**MATEMATİK ÇARKI**

160201079 İlknur GÖK,

160201012 Elanur OCAK

Bilgisayar Mühendisliği Bölümü

Kocaeli Üniversitesi

[ilknurgok.1@hotmail.com](mailto:ilknurgok.1@hotmail.com) , [elaocak45@gmail.com](mailto:elaocak45@gmail.com)

**Özet**

Projeye başlarken ilk aşamada hazırladığımız cark() fonksiyonunun kaç kez döneceğini kullanıcıdan istiyoruz ve fonksiyonda rastgele verilen sayıların en sonuncusunun dörde göre modunu aldırıyoruz. Moda göre çıkan sayı hangi işlemle devam edeceğimizi belirliyor.

Sayı 1 ise, önce kullanıcıdan matris boyutunu ve matrisi aldırıp aldırdığımız matrisi ozdegerbul() fonksiyonuna gönderiyor ve bu fonksiyonla matrisin öz değerlerini buluyoruz.

Sayı 2 ise, schur() fonksiyonuyla kullanıcıdan matris boyutu ve matris aldırıp matrisi ozdegerbul() fonksiyonuna gönderiyoruz ve bu fonksiyondan öz değerleri geri çağırıyoruz. Çağırdığımız öz değerlerle schur() fonksiyonunda sınır değerlerini elde ediyoruz .

Sayı 3 ise, ozvektorbul() fonksiyonuyla matrisin öz vektörlerini hesaplıyoruz (öz vektörleri hesaplarken de ozdegerbul() fonksiyonundan yararlanıyoruz.) ve sayı 0 ise, nilpotent() fonksiyonu kullanıcıdan aldığımız matris boyutu ve bu boyuta göre rastgele aldığımız kare matrisin nilpotent olup olmadığını (herhangi bir üssünün sıfır matrise eşit olup olmadığını) buluyoruz. Yaptırdığımız bu işlemlerle polinom denklemleri ve birkaç tane ara matris elde ediyoruz. Bu dört fonksiyondan elde ettiğimiz ara matrisleri ve değerleri ozdeger.txt, schur.txt, ozvektor.txt ve nilpotent.txt dosyalarına aktarıyoruz.

**1.Giriş**

Projenin genel olarak konusu, yazılan kod ile oluşturulan çarkın dönmesiyle çıkan sayının moduna göre projenin bazı işlemleri gerçekleştirip dosyaya atamasıdır.

Hazırladığımız projenin amacı, elimizdeki matrisi öz değer, schur, öz vektör ve nilpotent teoremleriyle kullanıcının istediği sonuca ulaştırmaktır.

Matrisin öz değerlerini ve öz vektörlerini açıklayacak olursak; bir A(nxn) matrisin öz değer problemi lineer sistemi

**X = 0** dışında çözümlere sahip olacak şekilde    
sayılarının belirlenmesini gerektirir.

Sonuç olarak temel denklem

Ax = λx (1)

şeklindedir. Burada λ skaleri A matrisinin bir öz değeridir. X skaleri ise A matrisinin bir öz vektörüdür. Bundan dolayı (1) denklemi

**(A-I)x = 0 (2)**

şeklinde yazılabilir. Burada **I** , *nxn* tipinde birim matristir.

Schur Teoremi ise,  tipinde bir matris ve nın öz değerleri de ’dir. Bu durumda sınır değerlerini bulduran denklem,

 (3)

olur.

[8]

Son olarak nilpotent teoremi ise bir kare matrisin herhangi bir üssünün (biz projede matrisin üssünü 100 ile sınırlandırdık.) sıfır matrise eşit olmasıdır.

Örneğin;

A3  = (4)

A nilpotent matristir.

**2.Temel Bilgiler**

Proje C dilinde yazıldı ve CodeBlocks’ta hazırlandı. Dosya oluşturma kısmı Notepad’de tamamlandı.

* Code::Blocks, özgür açık kaynak kodlu bir C++ tümleşik geliştirme ortamıdır. wxWidgets tabanlı tamamen özelleştirilebilir arabirimiyle, GNU/Linux, Microsoft Windows, MacOS platformlarında sorunsuzca kullanılabilmektedir.
* Notepad++, Windows işletim sistemi içerisine gömülü olarak gelen Notepad yazılımının yerine kullanılmak üzere C++ ile saf Win32 API ve STL ile geliştirilmiş GPL ile dağıtılan açık kaynak kodlu bir kaynak kod düzenleyicisidir.

**3.Diğer Bölümler**

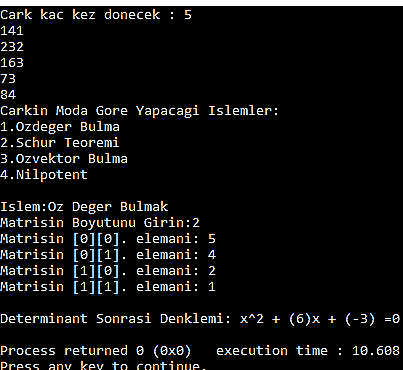
Projede ilk olarak kullanıcıdan çarkın kaç kez döneceği bilgisini alıp cark() fonksiyonuna gönderiyoruz. Gönderdiğimiz sayı kadar 0 ile 241 arası rastgele sayı üretip en son çıkan sonucu main fonksiyonuna gönderiyoruz. Main() fonksiyonunda mod aldırıyoruz ve çıkan moda göre ;

**3.1. Öz değerler**

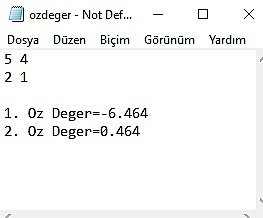
Eğer sonuç 1 çıkarsa kullanıcıdan matris boyutu ve matris alıyoruz. Daha sonra boyutu ve matrisi ozdegerbul() fonksiyonuna gönderiyoruz.  Fonksiyona gönderilen boyut 2 ise, determinant sonrası oluşan polinom denklemini yazdırıyoruz (detAl() fonksiyonundan yararlanıyoruz.).

ax² + bx + c=0 (5)

Daha sonra iki bilinmeyenli denklem çözümü için diskriminantını buluyoruz. Diskriminant 0'dan küçükse kök bulunamadığını belirtiyoruz. Diskriminant 0'a eşit ise denklemin köklerinden bir adet öz değer elde ediyoruz ve 0'dan büyük ise iki adet kök bulup öz değerleri yazdırıyoruz.



Şekil 1:2 Boyutlu Matrisin Öz Değer Çıktısı



Şekil 2:2 Boyutlu Matrisin Öz Değer Text Dosyası

Fonksiyona gönderilen boyut 3 ise, determinant sonrası polinom denklemini yazdırıyoruz.

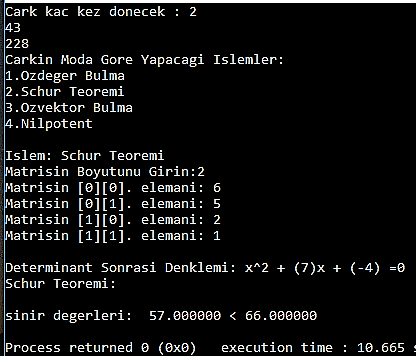
ax3+bx² + cx + d=0 (6)

Daha sonra bir kaynaktan yararlanarak denklemin diskriminantını buluyoruz. Diskriminant 0'dan küçükse denkleme ait kök bulunamadığını , diskriminant 0'a eşitse bir

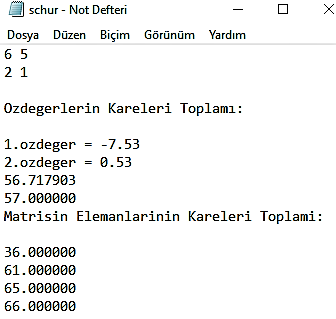
adet öz değer bulunduğunu veya diskriminant 0'dan büyükse üç adet öz değer bulunduğunu belirtiyoruz ve bu öz değerleri yazdırıyoruz.

**3.2. Schur Teoremi**

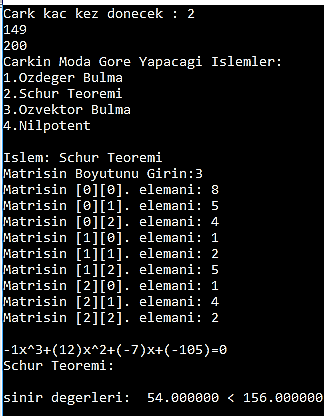
Eğer sonuç 2 çıkarsa main () fonksiyonundan schur() fonksiyonuna geçiyoruz. Kullanıcıdan kare matris boyutunu ve matrisi aldırıyoruz (Eğer kullanıcı matris boyutunu 4'ten büyük girerse 4'ten küçük bir değer girilmesini istiyoruz.) . Sonra matrisi ozdegerbul() fonksiyonuna gönderip girilen matrise ait öz değerleri schur() fonksiyonuna geri çağırıyoruz. Schur teoremini uygulayabilmek için fonksiyona çağırılan öz değerlerin kareleri toplamı buluyoruz, bu bize teorem için birinci sınır değerini veriyor. Daha sonra kullanıcıdan alınan matrisin elemanlarının kareleri toplamını buluyoruz ve bu da bize teorem için ikinci sınır değerini veriyor. Fonksiyonun sonunda da bu iki sınır değerini karşılaştırıyoruz.



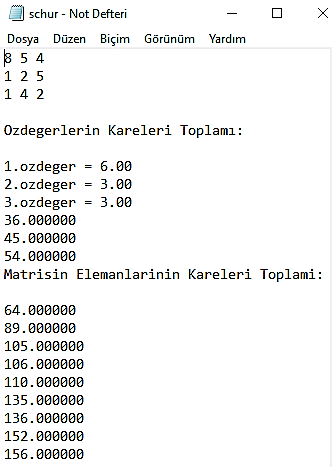
Şekil 3:2 Boyutlu Matrisin Schur Teoremi Çıktısı



Şekil 4:2 Boyutlu Matrisin Schur Teoremi Text Dosyası



Şekil 5:3 Boyutlu Matrisin Schur Teoremi Çıktısı



Şekil 6:3 Boyutlu Matrisin Schur Teoremi Text Dosyası

**3.3. Öz Vektörler**

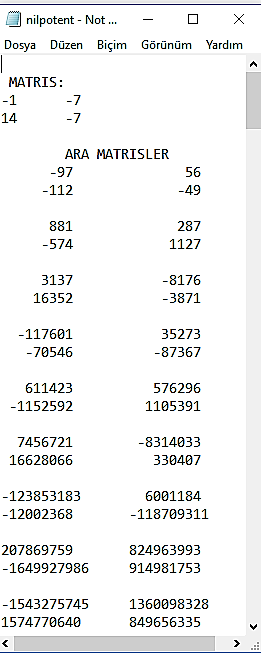
Eğer sonuç 3 çıkarsa main() fonksiyonundan ozvektorbul() fonksiyonuna geçiyoruz. (Fakat bizim projemiz öz vektör için çıktı vermemektedir.)

**3.4. Nilpotent Teoremi**

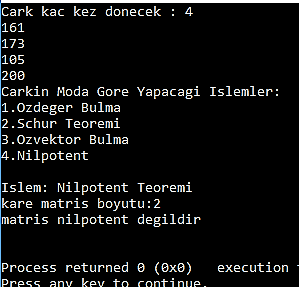
Eğer sonuç 0 çıkarsa main() fonksiyonundan nilpotent() fonksiyonuna geçiyoruz. Kullanıcıdan kare matrisin boyutunu alıyoruz.  Alınan boyuta göre matrisin elemanlarına -15 ile 15 arası rastgele değerler atıyoruz. Oluşan matrisi başka bir matris ile eşitliyoruz.  Kullanıcıdan alınan ve ona eşitlediğimiz matrisi birbiri ile çarptırıp yeni bir çarpım matrisi oluşturuyoruz. Çarpım sonucunda oluşan bu yeni matrisi, kullanıcıdan aldığımız matris ile çarptığımız tekrar o matrise eşitliyoruz (Döngü kısmında oluşan matrisin kaybolmaması için yapılıyor bu işlem.). Sonrasında çarpım matrisinin bir sıfır matris olup olmadığını kontrol ediyoruz.

Eğer çarpım matrisi sıfır matrise eşitse , bu matrisin nilpotent matris olduğunu belirtiyoruz ve döngüden çıkış yapıyoruz. Eğer çarpım matrisi sıfır matrise eşit değil ise, bu çarpım matrisi fonksiyonda tekrar çarpma işlemine alıyoruz ve bu çarpım işlemi sonucunda çıkan matris sıfır matrise eşit olmadığı sürece döngü yardımıyla 100 kez işlemi tekrarlıyoruz. Bu tekrarlar sonucunda matris hala sıfır matrise eşit olmaz ise ekrana nilpotent matris olmadığını yazdırıyoruz.

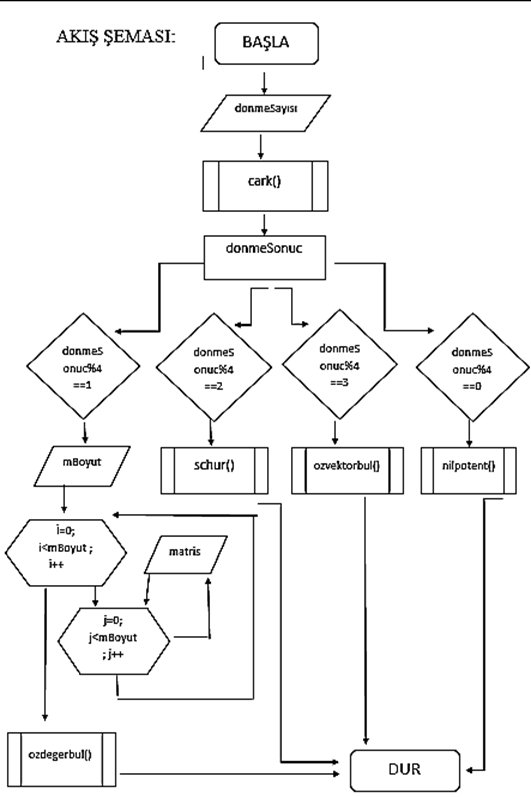
Son olarak fonksiyonlarda işlem yapabilmek için kullandığımız ara matrisleri ve bulduğumuz öz değerleri text dosyalarına (ozdeger.txt, schur.txt, ozvektor.txt, nilpotent.txt) aktarıyoruz.



Şekil 7:2 Boyutlu Matrisin Nilpotent Teoremi Text Dosyası



Şekil 8:2 Boyutlu Matrisin Nilpotent Teoremi Çıktısı

****

Şekil 9:Projenin Akış Şeması

**4. Sonuçlar**

Çalışmamızda genel olarak lineer cebirsel ve matematiksel işlemlere dayalı formülleri veya teoremleri kullanmamız gerekiyordu. Bizden istenilenlere ek olarak determinant almak gibi fonksiyonlar kullandık.

Projenin ilk kısmında bizden cark() fonksiyonu yardımıyla rastgele sayılar almamız ve bu sayıya göre işlemi belirlememiz istendi. Bunu sorunsuz çalıştırdık.

Projemizin ikinci kısmında bizden bir boyutu ve matris almamız ve bu matrisin öz değerlerini bulmamız istendi. Biz bu matrisin 2. ve 3. dereceden öz değerlerini bulabildik fakat 4 ve üzeri derecelerini bulamadık.

Üçüncü kısımda bizden bir matris boyutu ve matris almamız istendi ve bu değerlere göre matrisin öz değerleri de kullanarak Schur teoremiyle sınır değerlerinin bulunması istendi. 2. ve 3. dereceden matrislerin Schur teoremine göre sınır değerlerini bulduk fakat yine 4 ve üzeri derecelerin sınır değerlerini elde edemedik.

Dördüncü kısımda bizden matrisin öz vektörlerinin bulunması istendi. Fakat elimizdeki matematiksel formülü koda dökemedik. Dolayısıyla öz vektörler elde edemedik.

Son olarak beşinci kısımda ise bizden bir matris boyutu ve boyuta göre rastgele matris atamamız istendi. Bu matrisin herhangi n. dereceden katının sıfır matrise eşit olup olmadığını (nilpotent olup olmadığı) bulmamız gerekiyordu. Biz bunu 100 kere ile sınırlandırıp nilpotent matrise eşit olup olmadığını bulduk.

Projenin en sonunda elde ettiğimiz ara matrisler, öz değerler, sınır değerleri vs. dosyalama işlemleriyle birlikte text dosyalarına atamamız istendi. Bunu sorunsuz çalıştırdık.

**5.Kaynakça**

[1]. Türker Ulutaş, Yücel ve Ömür, Neşe . “Lineer Cebir ve Çözümlü Problemler”, Umuttepe Yayınları, İstanbul, 3. Baskı, Ekim 2015.

[2]. Web Site <http://www.matematikdunyasi.org/arsiv/PDF_eskisayilar/1994_3_9_13_CEBIRSEL.pdf>

[3]. Web Site <http://www.mmsrn.com/3-dereceden-denklem-kokleri-bulan-c-programi-kodlari/>

[4]. Web Site <https://gist.github.com/mertyildiran/8054471>

[5]. Web Site <https://stackoverflow.com/questions/2425167/use-of-exit-function>

[6]. Web Site <http://bilgisayarkavramlari.sadievrenseker.com/2008/11/19/matrisin-tersinin-alinmasi-mantrix-inverse/>

[7]. Yorulmaz, Muhammet ve Yorulmaz, Seher . “Programlamayı C ile Öğreniyorum”, Palme Yayıncılık, Ankara, 6. Baskı, 2016.

[8].Web Site www.yildiz.edu.tr/~nguzel/1\_OZDEGERLER\_VE\_OZVEKTORLER\_Giris.doc