,

- OpenGL temelli grafik uygulamaları, çok çeşitli sistemler üzerinde koşulabilir. (tüketici elektroniği – consumer electronics, PC, iş istasyonu – workstation, süper bilgisayarlar gibi)
- OpenGL grafik kütüphanesi kullanılarak, çok daha az bir kod satırıyla daha yüksek performansa sahip uygulamalar geliştirmek mümkündür.
- OpenGL grafik kütüphanesine dair teknik bilgi içeren birçok kaynak mevcuttur.(internet, kitaplar, vs.)
- Birçok programlama dili (C, C++, Fortran, Ada, Java gibi) OpenGL tabanlı uygulama geliştirmemize olanak sağlar.

2.2.Open GL Utility (GLUT)

OpenGL platformdan bağımsız olduğu için bazı işlemler bu kitaplık ile yapılamaz. Örneğin kullanıcıdan veri almak, bir pencere çizdirmek gibi işler hep kullanılan pencere yöneticisi ve işletim sistemine bağlıdır. Bu yüzden bir an için OpenGL'in platform bağımlı olduğu düşünülebilir. Çünkü çalışma penceresini her pencere yöneticisinde (her ortamda) farklı çizdirecek bir canlandırma programı yazmak demek her bilgisayarda çalışacak ayrı pencere açma kodu yazmak demektir. Bu ise OpenGL'in doğasına aykırıdır. Bu gibi sorunları aşmak için OpenGL Araç Kiti (GLUT - OpenGL Utility Toolkit) kullanılmaktadır.

GLUT, birçok işletim sistemine aktarılmış bir kitaplıktır. Amacı OpenGL programlarının pencerelerini oluşturmak, klavye ve fareden veri almak gibi ihtiyaçları karşılamaktır.

GLUT olmadan da OpenGL programlama yapılabilir, örneğin Linux'ta kullanılan X-Window sistemin kendi işlevleri kullanılarak pencere çizdirilebilir fakat bu kod sadece X-Window'da çalışır. Kod Windows'a götürülüp derlendiğinde çalışmaz, çünkü Windows'da X-Window işlevleri yoktur. Benzer şekilde Windows tabanlı işletim sistemlerinin de kendilerine has pencere oluşturma işlevleri vardır.

Bu yüzden bu deneyde GLUT kitaplığı kullanılarak klavye ve fare için işletim sisteminden bağımsız giris/çıkış işlemleri yapılması sağlanmıştır.

2.3.OpenGL Söz dizimi

Opengl komutları, gl öneki ile başlarlar. (örnek; glClearColor()). Benzer şekilde OpenGL tarafından tanımlı sabitler de GL_ öneki ile başlarlar ve kelimeler birbirinden _ ile ayrılacak şekilde büyük harfle yazılırlar. (örnek; GL_COLOR_BUFFER_BIT).

glColor3f komutundaki 3 sayısı da 3 parametre alacağı anlamına gelmektedir. f ise verilen parametrelerin float olacağı anlamına gelmektedir.

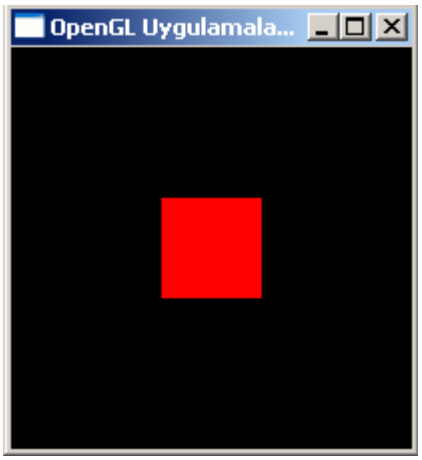
```
glVertex2i(1,3);   ya da
glVertex2f(1.0,3.0);   gibi
```

2.4.İlk OpenGL Programı

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

void ayarlar(void)
{
   glClearColor(0.0,0.0,0.0,0.0);
```

```
    glShadeModel(GL_FLAT);
    glOrtho(-2.0, 2.0, -2.0, 2.0, -1.0, 1.0);

}

void gosterim(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(1.0, 0.0, 0.0);

    glBegin(GL_POLYGON);
        glVertex2f(-0.5, -0.5);
        glVertex2f(-0.5, 0.5);
        glVertex2f(0.5, 0.5);
        glVertex2f(0.5, -0.5);
    glEnd();
    glFlush();
}

int main(int argc,char ** argv)
{
   glutInit(&argc,argv);
   glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB );
   glutInitWindowPosition(0,0);
   glutInitWindowSize(200,200);
   glutCreateWindow("OpenGL Uygulamaları-I");
   ayarlar();
   glutDisplayFunc(gosterim);
   glutMainLoop();
   return 0;
}
```

Programda ilk olarak bir pencere yaratılmakta daha sonra da gösterim fonksiyonu ayarlanmaktadır. Gösterim fonksiyonu içerisinde de her defasında çizilecek olan grafik çizilmektedir. Bu hali ile verilen kod basit bir openGL programının iskeletini oluşturmaktadır. Program çalıştırıldığında elde edilen ekran görüntüsü Şekil 1'de verilmiştir.



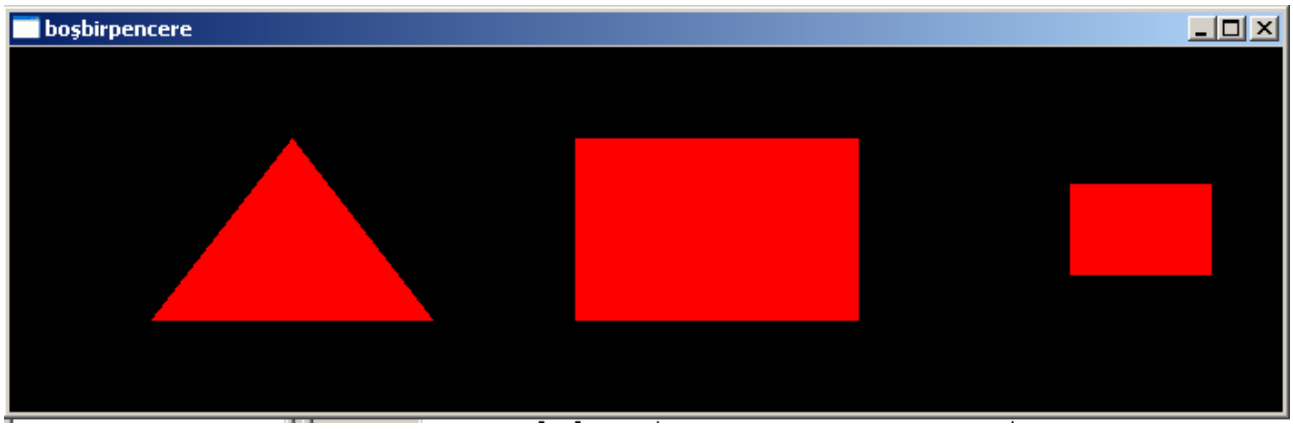
Şekil 1. OpenGL ile çizilmiş basit bir şekil

2.5.OpenGL ile Birden Fazla Şekil Çizimi

Birden fazla şekil çizmek için yukarıda verilen iskelet program üzerinde sadece gosterim isimli fonksiyon aşağıdaki şekilde değiştirilir.

```
void gosterim(void)
{
    glLoadIdentity();
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(1.0, 0.0, 0.0);
    glOrtho(-2.0, 7.0, -2.0, 2.0, -1.0, 1.0);
    glBegin(GL_TRIANGLES);
            glVertex3f( 0.0f, 1.0f, 0.0f);
            glVertex3f(-1.0f,-1.0f, 0.0f);
            glVertex3f( 1.0f,-1.0f, 0.0f);
    glEnd();

    glTranslatef(3.0f,0.0f,0.0f);
    glBegin(GL_QUADS);
            glVertex3f(-1.0f, 1.0f, 0.0f);
            glVertex3f( 1.0f, 1.0f, 0.0f);
            glVertex3f( 1.0f,-1.0f, 0.0f);
            glVertex3f(-1.0f,-1.0f, 0.0f);
   glEnd();

    glTranslatef(3.0f,0.0f,0.0f);
    glBegin(GL_POLYGON);
       glVertex2f(-0.5, -0.5);
       glVertex2f(-0.5, 0.5);
       glVertex2f(0.5, 0.5);
       glVertex2f(0.5, -0.5);
    glEnd();
    glFlush();

}
```

Verilen programın ekran görüntüsü Şekil 2'de verilmiştir.



Şekil 2. OpenGL ile birden çok şekil çizmek

,

Tartışma Sorusu-1 : Gösterim fonksiyonunda kullanılan *glLoadIdentity();* fonksiyonunun işlevi nedir? Neden kullanılmıştır? Bu fonksiyon kullanılmasaydı ekran görüntüsü nasıl olurdu? Tartışınız…

2.6.OpenGL ile Farklı Renklerle Çalışmak

Verilen taslak program üzerinde gösterim fonksiyonu aşağıdaki gibi değiştirilerek çizilen şekiller

3. Deneye Hazırlık

Bu bölüm, deneye gelmeden önce her öğrenci tarafından yapılması gereken maddeleri içermektedir.

1. Deneye gelmeden önce Dev C++ programı bilgisayara kurulmalı ve temel düzeyde programın kullanılışı öğrenilmelidir. Programı şu adresten (http://www.bloodshed.net/download.html) indirilebilir.
2. Ek-1'de verilen adımlar takip edilerek Dev C++ programına OpenGL ve GLUT kütüphaneleri kurulmalıdır.
3. Deney föyü dikkatlice okunmalı ve Deneye hazırlık soruları cevaplanmalıdır.
4. Ek-2'de verilen örnek program yazılarak koşulmalı ve programın çalışma mantığı anlaşılmalıdır.

4. Deneyin Yapılışı

5. Deney Soruları

6. Kaynakça

Addison Wesley, "OpenGL Programming Guide", 6th Edition, 2008.

Yapıacaklar

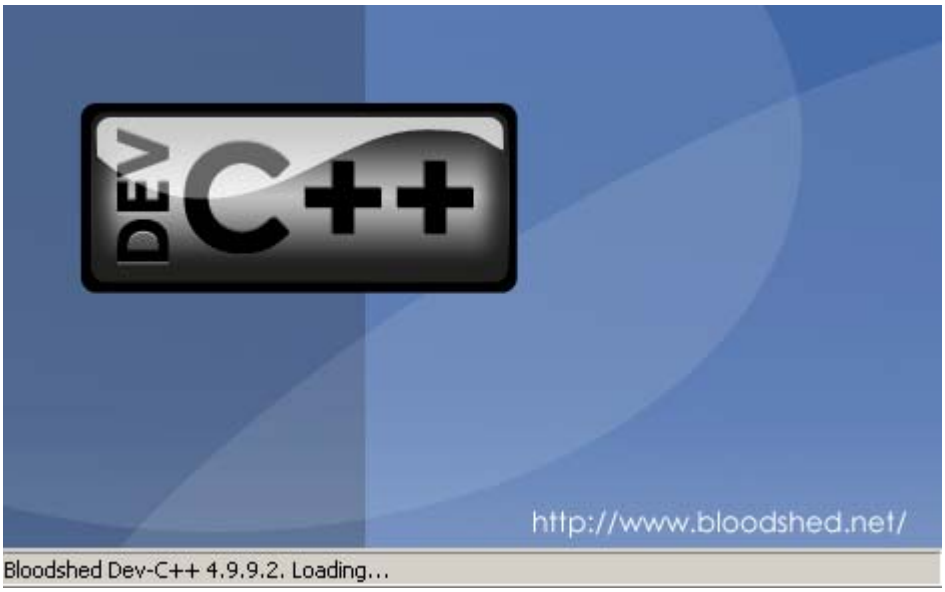
- Ek 1 de dev c++ a glut un kurulusu anlatılacak
- Ek-2 de tüm yazılan kodlar anlatılacak.
- Deneyin yapılışı adım adım anlatılacak
- Deney soruları verilecek

,

7. Ek 1

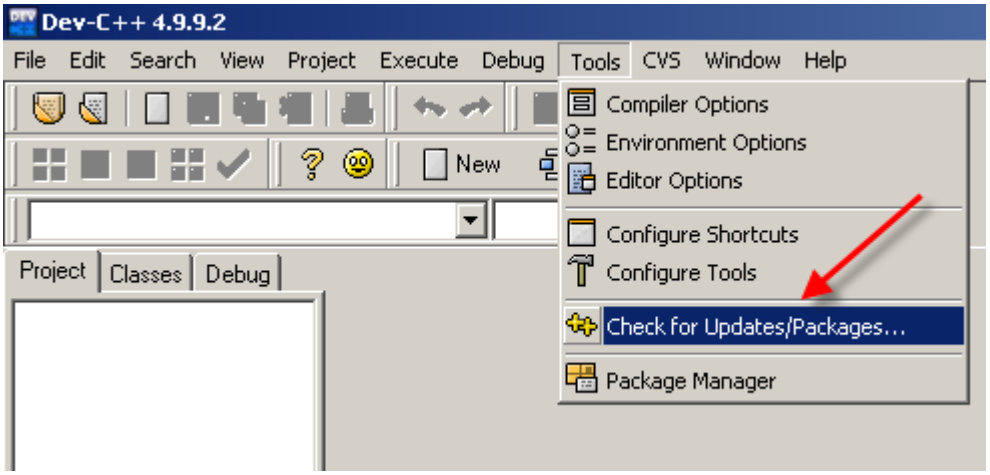
7.1.Adım 1

Dev C++ programını kurun



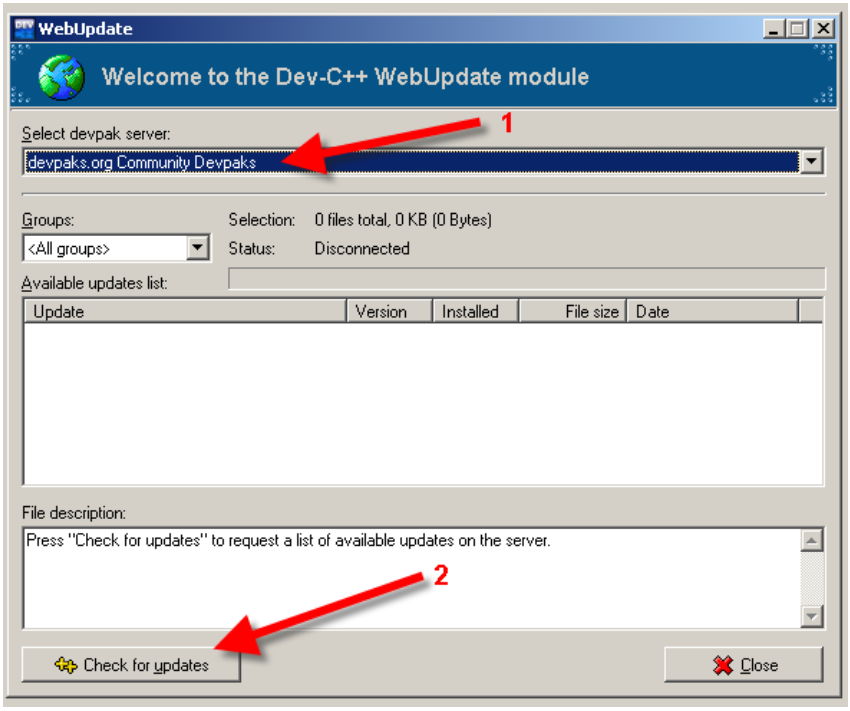
7.2.Adım 2

Tools / Check for Updates/Packages…



,

7.3.Adım 3



7.4.Adım 4