıtmak için gerekli fonksiyon atamaları yapıldı. Evin ekranda gözükmesi ve oluşturulması için **display_screen()**, yeniden boyutlandırmak için **reshape()**, kullanıcıdan girdi alıp işlem yapmak için **get_user_input()** ve son olarak çevrenin oluşturulması için **initialize_env()** fonksiyonları implement edilmiştir. Fonksiyonların içinde bulunan kodu anlatmak gerekir ise:

1) **Display_screen()**

```
void display_screen()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    draw_ilk_kat();
    draw_kapi();
    draw_pencere();
    draw_ust_kat();
    draw_balkon();
    draw_cati();
    draw_baca();

    glFlush ();
    glPopMatrix();
    glutSwapBuffers();
}
```

Bu fonksiyonda evin oluşturulması için gerekli fonksiyonlar bulunmaktadır. Kodun okunabilirliğini arttırmak ve hata olduğunda rahat bir şekilde düzeltemk adına, her bir nesnenin çizimi ayrı fonksiyonlara atanmıştır. Örneğin çatıyı

çizen kısım **draw_cati()** fonksiyonu, balkonu çizen kısım ise **draw_balkon()** fonksiyonunda bulunmaktadır. Çizim yapılırken tüm nesnelerde **glVertex3f()** fonksiyonu kullanılmıştır. Nesnelerin şekillerinin oluşturulabilmesi için sadece çatı kısmında **GL_TRIANGLES,** geriye kalan tüm bölgeler için **GL_QUADS** kullanılmıştır. Her bir nesnenin detaylı bölgeleri kodun yorum satırlarında bulunmaktadır.

2) Reshape()

```
void reshape(int w, int h)
{
    if (h == 0)
        h = 1;
    GLfloat aspect = (GLfloat)w / (GLfloat)h;
    glViewport(0, 0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(45.0f, aspect, 0.1f, 100.0f);
}
```

Bu fonksiyon nesnelerin yeniden boyutlandırılması için kullanılmaktadır. OpenGL fonksiyonları kullanılarak, nesnelerin yeniden boyutlandırılmasına yardımcı oluyor.

3) Get_user_input()

```
{
    // Ileri hareket et
    if(key == 'w')
        glTranslatef(0.0, 0.0, 10.0);
    // Sola hareket et
    else if(key == 'a')
        glTranslatef(-10.0, 0.0, 0.0);
    // Geri hareket et
    else if(key == 's')
       glTranslatef(0.0, 0.0, -10.0);
    // Saga hareket et
    else if(key == 'd')
        glTranslatef(10.0, 0.0, 0.0);
    // Yukari hareket et
    else if(key == 'r')
        glTranslatef(0.0, -10.0, 0.0);
    // Asagi hareket et
    else if(key == 'f')
        glTranslatef(0.0, 10.0, 0.0);
    // Rotate
    //Sola don X eksen
    else if(key == 'y')
        glRotatef(10.0, 30.0, 0.0, 0.0);
    // Saga don X eksen
    else if(key == 'u')
        glRotatef(10.0, -30.0, 0.0, 0.0);
    // Sola don Y eksen
    else if(key == 'h')
        glRotatef(10.0, 0.0, 30.0, 0.0);
    // Saga don Y eksen
    else if(key == 'j')
        glRotatef(10.0, 0.0, -30.0, 0.0);
     // Sola don Z eksen
    else if(key == 'b')
        glRotatef(10.0, 0.0, 0.0, 30.0);
    // Saga don Z eksen
    else if(key == 'n')
        glRotatef(10.0, 0.0, 0.0, -30.0);
    // Buyut
    else if(key == '1')
        glScalef(2, 2,2);
    // Kucult
    else if(key == '2')
        glScalef(0.5, 0.5, 0.5);
    display_screen();
```

Bu fonksiyonda ise, kullanıcıdan gelen girdilere göre nesneye ödevde belirtildiği gibi translation, rotation ve scaling işlemleri yapılabilmektedir. Bunları yapmak için **glTranslatef(), glRotatef(), glScalef()** olan OpenGL fonksiyonları kullanılmıştır. Ardından ne güncellenmesi için **display_screen()**

fonksiyonu tekrardan çağırılmıştır. Bu işlemlere atanan tuşları detaylı belirtmek gerekirse:

- **'w', 'a', 's', 'd', nesneyi ileri, sola, geriye ve sağa translate etmeyi sağlar.**
- **'r' ve 'f', nesneyi yukarı ve aşağı translate etmeyi sağlar.**
- **'y' ve 'u', nesneyi X ekseninde saat yönü ve saat yönünün tersinde döndürmeyi sağlar.**
- **'h' ve 'j', nesneyi Y ekseninde saat yönü ve saat yönünün tersinde döndürmeyi sağlar.**
- **'b' ve 'n', nesneyi Z ekseninde saat yönü ve saat yönünün tersinde döndürmeyi sağlar.**
- **'1' ve '2', nesneyi büyültüp küçülterek scale etmeyi sağlar.**

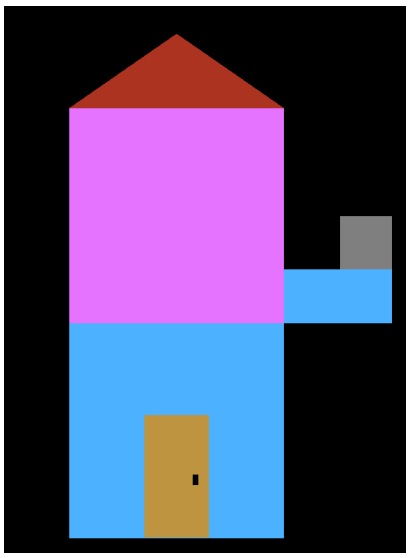
NOT: Klavye ile uygulamaya girdi verirken, CAPS LOCK açık olmadığından emin olunuz. Yalnızca küçük harfler üzerinden girilen girdilere karşılık verilmektedir.

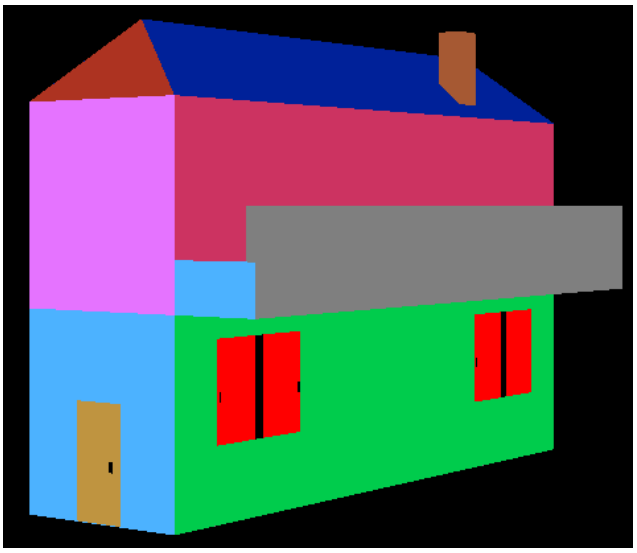
4) İnitialize_env()

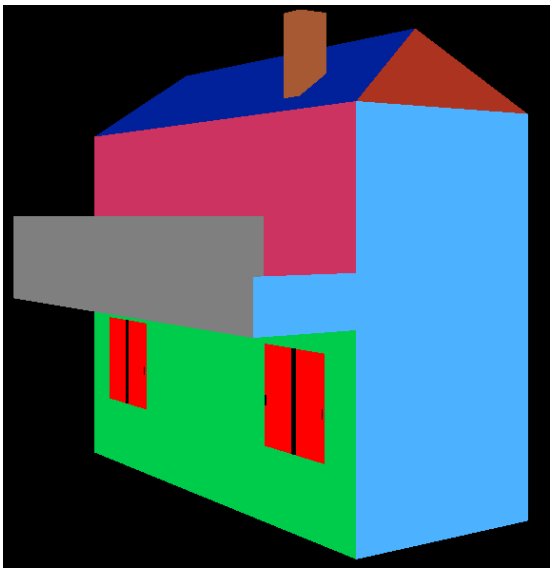
```
void initialize_env()
{
    glMatrixMode(GL_MODELVIEW);
    glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
    glClearDepth(1.0f);
    glEnable(GL_DEPTH_TEST);
    glDepthFunc(GL_LEQUAL);
    glShadeModel(GL_SMOOTH);
    glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
}
```

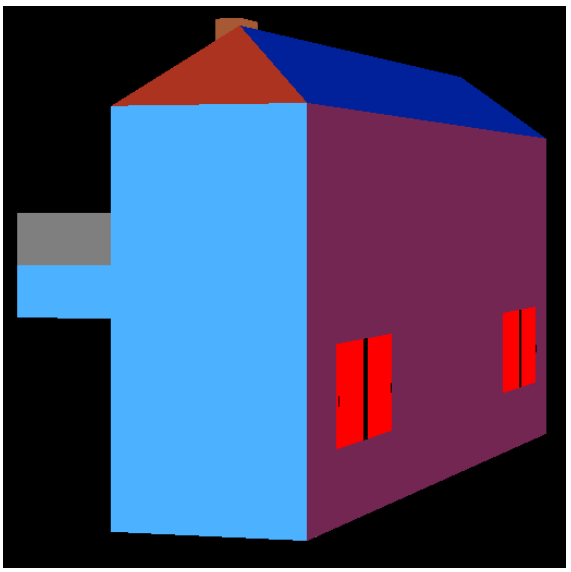
Bu fonksiyonda, uygulamamızda gerekli olan OpenGL fonksiyonlarını kullanarak çevre ayarlamaları yapılır. Evin renklerinin belli olması için etraf siyah renkli yapılmıştır.

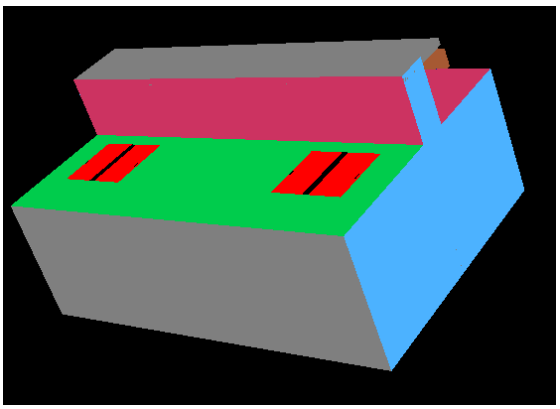
Son Çıktı





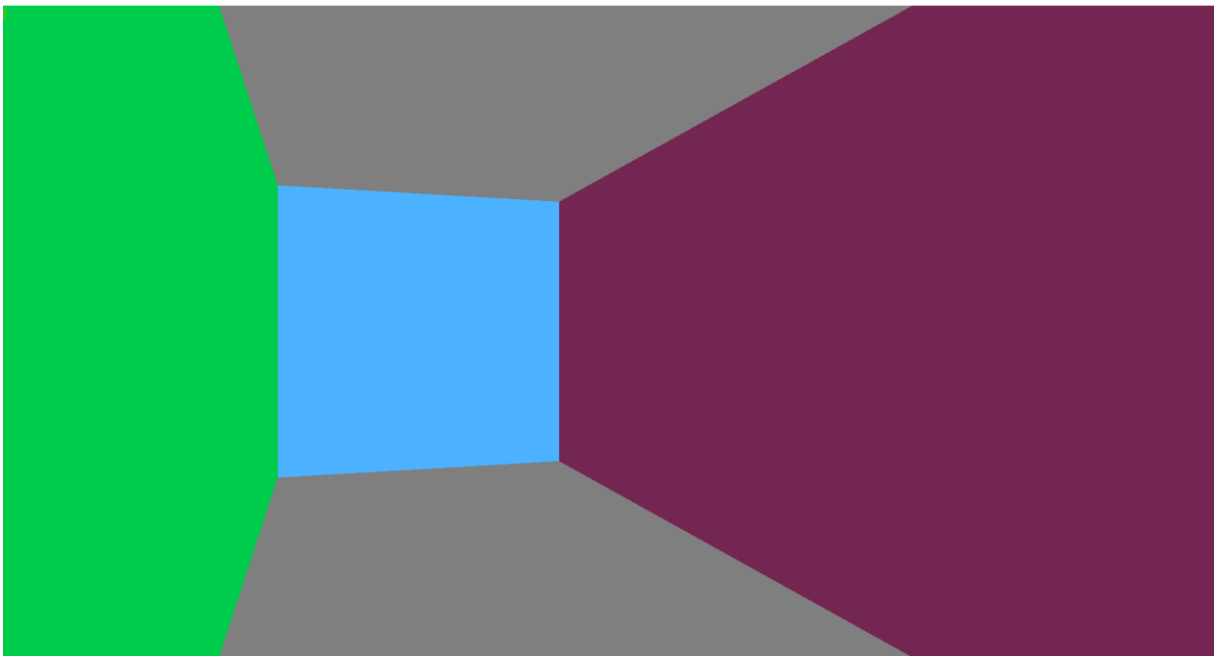






Evin içi :

Kat 1 :



Kat 2 :