BEYKENT ÜNİVERSİTESİ

FEN-EDEBİYAT FAKÜLTESİ

MATEMATİK BÖLÜMÜ

KUADRİKLER VE PYTHON

HAZIRLAYANLAR

2001006055

ŞİLAN ABA

2001006062

BUSE ŞEVVAL AL

2001006015

ŞEVVAL BEGÜM ASLAN

2001006008

SEZAİ DAĞHAN

2001006007

SENA KAYATEPE

DANIŞMAN

Dr. Öğr. Üyesi MÜCAHİT AKBIYIK

İSTANBUL 2024

İÇİNDEKİLER

Teşekkür

Bu çalışma,İstanbul Beykent Üniversitesi Fen Edebiyat Fakültesi Matematik Bölümü Lisans programı içerisinde yer alan Bitirme Projesi kapsamında hazırlanmıştır.Çalışma süresince desteği,yol göstericiliği ve yönlendirmeleriyle bize rehberlik eden,yeni görüş ve fikirler edinmemizi sağlayan Sn.Dr.Öğr Üyesi Mücahit Akbıyık'a,desteklerini esirgemeyen bölüm başkanımız Sn Doç.Dr Fatma Karaca'ya,ayrıca bize her konuda destek olan ailemize teşekkürlerimizi sunarız.

KUADRATİK YÜZEYLER

$\ddot{O}klid\ uzayında\ (x, y, z)$ koordiatları arasında bir $F(x, y, z) = 0$ eşitliğini sağlayan noktalar kümesi ele alalım. Eğer yüzeyi tanımlayan bu F bağıntısı (x, y, z) değişkenlerine göre ikinci dereceden bir denklem ise elde edilen bu yüzeye **kuadratik yüzey** denir.

Silindirik yüzeyler, elipsoidler ve bunların bir özel hali olan küre yüzeyi, koni yüzeyleri, paraboloidler ve hiperparaboloiler temel kuadratik yüzey örnekleridir.

KUADRATİK YÜZEY ÇEŞİTLERİ

- Küre
- Elipsoid
- Hiperboloid
- Paraboloid
- Silindir

1.1 KÜRE

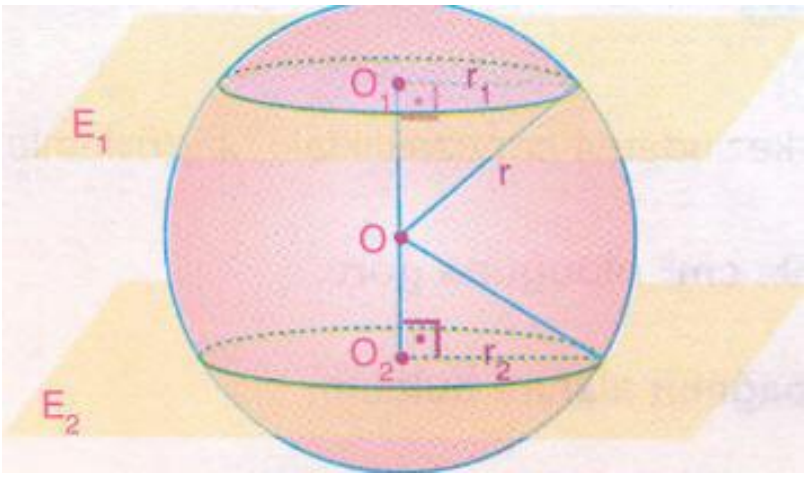
Küre kusursuz simetriye sahip geometrik bir nesnedir.

Uzayda sabit $bir\ M$ noktasıdan $r\ uzaklıktaki$ noktaların geometrik yerine **küre** denir.[35]

Yüzü olmadığı gibi hiçbir ayrıtı ve köşesi de bulunmayan bir geometrik şekildir. Küre'nin sadece bir eğri yüzeyi bulunmaktadır.

Analitik geometride XYZ dik koordinat düzleminde, $merkezi\ 0(0,0,0)$ ve yarıçapı r olan küre yüzeyinin denklemi,

$x^2 + y^2 + z^2 = r^2$ dir.[35]



İki nokta arasındaki uzaklık formülünden, küre yüzeyinin denklemi,

- $(x-a)^2+(y-b)^2+(z-c)^2=r^2$ olur.
- $x^2+y^2+z^2+Dx+Ey+Fz+G=0$

Örnek :

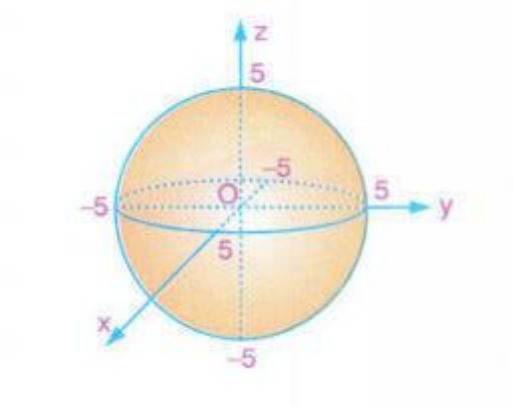
Merkezi 0(0, 0, 0) ve yarıçapı 5 birim olan kürenin denklemini yazalım. Eksenleri kestiği noktaları bulalım.

$x^2+y^2+z^2=5^2$

$x^2+y^2+z^2=25$

Küre yüzeyinin eksenleri kestiği noktaların koordinatları,

$(5,0,0),(-5,0,0),(0,5,0),(0,-5,0),(0,0,5),(0,0,-5)$olur.



Parametrik denklem

X=r.sinu.cosv

Y=r.sinu.sinv

Z=r.cosu

ile bulunur.

$Q(u,v)=(Rcosucosv,Rcosusinv,Rsinu)$ olarak verebiliriz.

Örnek:

$(x-1)^2+(y-2)^2+(z-3)^2=4$ küre denklemini parametrik olarak yazınız.

$x-1=2sinu$

$y-2=2cosusinv$

$z-3=2cosucosv$

$4sin^2u+4cos^2usin^2v+4cos^2ucos^2v=4$

$4sin^2u + 4cos^2u(sin^2v + cos^2v) = 4\,,\quad 4 = 4$

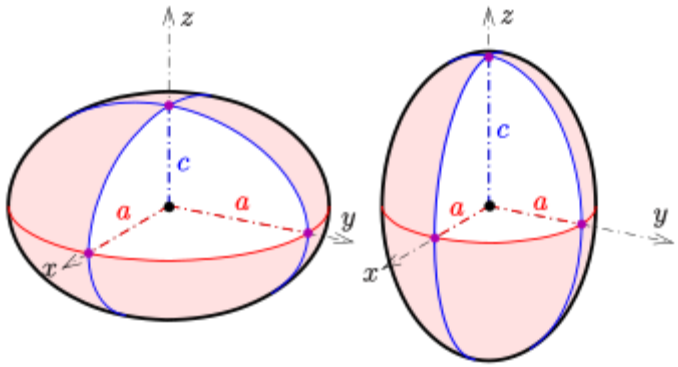
1.2 ELİPSOİD

- Bir elipsoid merkez noktasına, koordinat eksenlerine ve koordinat düzlemlerine göre simetriktir.
- Bir elipsoidde a,b ve c sayılarından ikisi eşit ve üçüncüsü bunlardan farklı ise yüzeye **dönel elipsoid** veya **spheroid** adı verilir. Bu yüzey bir elipsi uzunluğu farklı olan ekseni etrafına döndürülerek elde edilir.
- Bir elipsoidde a=b=c=r se yüzey **küre yüzeyi**dir. Spheroid ve küre yüzeyi aynı zamanda birerdönel yüzeydir.[12]

$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2} = 1$ denklem sonucu bir ise **reel elipsoid**

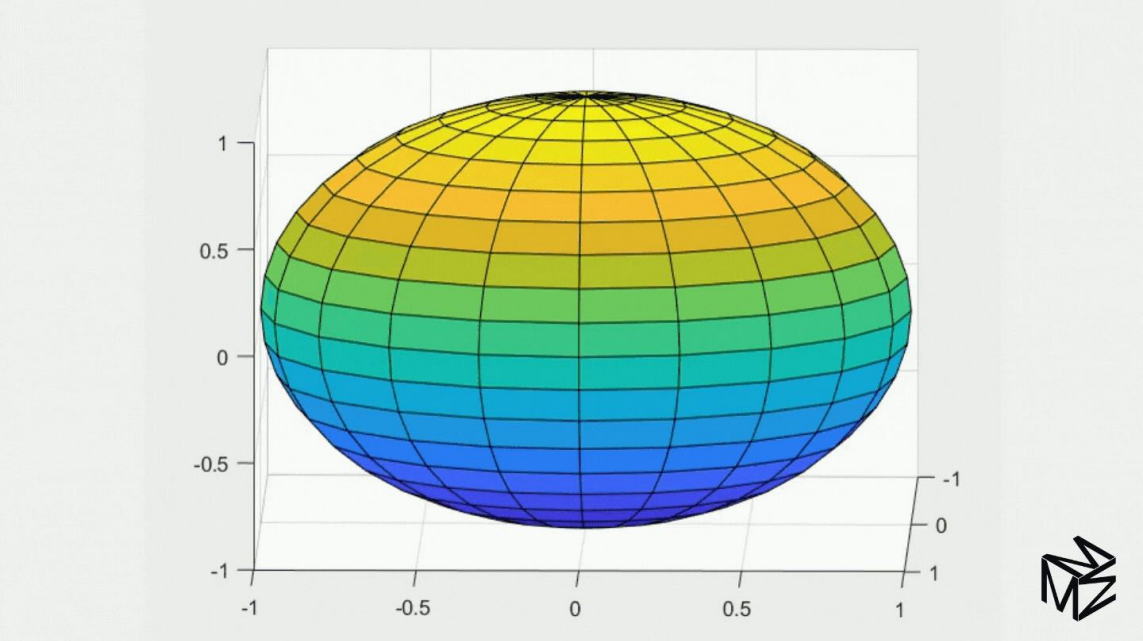
$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2} = -1$ denklem sonucu eksi bir ise **sanal elipsoid**

$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2} = 0$ denklem sonucu sıfır ise **nokta elipsoid** denir.



Dönel elipsoid Jeodezide kullanılan dönel elipsoid,bir elipsin (meridyen elipsinin) düşey Z ekseni (küçük yarı ekseni) etrafında 180° döndürülmesiyle elde edildiğinden X ve Y eksenleri boyunca yarı eksen uzunlukları birbirine (a = c) eşit olur. Elipsoid üç boyutlu geometrik kapalı şekillerden kürenin genel bir halidir. Diğer bir deyişle küre elipsoidin özel bir halidir$(a = b = c = R)$.

Dönel elipsoid jeodezik uygulamalarda çokca kullanıldığından dolayıdır ki dönel elipsoid kavramı yerine sadece elipsoid kelimesinin kullanıldığını görmekteyiz. [12]

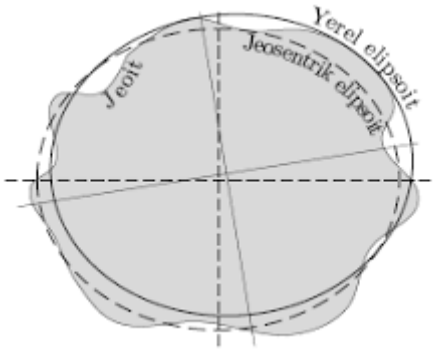


Dünya elipsoidinin boyutlarının belirlenmesi:

Dünya elipsoidinin büyüklüğü ve basıklığının belirleme yöntemleri iki ana grupta toplanabilir. Bunlar geometrik yöntem ve dinamik yöntemdir. Geometrik yöntemde yer üzerinde yapılan yay boyu ölçümlerinden ve astronomik gözlemlerden yararlanılır. Dinamik yöntem ise sarkaç veya gravimetri aletleri yardımıyla yapılan yer çekimi ölçülerine dayanır. Karalarda olduğu kadar denizlerde ve havada da gravite ölçüleri yapılabildiğinden dinamik yöntem daha pratik ve daha doğru sonuçlar verir. Ancak dinamik yöntemle yalnız elipsoidin basıklık değeri bulunabilir. [46]

Referans Elipsoidi:

Referans elipsoidi iki yaklaşım vardır. Bunlardan biri yeryuvarının tamamını temsil eden ortalama yer elipsoidi diğeri ise yerel elipsoiddir. Ortalama yer (jeosentrik) elipsoidinin merkezi dünyanın ağırlık merkezi ile çakışır. Ünlü jeodezi bilgini Helmert seçilecek böyle bir ortalama yer elipoidi ile jeoid arasındaki farkların ± 100m den fazla olamayacağını ileri sürmüştür.



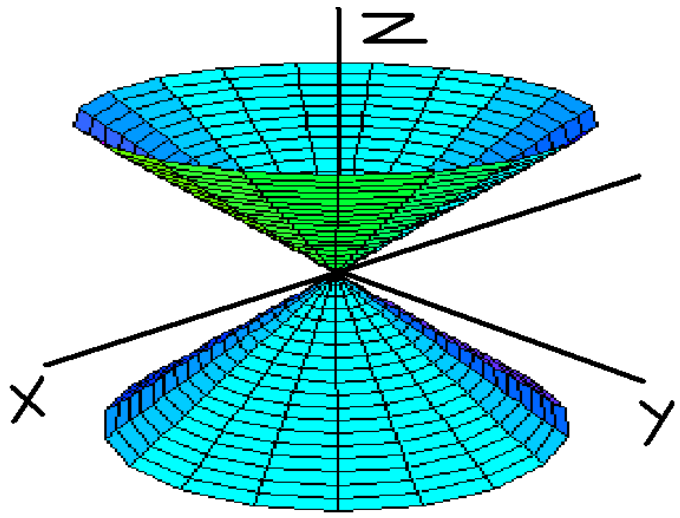
Günümüze kadar dünya için pek çok ortalama yer elipsoidi tanımlanmıştır. Teknolojideki ilerlemeye paralel olarak gelişen jeodezik ölçüm ve değerlendirme teknikleri sayesinde dünya için daha gerçekçi elipsoid parametreleri saptanmıştır. Tarihsel süreç içinde tanımlanan elipsoidlere örnek olarak: Delambre (1810), Airy (1830),Bessel Elipsoidi 1841, Hayford Elipsoidi 1910, Krassowsky Elipsoidi 1940, GRS80 Elipsoidi, WGS84 Elipsoidi sayılabilir. Bunlardan en çok kullanılan Hayford Elipsoidi (International Elipsoid) dir. Ülkemizde de temel jeodezik çalışmalarda bu elipsoid kullanılmıştır. Son yıllarda GPS uygulamalarının yaygınlaşması nedeniyle ülkemizde de WGS84 elipsoidi kullanılmaya başlamıştır. Aşağıda her iki elipsoide ait parametreler tabloda verilmiştir. [46]

1.3 TEK KANATLI HİPERBOLOİD

Hiperboloid:

Matematikte bir yüzey türüdür. İki konik yüzeyin kesilmesiyle oluşur ve çift taraflı olarak simetridir. Hiperboloid yüzeyi iki ayrı sonsuzlama çizgisinin çaprazına yerleştirilir ve bu çizgiler hiperboller olarak bilinir. Hiperboloid yüzeyi, birçok farklı şekilde ortaya çıkabilir ve birçok farklı uygulama ve disiplinde kullanılır. Örneğin, mühendislikte havaalanı kuleleri veya köprü kuleleri gibi yapılar için kullanılabilir. Ayrıca, ışıklandırmada veta sanat eserlerinde de kullanılabilir. Hiperboloid, matematiksel teoremlerin çözümünde de önemli bir rol oynar ve birçok farklı matematik disiplinde incelenir. [44]

$\frac{x^2}{a^2}+\frac{y^2}{b^2}-\frac{z^2}{c^2}=1$ bu eşitliği sağlayan **tek kanatlı hiperboloid** denir.



Bu eşitlikte

$x = 0$ *alınırsa oyz düzleminde* $\frac{y^2}{b^2} - \frac{z^2}{c^2} = 1$ *hiperbolü*

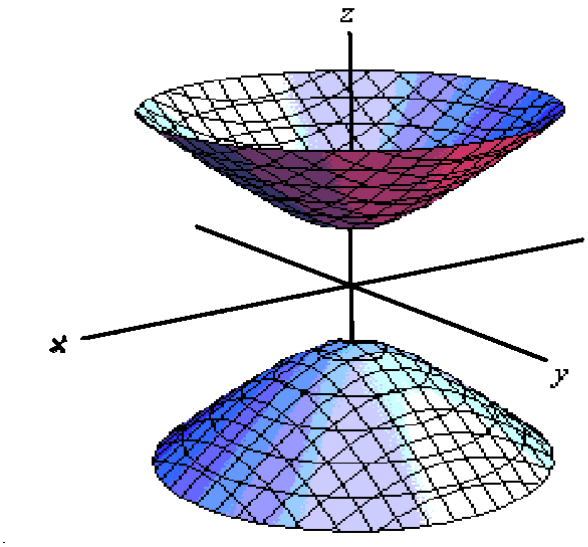
$y = 0$ *alınırsa oxz düzleminde* $\frac{x^2}{a^2} - \frac{z^2}{c^2} = 1$ *hiperbolü ve*

$z = 0$ *alınırsa oxy düzleminde* $\frac{x^2}{a^2} + \frac{y^2}{y^2} = 1$ *elipsi elde edilir.*

Burada a b'*yeeşit olmadığı için* ***eliptik***, $a = b$ *ise* ***dönel*** *yüzeydir.*

1.4 ÇİFT KANATLI HİPERBOLOİD

$\frac{z^2}{c^2} - \frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$ *bu eşitliği sağlayan denkleme* ***çift kanatlı hiperboloid*** *denir.*



Bu eşitlikte

$x = 0$ *alınırsa oyz düzleminde* $\frac{z^2}{c^2} - \frac{y^2}{b^2} = 1$ *hiperbolü ve*

$y = 0$ *alınırsa oxz düzleminde* $\frac{z^2}{c^2} - \frac{x^2}{a^2} = 1$ *hiperbolü elde edilir.*

z *sıfırdan farklı olmak zorundadır.* (*neden?*)

Burada b c den farklı yüzey için ***eliptik***, *b ve c eşit ise* ***dönel*** *yüzeydir.*[17]

1.5 ELİPTİK KONİ

Koni, matematikte bir düzlem içindeki dairenin her noktasını, düzlem dışındaki bir noktaya birleştiren doğru parçalarının meydana getirdiği geometrik şekil.

Dik üçgenin bir dik kenarı etrafında döndürülmesiyle elde edilen koniye, dik koni veya dönel koni denir. Koniler, tabanlarına göre; dairesel koni, eliptik koni gibi isimler alırlar. Dairesel bir dik koninin taban merkezini tepe noktasına birleştiren doğru parçasına, bu koninin ekseni veya yüksekliği denir.

Taban çevresinin herhangi bir noktasını tepeye birleştiren doğru parçasına koninin ana doğrusu veya *apotemi* adı verilir. Taban çevresinin her noktasını tepeye birleştiren doğru parçalarının meydana getirdiği yüzey, koninin yanal yüzeyi adını alır. Yanal yüzeyin alanı, taban çevresi ile apoteminin çarpımının yarısına eşittir. [28]

Taban yarıçapının uzunluğu r, apotemi uzunluğu a ise yanal yüzey alanı= π·r·a olur. Bir dairesel dik koninin hacmi de, taban alanı ile yüksekliğin çarpımının üçte biri alınarak elde edilir.

$v = \frac{1}{3}\pi r^2 h$ *bu formül ile bulunur. Bir dönel koninin düzlemlerle arakesitine,*

konikler (elips, parabol, hiperbol)adı verilir.

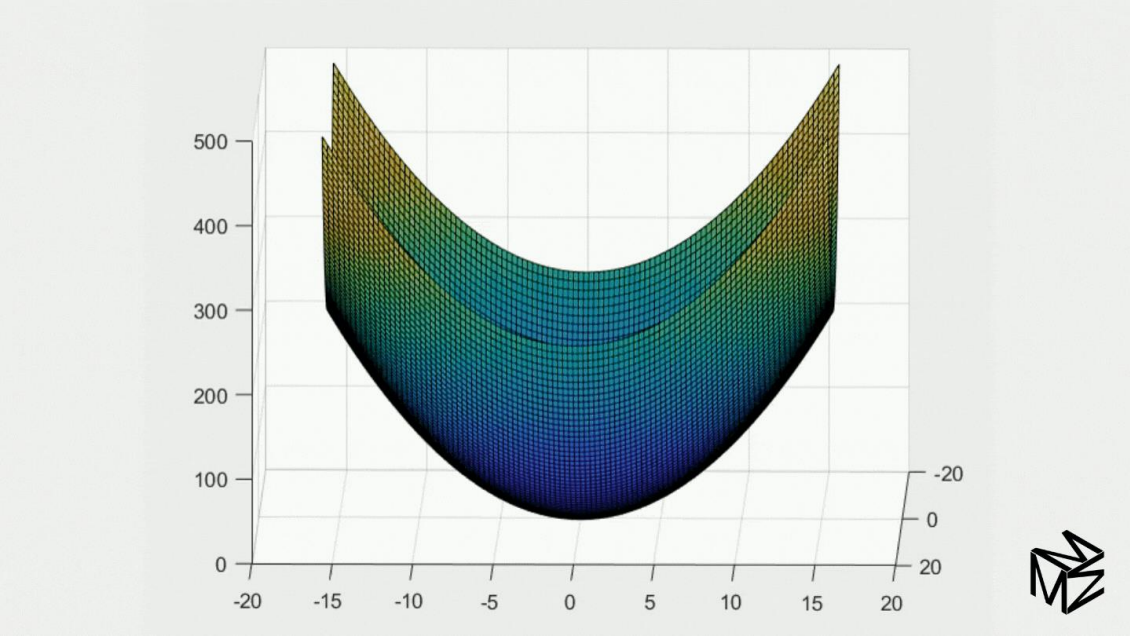
$\frac{x^2}{a^2} + \frac{y^2}{b^2} = \frac{z^2}{c^2}$ *bu formül eliptik koni formülüdür.*

Orijinden geçen bir doğruyu z ekseni etrafında döndürerek koni yüzeyi elde edebiliriz.

Orijin noktasına, koordinat eksenlerine ve koordinat düzlemlerine göre simetriktir.[28]

1.6 ELİPTİK PARABOLOİD

$\frac{x^2}{a^2} + \frac{y^2}{b^2} = \frac{z}{c}$ *yatay kesitler elips, düşey kesitler paraboldür.*



$\frac{x^2}{a^2} + \frac{y^2}{b^2} = cz$ *z ekseni boyunca uzanan z'nin her noktası için çizilen parabollerin birleşimi*

$\frac{x^2}{a^2} + \frac{z^2}{b^2} = cy$ *y ekseni boyunca uzanan y'nin her noktası için çizilen parabollerin birleşimi*

$\frac{y^2}{a^2}+\frac{z^2}{b^2}=cx$ *x ekseni boyunca uzanan x'inher noktası için çizilen parabollerin birleşimi* [12]

1.7 HİPERBOLİK PARABOLOİD

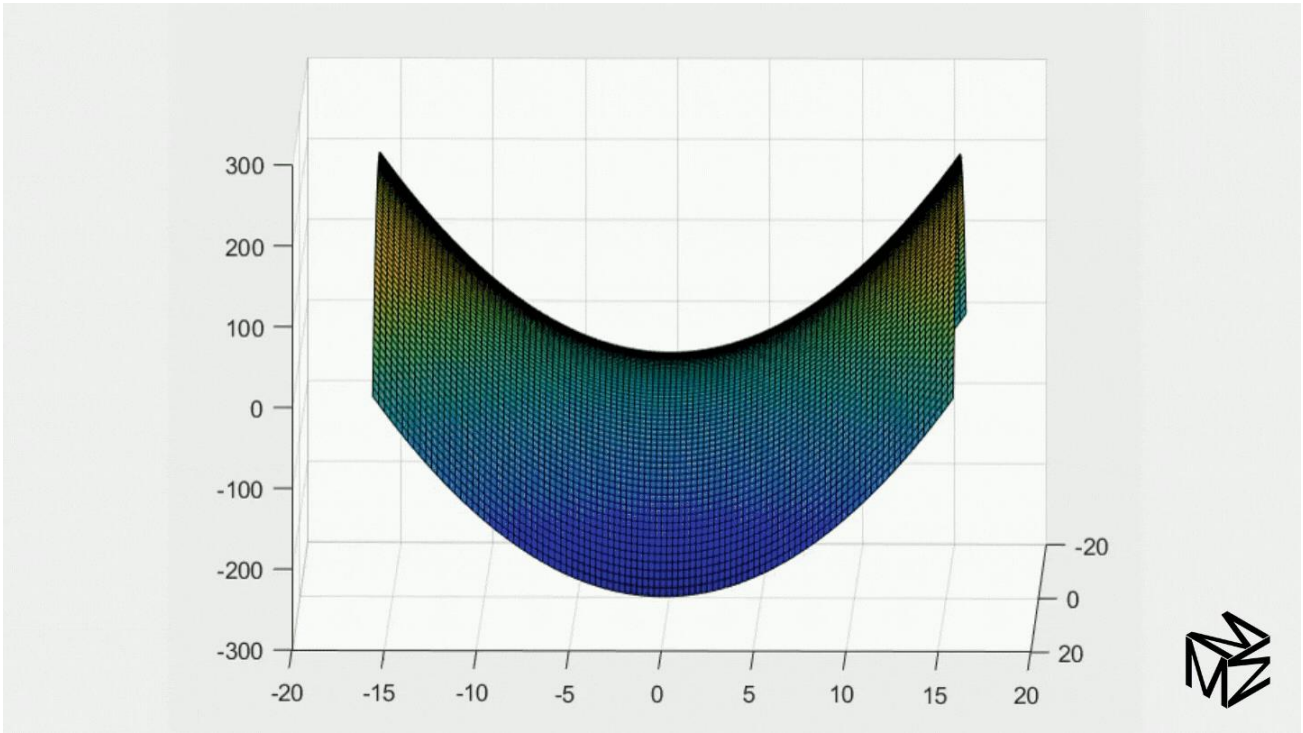
$z=\frac{x^2}{a^2}-\frac{y^2}{b^2}$ *eşitliği ile tanımlanan bir yüzeydir. Bu eşitlikte*

$x=0$ *alınırsa oyz düzleminde*

$z=-\frac{y^2}{b^2}$ *parabol*ü

$y=0$ *alınırsa oxz düzleminde*

$z=\frac{x^2}{a^2}$ *parabol*ü *elde edilir.*



$\frac{x^2}{a^2} - \frac{y^2}{b^2} = cz$ *z ekseni boyunca uzanan z'ninher noktası için çizilen hiperbol*ün *birleşimi*

$\frac{x^2}{a^2} - \frac{z^2}{b^2} = cy$ *y ekseni boyunca uzanan y'ninher noktası için çizilen hiperbollerin birleşimi*

$\frac{y^2}{z^2} - \frac{z^2}{b^2} = cx$ *x ekseni boyunca uzanan x'inher noktası için çizilen hiperbollerin birleşimidir.* [12]

· SİLİNDİRLER

Bir silindir bir doğruyu verilen sabit bir doğruya paralel tutarak, verilen düzlemsel bir eğri boyunca hareket ettirmekle üretilen bir yüzeydir. Eğri, silindirin üretici eğrisidir.

Silindirin dairesel silindir anlamına geldiği katı geometrisinde, üretici eğriler çemberlerdir, fakat artık her türlü üretici eğriye izin verilmektedir.

Bir silindir veya başka bir yüzeyi elle çizerken veya bilgisayarla üretilen bir tanesini incelerken, yüzeyin koordinat düzlemlerine paralel düzlemlerle kesişmesiyle oluşan eğrilere bakmak yardımcı olur. Bu eğrilere kesit veya iz denir.

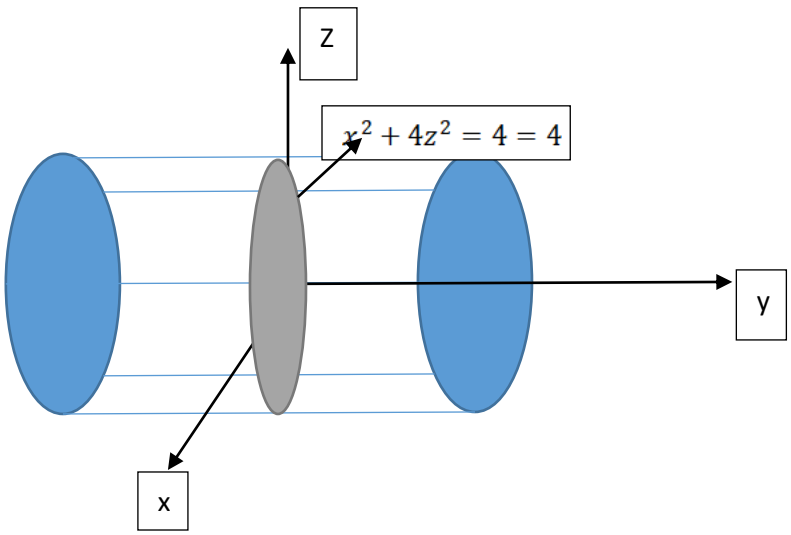
1.8 Eliptik Silindir

$x^2 + 4z^2 = 4$ eliptik silindiri y- eksenine paralel olan ve $x^2 + 4z^2 = 4$ elipsinden geçen doğrulardan yapılmıştır. Silindirin y-eksenine paralel düzlemlerdeki kesitleri veya "izleri" üretici elipse benzer elipslerdir. Silindir y-ekseni boyunca uzanır.

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

$$\frac{x^2}{a^2} + \frac{z^2}{c^2} = 1$$ Denklemleri Eliptik Silindir Belirtir.

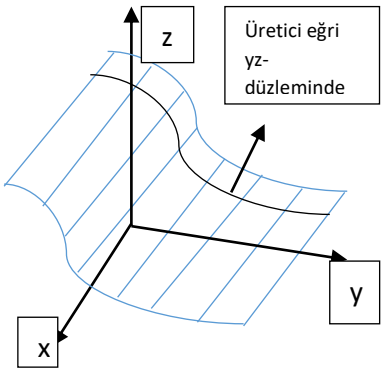
$$\frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$$

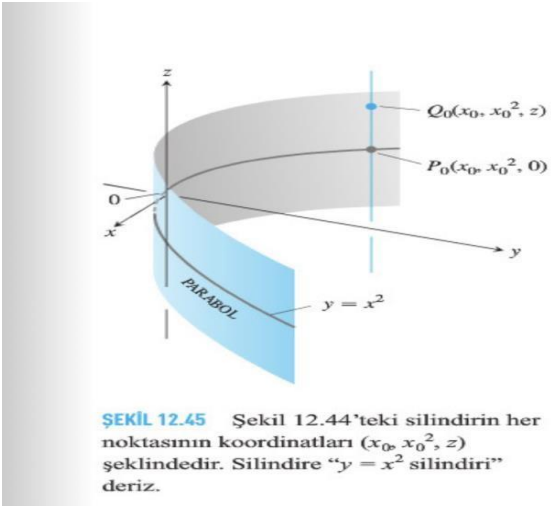


1.9. Parabolik Silindir

Z-eksenine paralel olan ve $y = x^2$, z = 0 parabolünden geçen doğruların ürettiği silindirin denklemi ;

$P_0(x_0, {x_0}^2, 0)$noktasının xy-düzlemindeki $y = x^2$ parabolünde bulunduğunu varsayalım. Bu durumda, z'nin herhangi bir değeri için, $Q(x_0, {x_0}^2, 0)$noktası silindir üzerinde bulunacaktır, çünkü P_0'dan geçen ve z-eksenine paralel olan $x = x_0, y = {x_0}^2$doğrusu üzerinde olacaktır . Tersine, y-koordinatı x-koordinatının karesine eşit olan her $Q(x_0, {x_0}^2, 0)$ noktası silindir üzerindedir. Çünkü P_0'dan geçen ve z- eksenine paralel olan $x = x_0, y = {x_0}^2$doğrusu üzerinde bulunmaktadır. Dolayısıyla, z'nin değeri ne olursa olsun, yüzey üzerindeki noktalar koordinatları $y = x^2$ denklemini sağlayan noktalardır. Bu $y = x^2$'yi silindirin denklemi yapar. Bu nedenle silindire "$y = x^2$ silindiri(parabolik silindir)" deriz.





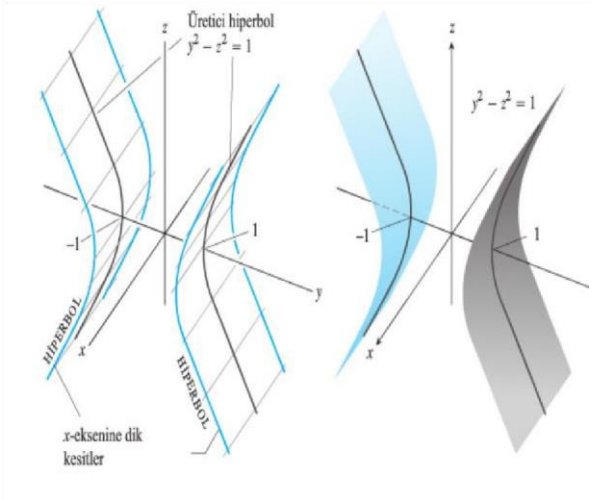
ŞEKİL 12.45 Şekil 12.44'teki silindirin her noktasının koordinatları (x_0, x_0^2, z) şeklindedir. Silindire "$y = x^2$ silindiri" deriz.

1.10. HİPERBOLİK SİLİNDİR

$y^2 - z^2 = 1$ Hiperbolik silindiri x-eksenine paralel ve yz-düzlemindeki $y^2 - z^2 = 1$ hiperbolünden geçen doğrularla yapılmıştır. Silindirin x-eksenine dik düzlemlerdeki kesitleri üretici hiperbole benzer hiperbollerdir.

$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$, $-\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$, $\frac{y^2}{b^2} - \frac{z^2}{c^2} = 1$ Denklemleri Hiperbolik

$\frac{x^2}{a^2} - \frac{z^2}{c^2} = 1$, $-\frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$ $-\frac{x^2}{a^2} + \frac{z^2}{c^2} = 1$ Silindir Belirtir.



1.11 KESİŞEN BİR ÇİFT DÜZLEM

$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 0$, $\frac{x^2}{a^2} - \frac{z^2}{c^2} = 0$, $\frac{y^2}{b^2} - \frac{z^2}{c^2} = 0$

denklemleri kesişen bir çift düzlemi belirtir.

1.12 ÇAKIŞIK BİR ÇİFT DÜZLEM

$x^2 = 0$, $y^2 = 0$, $z^2 = 0$

denklemleri çakışık bir çift düzlem belirtir.

1.13 GENEL KUADRİK DENKLEMİ

E^3, 3-boyutlu Öklid uzayında bir $\{O; x, y, z\}$ dik koordinat sistemi verilsin. x, y, z ye göre 2. dereceden bir $\varphi(x,y,z) = A_1x^2 + A_2y^2 + A_3z^2 + B_1xy + B_2xz + B_3yz + C_1x + C_2y + C_3z + D = 0$ denkleminin belirttiği yüzeye kuadrik denir ve $\exists A_i, B_i \neq 0,\ 1 \leq i \leq 3$ olmalıdır.

ÖZEL DURUMLAR

1) $A_1 = A_2 = A_3 = B_1 = B_2 = B_3 =$ **0** ise $C_1x + C_2y + C_3z + D = 0$ elde edilir. Bu bir düzlem denklemidir. O halde düzlem, 2. dereceden bir yüzeydir.

2) $A_i = A\ olsun.$

$Ax^2 + Ay^2 + Az^2 + B_1xy + B_2xz + B_3yz + C_1x + C_2y + C_3z + D = 0$ veya

$x^2 + y^2 + z^2 + B'_1xy + B'_2xz + B'_3yz + C'_1x + C'_2y + C'_3z + D = 0$ elde edilir.

Bu bir küre denklemi olup küre özel bir kuadriktir.

3) $C_i = 0$, $D = 0\ ise$

$A_1x^2 + A_2y^2 + A_3z^2 + B_1xy + B_2xz + B_3yz = 0$ bir koni yüzeyi belirtir.

1.14. KUADRİKLERİN MERKEZİL HALE DÖNÜŞTÜRÜLMESİ

Koniklere benzer olarak genel kuadratik denklemler ; C_1, C_2, C_3 lineer terimlerin

katsayılarını yok edecek şekilde öteleme, B_1, B_2, B_3 karışık terimlerin katsayılarını yok edecek şekilde bir dönme sayesinde merkezil hale (standart) getirilebilir.

1.14.1 Öteleme

E^3 Öklid uzayında $\{O; x, y, z\}$ ve $\{O'; x', y', z'\}$ dik koordinat sistemi verilsin. O' noktasının $\{O; x, y, z\}$ dik koordinat sistemindeki koordinatları O'= (p, q, r) olsun. O halde elde edilen

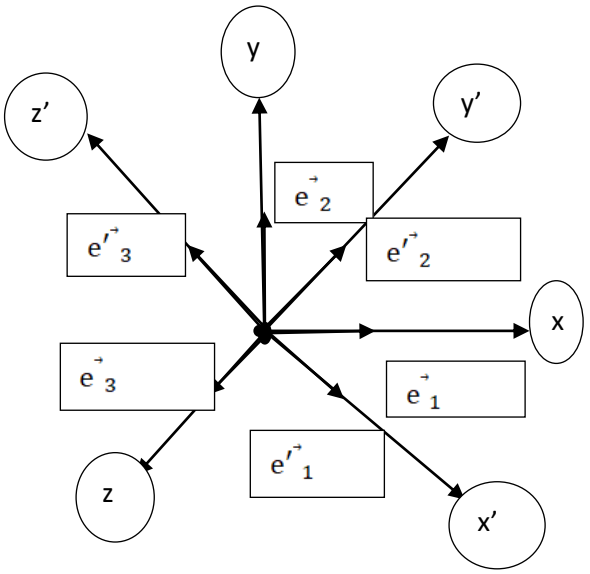
$$\left.\begin{aligned} x &= x' + p \\ y &= y' + q \\ z &= z' + r \end{aligned}\right\}$$ ötelemesi $\varphi(x, y, z) = 0$ denkleminde yerine yazılır ve x',y',z' lü terimlerin katsayıları sıfıra eşitlenirse p,q,r değerleri bulunur. Bu esnada $\varphi|_{O'} = D' = \varphi(p, q, r)$ olduğu görülür.

UYARI

$$\left.\begin{aligned} \varphi_x|O' &= 0 \\ \varphi_y|O' &= 0 \\ \varphi_z|O' &= 0 \end{aligned}\right\}$$ denklem sisteminin çözümü ile p,q,r katsayılarını ve

$D' = \varphi(p, q, r)$ katsayısı bulunur.

1.14.2 Dönme



$$\left.\begin{aligned} \measuredangle(e_1, e'_i) &= \alpha_i, \\ \measuredangle(e_2, e'_i) &= \beta_i, \\ \measuredangle(e_3, e'_i) &= \gamma_i, \end{aligned}\right\}$$ olmak üzere $\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} cos\alpha_1 & cos\alpha_2 & cos\alpha_3 \\ cos\beta_1 & cos\beta_2 & cos\beta_3 \\ cos\gamma_1 & cos\gamma_2 & cos\gamma_3 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$

yazılabilir. Bu dönme denklemi $\varphi(x,y,z)=0$ kuadriğine uygulanırsa karışık terimlerin katsayıları yok edilir. Böylece
$A_{,1}x'^2 + A'_2y'^2 + A'_3z'^2 + C'_1x' + C'_2y' + C'_3z' + D = 0$ elde edilir.

1.15. MATRİS FORMU İLE MERKEZİL HALE GETİRME

$\varphi(x,y,z)=$
$a_{11}x^2 + a_{22}y^2 + a_{33}z^2 + 2a_{12}xy + 2a_{13}xz + 2a_{23}yz + 2a_{14}x + 2a_{24}y + 2a_{34}z + a_{44} = 0$
kuadriği verilsin.
$a_{ij} = a_{ji}$ olmak üzere

C=$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$ matrisi simetriktir. O halde ;

A = $\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{12} & a_{22} & a_{23} \\ a_{13} & a_{23} & a_{33} \end{bmatrix}$, B = $\begin{bmatrix} 2a_{14} \\ 2a_{24} \\ 2a_{34} \end{bmatrix}$, Z = $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$ olmak üzere $\varphi(x,y,z)=0$ kuadriği
$Z^tAZ + B^tZ + a_{44} = 0$
Şeklinde matris formunda yazılabilir.

1.15.1. Sınıflandırma

1). $|C| \neq 0$, $|A| > 0$ ise kuadrik , elipsoid sınıfındadır.

2). $|C| \neq 0$, $|A| < 0$ ise kuadrik, hiperboloid sınıfındadır.

3). $|C| \neq 0$, $|A| = 0$ ise kuadrik, paraboloid sınıfındadır.

4). $|C| = 0$, $|A| = 0$ ise kuadrik, silindir sınıfındadır.

5). $|C| = 0$, $|A| \neq 0$ ise kuadrik, koni sınıfındadır.

1.15.2 Öteleme

$\varphi(x,y,z)=0$ kuadriğine x',y',z' terimlerini yok edecek şekilde

$\begin{cases} x = x' + x_0 \\ y = y' + y_0 \\ z = z' + z_0 \end{cases}$ ötelmesi uygulayalım:

$$a_{11}x'^2 + a_{22}y'^2 + a_{33}z'^2 + 2a_{12}x'y' + 2a_{13}x'z' + 2a_{23}y'z' + 2x'(a_{11}x_0 + a_{12}y_0 + a_{13}z_0 + a_{14}) + 2y'(a_{22}y_0 + a_{12}x_0 + a_{23}z_0 + a_{24}) + 2z'(a_{33}z_0 + a_{13}x_0 + a_{23}y_0 + a_{34}) + \begin{pmatrix} a_{11}x_0^2 + a_{22}y_0^2 + a_{33}z_0^2 + 2a_{12}x_0y_0 + 2a_{13}x_0y_0 + 2a_{23}x_0y_0 + 2a_{14}x_0 + 2a_{24}y_0 \\ +2a_{34}z_0 + a_{44} \end{pmatrix} = 0$$

ifadesinin lineer terimlerinin katsayıları sıfıra eşitlenirse,

$$a_{11}x_0 + a_{12}y_0 + a_{13}z_0 + a_{14} = \frac{1\partial\varphi}{2\partial x} = 0\ ,$$

$$a_{22}y_0 + a_{12}x_0 + a_{23}z_0 + a_{24} = \frac{1\partial\varphi}{2\partial y} = 0\ ,$$

$$a_{33}z_0 + a_{13}x_0 + a_{23}y_0 + a_{34} = \frac{1\partial\varphi}{2\partial z} = 0$$

elde edilir.Ayrıca sabit kayşayı için

$$F' = x_0[a_{11}x_0 + a_{12}y_0 + a_{13}z_0 + a_{14}] + y_0[a_{22}y_0 + a_{12}x_0 + a_{23}z_0 + a_{24}] + [z_0a_{33}z_0 + a_{13}x_0 + a_{23}y_0 + a_{34}] + a_{14}x_0 + a_{24}y_0 + a_{34}z_0 + a_{44} = 0$$

yazılabilir. O halde

$F' = \varphi(x_0, y_0, z_0)$ veya $F' = a_{14}x_0 + a_{24}y_0 + a_{34}z_0 + a_{44}$ elde edilir. Böylece

$\varphi(x, y, z) = 0$ kuadriğine

$$\begin{cases} x = x' + x_0 \\ y = y' + y_0 \\ z = z' + z_0 \end{cases}$$ ötelmesi uygulanırsa

$$a_{11}x'^2 + a_{22}y'^2 + a_{33}z'^2 + 2a_{12}x'y' + 2a_{13}x'z' + 2a_{23}y'z' + F' = 0$$ veya

$Z' = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$ için $Z'^tAZ' + F' = 0$ elde edilir.

1.15.3. Dönme

$Z^tAZ + B^tZ + a_{44} = 0$ matris formunda kuadrik verilsin. A matrisinin özdeğerleri λ_1, λ_2,λ_3 ve bu özderlere karşılık gelen birim özvektörler Λ_1 , Λ_2, Λ_3 olmak üzere

$P = [\Lambda_1, \Lambda_2, \Lambda_3]$ matrisi elde edilir. Şimdi $Z = PZ'$ dönüşümünü $\varphi(x, y, z,) = 0$ kuadriğe uygulayalım: $Z'^tAZ + B^tZ + a_{44} = 0$ ve $Z = PZ'$ $için$ $(PZ')^tA(PZ') + B^t(PZ') + a_{44} = 0$ veya $Z'^t(P^tAP)Z' + (B^tP)Z' + a_{44} = 0$ bulunur.

$P^tAP = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$ olmak üzere $\lambda_1 x'^2 + \lambda_2 y'^2 + \lambda_3 z'^2 + (ax' + by' + cz')a_{44} = 0$ veya

$\frac{(u_1 x' + v_1)^2}{m_1{}^2} \pm \frac{(u_2 y' + v_2)^2}{m_2{}^2} + \frac{(u_3 z' + v_3)^2}{m_3{}^2} = 1$ elde edilir.

UYARI !!

Eğer önce öteleme uygulanırsa kuadriğin son halinin $Z^tAZ + F = 0$ olduğunu biliyoruz. Şimdi bu son eşitliğe dönme uygulansın. P matrisi A matrisinin ortonormal öz vektörlerinden elde edilen bir matris ise

$P^tAP = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$ yazılabilir. $Z'AZ + F = 0$ kuadriği ve Z = PZ' dönüşümü için

$(PZ')^tA(PZ') + F = 0$ veya $Z'^t(P^tAP)Z' + F = 0$ yazılabilir. Buradan

$[x' \quad y' \quad z'] \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} + F = 0$ veya $\lambda_1 x'^2 + \lambda_2 y'^2 + \lambda_3 z'^2 + F = 0$ elde edilir.

$\lambda_1 x'^2 + \lambda_2 y'^2 + \lambda_3 z'^2 = -F$ eşitliği için (F>0)

1.Eşitliğin sol tarafındaki terimler pozitif ise kuadrik elipsoiddir.
2.Eşitliğin sol tarafındaki terimlerden bir tanesi negatif ise kuadrik tek kanatlı hiperboloiddir.
3.Eşitliğin sol tarafındaki terimlerinden iki tanesi negatif ise kuadrik çift kanatlı hiperboloiddir.
4.Eşitliğin sağ tarafındaki terimler sıfır ise kuadrik konidir.

ÖRNEK

$\varphi(x, y, z) = x^2 + y^2 + z^2 + 4xy + 2y + 4z + 1 = 0$ kuadriğinin cinsini belirleyiniz ve merkezil hale getiriniz.

ÇÖZÜM

$C = \begin{bmatrix} 1 & 2 & 0 & 0 \\ 2 & 1 & 0 & 1 \\ 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 1 \end{bmatrix}$ ve $A = \begin{bmatrix} 1 & 2 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ olmak üzer $|C| \neq 0 \; ve \; |A| = -3$ olduğundan kuadrik hiperboloid sınıfındadır.

Öteleme

$$\begin{cases} \frac{\partial \varphi}{\partial x} = 2x + 4y \\ \frac{\partial \varphi}{\partial y} = 2y + 4x + 2 \\ \frac{\partial \varphi}{\partial z} = 2z + 4 \end{cases}$$

ve O'=(p.q.r) olmak üzere

2p+4q=0
2q+4p+2=0
2r+4=0

Elde edilir. Buradan $p = -\frac{2}{3}, q = \frac{1}{3}, r = -2$ bulunur. Ayrıca

$F' = 0\left(-\frac{2}{3}\right) + 1\left(\frac{1}{3}\right) + 2(-2) + 1 = -\frac{8}{3}$ elde edilir. O halde

$$\begin{cases} x = x' - \frac{2}{3} \\ y = y' + \frac{1}{3} \\ z = z' - 2 \end{cases}$$ ötelemesi sonucunda

$(x')^2 + (y')^2 + (z')^2 + 4x'y' - \frac{8}{3} = 0$ elde edilir.

A matrisinin özdeğerleri

$|A - \lambda I| = \begin{bmatrix} 1-\lambda & 2 & 0 \\ 2 & 1-\lambda & 0 \\ 0 & 0 & 1-\lambda \end{bmatrix} = 0$ veya $(1-\lambda)[(1-\lambda)^2 - 4] = 0$ için

$\lambda_1 = 1, \lambda_2 = -1, \lambda_3 = 3$ elde edilir.Buradan

$\lambda_1 = 1$ için

$AX = \lambda X \ \ veya \begin{bmatrix} 1 & 2 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$ olmak üzere

$$\begin{cases} x_1 + 2x_2 = x_1 \\ 2x_1 + x_2 = x_2 \\ x_3 = x_3 \end{cases}$$ elde edilir. Buradan

$\Lambda_1 = \begin{bmatrix}0\\0\\1\end{bmatrix}$ bulunur.

$\lambda_2 = -1$ için

$AX = \lambda X \ \ veya \begin{bmatrix}1 & 2 & 0\\2 & 1 & 0\\0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix} = (-1)\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}$ olmak üzere

$\begin{cases} x_1 + 2x_2 = -x_1 \\ 2x_1 + x_2 = -x_2 \\ x_3 = -x_3 \end{cases}$ elde edilir. Buradan

$\Lambda_2 = \begin{bmatrix}\frac{1}{\sqrt{2}}\\\frac{-1}{\sqrt{2}}\\0\end{bmatrix}$ bulunur.

$\lambda_3 = 3$ için

$AX = \lambda X \ \ veya \begin{bmatrix}1 & 2 & 0\\2 & 1 & 0\\0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix} = 3\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}$ olmak üzere

$\begin{cases} x_1 + 2x_2 = 3x_1 \\ 2x_1 + x_2 = 3x_2 \\ x_3 = 3x_3 \end{cases}$ elde edilir. Buradan

$\Lambda_3 = \begin{bmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\\0\end{bmatrix}$ bulunur.

Böylece

$P = \begin{bmatrix}0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\\0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\\1 & 0 & 0\end{bmatrix}$

$P^t A = \begin{bmatrix}0 & 0 & 1\\\frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0\\\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0\end{bmatrix}\begin{bmatrix}1 & 2 & 0\\2 & 1 & 0\\0 & 0 & 1\end{bmatrix} = \begin{bmatrix}0 & 0 & 1\\\frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0\\\frac{3}{\sqrt{2}} & \frac{3}{\sqrt{2}} & 0\end{bmatrix} ve$

$$P^tAP = D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

elde edilir. O halde $\varphi(x', y', z') = 0$ kuadriğine

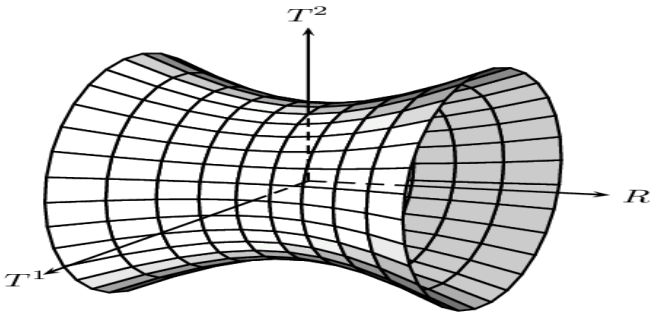
$Z' = PZ'', \left(Z' = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}, Z'' = \begin{bmatrix} x'' \\ y'' \\ z'' \end{bmatrix} \right)$ dönüşümü uygulanırsa $(Z')^tAZ' - \frac{8}{3} = 0$ için

$(PZ'')^tAZ'' - \frac{8}{3} = 0$ veya $(Z'')^tDZ' - \frac{8}{3} = 0$ elde edilir. Buradan

$$[x'' \quad y'' \quad z''] \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} x'' \\ y'' \\ z'' \end{bmatrix} - \frac{8}{3} = 0$$ veya $(x'')^2 - (y'')^2 + 3(z'')^2 = \frac{8}{3}$ veya

$$\frac{(x'')^2}{\frac{8}{3}} - \frac{(y'')^2}{\frac{8}{3}} + \frac{(z'')^2}{\frac{8}{9}} = 1$$ elde edilir.

$|C| \neq 0$ ve $|A| - 3 < 0$ olduğundan kuadarik hiperboloid sınıfındandır.Bu son denklemden ise kuadiriğin tek kanatlı hiperboloid olduğu görülür.



2. PYTHON

2.1 PYTHON'UN KISA TARİHÇESİ

Python Hakkında Genel Bilgi

Python, Hollandalı bir programcı olan Guido Van Rossum tarafından 90'lı yıllarda Amsterdam'da geliştirilmeye başlanmış bir bilgisayar dilidir. Nesneye yönelik kodlama ve açık kaynaklı dil olma özelliğine sahip olan Python, programcılar tarafından öğrenim ve kullanım kolaylığı sağlamasının yanı sıra, açık kaynak kütüphanelerle birçok projenin kolaylıkla tamamlanmasını sağladığı için son zamanlarda çok sık tercih edilen üst düzey bir programlama dilidir. [23]

Python günümüzde Dünya genelinde yazılım geliştiricilerin oluşturduğu büyük bir kitle tarafından mobil uygulama üretmek, web site oluşturmak, finans, oyun, bilim ve eğitim gibi çeşitli birçok alanda kullanılmaktadır.

Python'ı (C ve C++ vs.) diğer programlama dillerinden ayıran en önemli özelliği derlenme ihtiyacı olmamasıdır. Bu özelliği öğrenim kolaylığı ve hızla geliştirilmiş uygulamaları beraberinde getirmiştir. Kod parçacıklarının daha az satır kullanılarak yazılmasını sağlar. Anlaşılması kolay bir söz dizimi bulunan Python yeni bir program yazmayı da, yazılan bir program üstünde düzenleme yapmayı ve geliştirmeyi de kolaylaştırır. Bu da diğer programlama dillerine göre Python'ı ön plana çıkarmaktadır.

Python dili, Guido Van Rossum tarafından başta hobi bir proje olarak geliştirilmeye başlanmıştır. Yıllar içinde açık kaynak olmasının verdiği özelliklerle gelişimi ve kullanımı büyük kitlelere ulaşmış ve programlama dili hızlı geliştirmelerle Python Versiyonları olarak hayatımıza girmiştir. [4]

Python Versiyonları

1991 yılında Guido Van Rossum kaynak temel kodlarını 0.9.0 versiyonu ile dünya ile paylaşmıştır. 1994 yılında 0.9.0 versiyonu geliştirilerek lambda, map, filter ve reduce gibi fonksiyonları da bünyesine katarak 1.0 ve 1.6 sürümleri olarak paylaşılmıştır.

Piyasada bilinen en güncel iki farklı Python serisi vardır. Bunlar Python 2.7.15 ve Python 3.7.0'dır. Python sürümü 2 rakamıyla başlıyorsa o sürüm Python2, 3 rakamıyla başlıyorsa o sürüm Python3 serisine aittir.

2000 yılında Python 2.0 sürümü geliştirilmiştir. Her geçen gün gelişime devam eden Python programlama dili 2020 yılında 2.0 sürümünün de gelişimini tamamlayarak, 2.7 sürümünü çıkarmıştır. Bu sürümden sonra 2.0 gelişimi de son bulmuştur. Günümüzde bile birçok Python uygulamaları Python2 sürümleri ile çalışmaya devam etmektedir. 2008 yılında Python2 hatalarından arındırılmış daha güçlü bir sürüm olan Python3 geliştirilmeye başlandı. 3.8.3 sürümü ile Python3 serisi 3.0 olarak piyasaya giriş yapmıştır.

3 rakamıyla başlıyorsa, o sürüm Python3, 2 rakamıyla başlıyorsa o sürüm Python2 sürümlerine aittir. Python2 ile yazılan bir program aralarındaki büyük farklılıklar sebebiyle Python3 üzerinde çalışmaz. Aynı şekilde Python3 ile yazılan bir program da Python2 üzerinde

çalışmayacaktır. Birbirinin devamı diller olsalar da bulundurdukları modüllerin farklılıkları buna izin vermeyecektir. [27]

2.2 PYTHON'UN TEMEL KODLARI

Python da ilerleyebilmek için değişken kavramının ne olduğunu nasıl kullanıldığını anlamak gerekir. Yazılan tüm kod satırları değişken tanımlama ve bu değişkenleri fonksiyonlarla çalıştırma üzerinedir. Temel mantığı bu olsa da değişkenlerin ne olduğunu neye değişken dendiğini kavrayamamak ileride daha komplike projeler yapıldığında zorluk çıkarabilir.

Değişken herhangi bir bilgi içeren ifadedir. Diğer programlama dillerinde değişken önden tanımlanırken Python da değişken yazıldığı ilk değeri alır. Python da değişken türünün ne olduğunu -bir sonraki bölümde detaylı olarak görülecek- tam sayı, ondalık sayı, metin vs. belirtmeye gerek yoktur. Python da metin değişkenini diğer programlama dillerine göre çift tırnak ("),üst ayraç (') içine yazabilir iki yazım biçimini derleyicide kabul eder. [6,34]

```
[1]: x=7

[2]: y='beykent'

[3]: print(x)
     7

[5]: print(y)
     beykent
```

Burada bahsettiğimiz Python dilinin anlaşılır söz dizimi durumunu en temel de gözlemi görülür. Bu gibi esnek, kolay ve hızlı kod yazma özellikleri ile Python öğrenmesi kolay bir üst düzey programlama dili kabul edilir. [42]

Değişken atama yaparken bazı hususlara dikkat etmek gerekir. Bunlar;

- ✓ X ve y gibi kısa değerler de verilebilir, isim soy isim gibi daha uzun değerler de verilebilir.
- ✓ Değişkenimiz sayı ile başlayamaz.
- ✓ Harf veya alt çizgi ile başlayabilir. Büyük-küçük harf farklılıkları içerir 'PYT' ile '$pyth$' değişkenlerini farklı okur.

Aşağıdaki kod dizilimlerinde geçerli olan değişken adlarının kullanımlarının yazımı

```
[1]: #Geçerli değişken adları
[2]: degisken="beykent"
[3]: de_gisken="beykent"
[4]: _de_gis_ken="beykent"
[5]: deGisken="beykent"
[6]: DEGİSKEN="beykent"
[ ]: #Kuraldışı değişken adları
[ ]: 2degisken="beykent"
[ ]: de-gis-ken="beykent"
[ ]: degis-ken="beykent"
```

görülmektedir.

2.3 PYTHON VERİ YAPILARI

Veri Tipi: Programlama dillerinde her bilgisayar diline özel, derleyici denilen, kodların yazılmasını ve test edilmesini sağlayan yardımcı araçlar vardır. Proje bu derleyiciler aracılığıyla bilgisayara komutlar vererek oluşturulur. Derleyiciye yazılacak etkenlerin türünü, değerini, nasıl kullanılıp saklanacağını, nasıl işleneceğini tanımlayan kavrama veri tipi denir. [2,3]

Python için en temel 8 adet veri tipleri şunlardır;

1. **Tam Sayı (İnteger) Veri Tipi**
2. **Ondalık Sayı (Float) Veri Tipi**
3. **Metin (String) Veri Tipi**
4. **Mantıksal (Boolean) Veri Tipi**
5. **Liste (List) Veri Tipi**
6. **Demet (Tuple) Veri Tipi**
7. **Sözlük (Dictionary) Veri Tipi**
8. **Küme (Set) Veri Tipi**

Bu başlıklar dışında birçok veri tipi vardır. En temel veri tipleri olarak bu sekiz başlık kabul edilir. [15]

1.Tam Sayı (İnteger) Veri Tipi

Python, tam sayı olarak bilinen negatif-pozitif sayıları ve sıfır değerlerini alır. Bu değerleri Tam Sayı Veri Tipi kullanarak derleyiciye yazdırılır.

Derleyicide bir komut yazdırmak için yazıcı anlamına gelen ‘print()’ kavramı kullanılır belirtilen parantez içine yazarak istenilen sayı girişini yapılır.

ÖRNEK: Ekranda 2023 sayısının çıktısı alınan kod. Öncelikle bir sayi=2023 değeri atanır ve bu sayı değeri print() fonksiyonu ile ekrana yazdırılır.

```
[1]: sayi=2023

[2]: print(sayi)

     2023
```

ÖRNEK : İki farklı sayı değeri atanır, bu değerler toplam, fark ve çarpım olmak üzere üç farklı yazılı denkleme dönüştürülür. Daha sonra print() fonksiyonu kullanılarak işlemlerin sonuçları ekrana yazdırılır.

```
[1]: sayi1=10

[2]: sayi2=5

[3]: toplam=sayi1+sayi2

[5]: print("Toplam:", toplam)

     Toplam: 15

[6]: fark=sayi1-sayi2

[7]: print("Fark:",fark)

     Fark: 5

[8]: carpim=sayi1*sayi2

[9]: print("Çarpım:",carpim)

     Çarpım: 50
```

2.Ondalık Sayı (Float) Veri Tipi

Python da ondalık (kesirli, virgüllü) sayıları belirtmek ve yazdırmak için farklı bir veri tipi olan float adı verilen ondalık sayı veri tipini kullanmak gerekir. Aşağıdaki kod ekranından örneğini inceleyiniz.

```
[1]: float=3.14

[2]: ondalik_sayi=3.14

[3]: print(float)
     3.14

[4]: print(ondalik_sayi)
     3.14
```

Örnekte de görüldüğü gibi veri tipini İngilizce veya Türkçe yazmak sonucu değiştirmeyecektir. Derleyici iki dili de anlar. Tek dikkat edilmesi gereken konu Türkçe özel karakterler kullanmamaktır. Misal örnekte 'ondalik_sayi' yazmak yerine 'ondalık_sayı' yazılsaydı derleyici i harfini kabul etmeyecek ve hata verecekti. Bu özel karakterlere bilinen programlama dillerinde karşılık genel olarak alınmaz.

```
[1]: ondalık_sayı

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[1], line 1
----> 1 ondalık_sayı

NameError: name 'ondalık_sayı' is not defined
```

Programlama dillerinin hepsinde o dile özel yorum satırı dediğimiz sadece kodun incelenmesi sonucunda programcının görebileceği, içerisine yazdığım herhangi bir şeyi çıktı olarak almayan bir kavram vardır. Python için bu yorum satırı '#' işaretidir. Bu işaret devamında yazılan herhangi bir içerik çıktı olarak alınmaz. Derleyici de daha silinik tonda veya farklı renkte gösterilir. [5]

‘#’ özelliği sizden sonra kodu inceleyecek olan çalışma yapacak olan programcıya notlar bırakmak için veya bir kod dizilimini görünmez kılmak için kullanılır. Aşağıdaki da ki örnekte bu bahsedilen durumu inceleyiniz.

```
[1]: sayi1=13

[2]: sayi2=6

[3]: bolme= sayi1/sayi2

[4]: #Görülsüğü üzere çıktımız ondalıklı olarak yazılır.
     #Peki ya tam bölünen bir sayıyı float veri tipi ile yazarsak ne olur?

[5]: sayi3=12

[6]: bolme=sayi3/sayi2

[7]: print(bolme)
     2.0
```

Yukarıda ekran çıktısında görülüyor ki yorum satırı olarak yazılan parçalar derleyici için görünmez olur. Bu örnekte bir ondalıklı sonuç ve bir de tam bölünebilen bir sonuç elde edilir. Görülüyor ki float kullanıldığı için sayı tam bölünse de çıktısı ‘2.0’ olarak ondalıklı ifade edilir. Tam sayı ve ondalık sayı veri tipi farkını burada net olarak gözükmektedir.

3. Metin (String) Veri Tipi

Python da yazı verilerini belirtmek için metin(string) veri tipi kullanılır. Tek tırnak veya çift tırnak içine yazılabilir. İki yazımda doğrudur, derleyici ikisinin de çıktısını alabilir.

Aşağıdaki kod dizilimlerinde tek tırnak ve çift tırnak kıllanarak ekrana metin yazdırma işleminin sonuçlarının farklı olmayacağı iki çeşit kullanımın da doğru olduğu görülmektedir. Önce yazı1 ve yazi2 değerleri atanmış daha sonrasında print() fonksiyonu ile atanan yazı değerlerinin ekran çıktısı alınır.

```
[1]: #Tek tırnak kullanılarak metin tanımlama

[2]: yazi1="Merhaba,Beykent!"

[3]: #Çift tırnak kullanarak bir metin tanımlama

[4]: yazi2="Matematik Bölümü"

[5]: #Ekrana yazdırma

[6]: print(yazi1)

     Merhaba,Beykent!

[7]: print(yazi2)

     Matematik Bölümü
```

Aşağıdaki örnekte ise metin birleştirmeyi ve metin uzunluğunu bulmayı sağlayan kod dizilimi görülmektedir. Daha önce atanan metin1 ve metin2 değerleri '$birleşik\_metin$' ifadesi ile $metin1 + metin2$ olarak print() yazdırma fonksiyonu kullanılarak ekran çıktısı alınır. Devamında metnin uzunluk sayı değerinin bulunmasını sağlayan '$uzunluk = len(metin1)$' ifadesi yine print() fonksiyonu kullanılarak metnin uzunluk değerinin sayısal değeri çıktısı alınır.

```
#Tek tırnak kullanarak mmetin tanımlama

metin1='Merhaba, Beykent!'

#Çift tırnak kullanarak metin tanımlama

metin2="Matematik Bölümü"

#Metini ekrana yazdırma

print(metin1)
```

```
Merhaba, Beykent!
```

```
print(metin2)
```

```
Matematik Bölümü
```

```
#Metin birleştirme

birlesik_metin= metin1 +' '+ metin2

print(birlesik_metin)
```

```
Merhaba, Beykent! Matematik Bölümü
```

```
#Metin uzunluğunu bulma

uzunluk= len(metin1)

print('Metin uzunluğu:', uzunluk)
```

```
Metin uzunluğu: 17
```

İndex ve find metin içinde belli bir bölümü, kelimeyi, metin parçasının bulunmasına sebep olur. İndex eğer belirtilen metin yoksa ekranda hata verir. Find ise -1 döner ve kodun anlaşılırlığını kolaylaştırır.

```
#Metin içindeki belirli bir karakterin indeksini bulma

index= metin2.find('Matematik Bölümü') #İndex() yerine find() kullanmak daha güvenlidir

print('indeks:',index)
```

```
indeks: 0
```

```
#Metin küçük harfe çevirme

kucuk_harfler=metin1.lower()

print('Küçük harf hali:', kucuk_harfler)
```

```
Küçük harf hali: merhaba, beykent!
```

İndeks 0 çıkmış aranılan Matematik metninin metnin içinde bulunduğu anlamına gelmektedir.

4. Mantıksal (Boolean) Veri Tipi

Mantıksal (Boolean) olarak bilinen veri tipi ‘doğru’ ve ‘yanlış olmak üzere ’yalnızca iki cevaba karşılık gelir. İki tane girilen veriyi değerlendirerek sonucun doğru veya yanlış olduğunun belirtilmesini sağlar. Temel kullanımını inceleyelim. [19]

```
[1]: print(10>9)
```

```
True
```

```
[2]: print(10==9)
```

```
False
```

```
[3]: print(10<9)
```

```
False
```

Ekran çıktısı belirtildiği gibi verinin durumuna göre doğru ve yanlış olmak üzere iki çeşit sonuca bağlanır. Bu veri tipini kullanmak için ‘if’ ve ‘else’ yapılarının bilinmesi gerekmektedir. İf bir koşulun sağlanması sonucunda kullanılırken, else yapısı koşul

```
a = 500

b = 70

if a>b: print("a, b'den büyüktür")
else: print("a, b'den küçüktür")
```

```
a, b'den büyüktür
```

sağlanmadığında kullanılır.

Ekran çıktısı $b > a$ olmadığı için ekran çıktısı $'b, a'dan\ büyük\ değildir'$ olacaktır.

5. Liste (List) Veri Tipi

Liste veri tipi farklı veri tiplerine ait bilgileri bulundurabilir, sıralı ve değişebilir hale getirilmesini sağlar. Listeler [ ] arasına yazılanlarla tanımlanabilir. [16]

Metinsel(string) değerler tırnak içine yazılarak kullanılırken numerik değerler(int,float) normal yazılır. Liste içi değerler de virgül yardımıyla ayrılır. Aşağıdaki örnekte temel bir liste görebilirsiniz.

```
[1]: #Bir liste oluşturma
     liste= [1,2,3,4,5]

[2]: #Liste elemanlarına erişim
     print ("Listenin ilk elemanı:" , liste[0])

     Listenin ilk elemanı: 1

[3]: print("Listenin ikinci elemanı:", liste[1])

     Listenin ikinci elemanı: 2
```

6. Demet (Tuple) Veri Tipi

Demet veri tipi liste gibi sıralı değerlerden oluşur ama liste değiştirebilir veriler içerirken, demet(tuple) veri tipi içerisine yazılan değer değiştirilemez, güncellenemez, sabit kalır.

Aşağıdaki kod dizilimlerinde sırasıyla demet oluşturma, demet elemanlarına erişim ve demet uzunluğunun kullanımı görülmektedir. [22]

```
[1]: #Demet oluşturma

[2]: demet = (1, 2, 3, "Dört", 5.0)

[3]: #Demet elemanlarına erişim

[4]: print("Demetin ilk elemanı:", demet[0])

     Demetin ilk elemanı: 1

[5]: print("Demetin ikinci elemanı"), demet[1]

     Demetin ikinci elemanı
[5]: (None, 2)

[6]: print("Demetin son elemanı"), demet[-1]

     Demetin son elemanı
[6]: (None, 5.0)

[7]: #Demetin uzunluğu

[9]: print("Demetin uzunluğu:", len(demet))

     Demetin uzunluğu: 5

[10]: #Demetin elemanlarını sıralama(hata verir,çünkü demet değiştirilemez)
      #sirali_demet = sorted(demet)
      #print("Sıralı demet:" , sirali_demet)
```

7.Sözlük (Dictionary) Veri Tipi

Sözlük veri yapısı key ve value şeklinde verilerin saklanılacağı bir veri yapısıdır. Sözlük, demetler ve listeler gibi sıralı veri türlerinin işlemlerini gerçekleştiremez. Sözlükler gömülü eşleme türüne aittir.

ÖRNEK: Aşağıdaki örnekte bir sözlük kümesi oluşturulur. Daha sonrasında bu küme üzerinden print() fonksiyonu yardımıyla sözlük elemanlarına erişilir. Sözlük kümesine yeni bir eleman eklemek için değer atanır.

```
#Sözlük oluşturma

sozluk = {"anahtar1": "değer1", "anahtar2": "deger2", "anahtar3": "değer3"}

#Sözlük elemanlarına erişim

print("Anahtar1'in degeri:", sozluk["anahtar1"])
print("Anahtar2'nin degeri:", sozluk["anahtar2"])
```

```
Anahtar1'in degeri: değer1
Anahtar2'nin degeri: deger2
```

```
#Sözlüğe yeni bir eleman ekleme

sozluk["anahtar4"] = "deger4"

#Sözlükteki tüm anahtarlar

anahtarlar = list(sozluk.keys()) print("Sözlüğün anahtarları:", anahtarlar)
```

```
Sözlüğün anahtarları: ['anahtar1', 'anahtar2', 'anahtar3', 'anahtar4']
```

```
#Sözlükteki tüm değerler

degerler = list(sozluk.values()) print("Sözlüğün değerleri", degerler)
```

```
Sözlüğün değerleri ['değer1', 'deger2', 'değer3', 'deger4']
```

```
#Sözlükteki tüm çiftler

ciftler = list(sozluk.items()) print("Sözlük ciftleri", ciftler)
```

```
Sözlük ciftleri [('anahtar1', 'değer1'), ('anahtar2', 'deger2'), ('anahtar3', 'değer3'), ('anahtar4', 'deger4')
```

8.Küme (Set) Veri Tipi

Tıpkı listeler, demetler, karakter dizileri, sayılar gibi kümeler de Python'daki veri tiplerinden biridir. Matematikten bilinen "küme" kavramıyla bağlantılıdır tıpkı matematikteki gibi her elemanın yalnızca bir kez bulunabileceği bir veridir. Bu veri tipi, matematikteki kümelerin

sahip olduğu bütün özellikleri taşır. Yani matematikteki kümelerden bilinen kesişim, birleşim ve fark gibi özellikler Python'daki kümeler için de geçerlidir. Aşağıdaki kod diziliminde bu durumu inceleyiniz. [21]

```
[1]: kume = {1,2,3,4,5}

[2]: print(kume)

     {1, 2, 3, 4, 5}

[3]: kume.add(6) #Kümeye eleman ekleme
     kume.remove(3) #Kümeden eleman çıkarma
     print(kume)

     {1, 2, 4, 5, 6}

[4]: kume1= {1,2,3,4,5}
     kume2= {4,5,6,7,8}

[5]: birlesim = kume1| kume2 #Birleşim(Union)
     kesisim = kume1 & kume2 #Kesişim (Intersection)
     fark = kume1 - kume2 #Fark

[6]: print("Birleşim:", birlesim)
     print("Kesişim:", kesisim)
     print("Fark:", fark)

     Birleşim: {1, 2, 3, 4, 5, 6, 7, 8}
     Kesişim: {4, 5}
     Fark: {1, 2, 3}
```

Bu bölüme kadar verilen başlıklardaki 8 veri tipine dair bir temel oluşturulmuştur. En temel örneklerle veri tiplerinin çalışma prensibini kavramaya çalışılmıştır. Python da kodlama yaparken, başlıkları anlatırken uygulanan örnekler kadar basit kod dizilimleri ile uğraşılmaz. Daha komplike ve birden fazla veri tipinin kullanımı ile çeşitli kod bütünleri oluşturulur. Yukarıda veri tiplerini anlatırken verilen kullanım örnekleri tamamen veri tipinin mantığının anlaşılmasına yönelik temel uygulamalardır. Tüm veri tipleri çeşitli hikayelerle çeşitli kod satırlarına dönüşür. Aşağıda verilen örneklerde bahsedilen komplike kod dizilimlerinin birkaç örneğini inceleyiniz.

Python Veri Tipleri Örnek Kod Dizilimleri

ÖRNEK: Bir öğrencinin genel okul bilgilerinin öğrenilmesini sağlayan kod dizilimi aşağıdaki şekildedir. Öncelikle öğrenci bilgilerini içeren komutlar kümesi oluşturulmuş daha

sonrasında print() kullanılarak as, soyad, notlar, devamsızlık, mezuniyet durumu yazdırılır. İf else koşulu kullanılarak öğrenci bilgilerine göre mezunsa ekrana “Mezun” veya “Mezun Değil” yazdıran kod kümesi oluşturulur.

Kod diziliminin sonunda öğrenci notlarının 70 veya 70’den büyük olma durumunda, devamsızlık sayısı da 5’e eşit veya 5’den az ise öğrenci mezuniyet durumunun doğru veya yanlış olduğunu gösteren if koşulları koda dökülür.

```
def ogrenci_bilgileri_goster(ogrenci):
    print(f"{ogrenci['ad']} {ogrenci['soyad']}'ın Bilgileri:")
    print(f"Notlar:{ogrenci['notlar']}")
    print(f"Devamsızlık:{ogrenci['devamsizlik']} gün")
if ogrenci["mezuniyet_durumu"]:
    print("Mezun Durumu: Mezun")
else:
    print("Mezun Durumu: Mezun Değil")

#Öğrenci sınıfından bir öğrenci oluştur
ogrenci_bilgileri_goster(ogrenci)
#Mezuniyet durumunu kontrol et
if sum(ogrenci["notlar"].values()) / len(ogrenci["notlar"]) >= 70 and ogrenci["devamsizlik"] <=5:
    ogrenci["mezuniyet durumu"] = True
```

```
Mezun Durumu: Mezun Değil
BUSE AL'ın Bilgileri:
Notlar:{'matematik:90,fizik': 85, 'kimya': 78}
Devamsızlık:5 gün
```

ÖRNEK: Bir alışveriş sepeti uygulamasını, sepet durumunun kontrol edildiği kod dizilimi aşağıdaki gibidir. İlk olarak ürün adedi, fiyatı, ismi, stokta olup olmama durumu, sepetteki ürünlerin listesi, fatura bilgileri, ürün bilgileri, indirimlerin kümesi sırası ile integer(), float(), string(), boolean(), list(), tuple(), dictionary(), set() kavramları kullanılarak koda çevrilir.

[1]:
```
#Hikaye bir alışveriş sepeti uygulaması
#Ahmet, çevrimiçi alışveriş yapan bir kullanıcıdır.
#Sitemizdeki alışveriş sepetini kullanarak farklı veri tiplerini gösteren bir uygulama geliştirilsin.
```

[2]:
```
# 1.TAM SAYI(INTEGER) VERİ TİPİ : Ürün Adeti
urun_adedi = 3
```

[3]:
```
# 2. ONDALIK SAYI (FLOAT) VERİ TİPİ : Ürün Fiyatı
urun_fiyatı = 49.99
```

[4]:
```
# 3.METİN (STRİNG) VERİ TİPİ : Ürün İsmi
urun_ismi = "Akıllı Telefon"
```

[5]:
```
# 4.MANTIKSAL (BOOLEAN) VERİ TİPİ : Ürün Stok Durumu
urun_stokta_mi = True
```

[6]:
```
# 5.LİSTE(List) VERİ TİPİ : Sepet İçeriği
sepet = ["Laptop", "Kulaklık", "Mobil Şarj", "Kamera"]
```

[7]:
```
# 6.DEMET (TUPLE) VERİ TİPİ : Fatura Bilgileri
fatura_bilgileri = ("Ahmet", "Yılmaz", "123456789", "ahmet@gmail.com")
```

[8]:
```
# 7. SÖZLÜK (DİCTİONARY) VERİ TİPİ : Ürün Bilgileri
urun_bilgileri = {
    "urun adi": 199.99,
    "urun_stokta_mi": True
}
```

[20]:
```
# 8.KÜME (SET) VERİ TİPİ : İndirimler
indirimler = {"Black Friday", "Yılbaşı Kampanyası", "Öğrenci İndirimi"}
```

[10]:
```
#Sepet Toplamını Hesaplyan Fonksiyon
def sepet_toplamini_hesapla(urun_adedi,urun_fiyati):
    toplam = urun_adedi * urun_fiyatı
    return toplam
```

[12]:
```
#Alışveriş Sepeti İçeriğini Gösteren Fonksiyon
def sepet_icerigini_goster(sepet):
    print ("Alışveriş Sepeti İçeriği")
    for urun in sepet :
        print("-" + urun)
```

[14]:
```
#Fatura Bilgilerini Gösteren Fonksiyon
def fatura_bilgilerini_goster(fatura_bilgileri):
    ad, soyad, telefon, email = fatura_bilgileri
    print(f"Fatura Bilgileri:\nAd: {ad}\nSoyad: {soyad}\nTelefon: {telefon}\nEmail: {email}")
```

Kodlamanın devamında her bir başlık için girilen bilgiler def kod kümeleri oluşturularak sıralanır. Oluşturulan kod kümeleri derleyicide çalıştırılınca sepetteki ürünlerin durumu hakkında bilgilerin akışının sağlandığı çıktılar alınır.

[15]:
```python
#Ürün Bilgilerini Gösteren Fonksiyon
def urun_bilgilerini_goster(urrun_bilgileri):
    print("\nÜrün Bilgileri:")
    for key, value in urun_bilgileri.items():
        print (f"{key}: {value}")
```

[16]:
```python
#İndirimleri Gösteren Fonksiyon
def indirimleri_goster(indirimler):
    print("\nindirimler:")
    for indirim in indirimler:
        print("-"+ indirim)
```

[21]:
```python
#Kullanıcıya Sepet İçeriğini ve Fatura Bilgilerini Göater
print("Hoşgeldiniz, Ahmet!")
sepet_icerigini_goster(sepet)
fatura_bilgilerini_goster(fatura_bilgileri)
urun_bilgilerini_goster(urun_bilgileri)
indirimleri_goster(indirimler)
```

```
Hoşgeldiniz, Ahmet!
Alışveriş Sepeti İçeriği
-Laptop
-Kulaklık
-Mobil Şarj
-Kamera
Fatura Bilgileri:
Ad: Ahmet
Soyad: Yılmaz
Telefon: 123456789
Email: ahmet@gmail.com

Ürün Bilgileri:
urun adi: 199.99
urun_stokta_mi: True

indirimler:
-Yılbaşı Kampanyası
-Öğrenci İndirimi
-Black Friday
```

[19]:
```python
#Toplam Tutarı Hesapla ve Göster
toplam_tutar = sepet_toplamini_hesapla(urun_adedi,urun_fiyatı)
print("\nToplam Tutarı:",toplam_tutar, "TL")
```

```
Toplam Tutarı: 149.97 TL
```

ÖRNEK: Bir kütüphanede, adı 'Python Programlama' olan bir kitap hakkında bilgi edinilmesini sağlayan kod örneği aşağıdaki şekildedir. Öncelikle temel veri tipi olarak bildiğimiz integer(), float(), string(), boolean(), list(), tuple(), dictionary(), set() kullanılarak kütüphane kitap sayısii kitap fiyatı, kitap adı, kitap indirimleri, yazarları, yayımlanma yılı, bilgilerini veren kod kümeleri oluşturulur. Print() kullanılarak bu değerler çıktı olarak alınır.

[1]:
```
#1. TAM SAYI (INTEGER) VERİ TİPİ
kutuphane_kitap_sayisi = 100
```

[2]:
```
#2. ONDALIK SAYI (FLOAT) VERİ TİPİ
kitap_fiyati = 29.99
```

[3]:
```
#3. METİN (STRİNG) VERİ TİPİ
kitap_ad1 = "Pythın Programlama"
```

[4]:
```
#4. MANTIKSAL (BOOLEAN) VERİ TİPİ
kitap_indirimde_mi = True
```

[6]:
```
#5.LİSTE (LİST) VERİ TİPİ
yazarlar = ["Guido van Rossum", "Mark Lutz"]
```

[7]:
```
#6. DEMET (TUPLE) VERİ TİPİ
yayim_yili = (2010,)
```

[8]:
```
#7. SÖZLÜK (DİCTİONARY) VERİ TİPİ
kitap_bilgisi = {
    "ad": "Python Programlama",
    "yazarlar": ["Guido van Rossum", "Mark Lutz"],
    "yayim_yili":2010,
    "fiyat": 29.99 }
```

[9]:
```
#8. KÜME (SET) VERİ TİPİ
populer_kitaplar = {"Python Programlama", "Veri Bilimi", "Yapay Zeka"}
```

Kodlamanın devamında indirim durumunun kontrol edilmesini sağlayan temel bir if else koşul yapısı kullanılır. Kitap indirimde ise “Bu kitap indirimde!” indirimde değilse “Bu kitap maalesef indirimde değildir.”, ekran çıktısı alınacak kod dizilimi yazılır.

[10]:
```
#HİKAYE
print(f"{kitap_ad1} adlı kitap, {yazarlar[0]} ve {yazarlar[1]} tarafından yazılmıştır.")
print(f"Kitap {yayim_yili[0]} yılında yayımlanmıştır ve şu anda {2023 - yayim_yili[0]} yıl geçmiştir.")
```

```
Pythın Programlama adlı kitap, Guido van Rossum ve Mark Lutz tarafından yazılmıştır.
Kitap 2010 yılında yayımlanmıştır ve şu anda 13 yıl geçmiştir.
```

[11]:
```
#İNDİRİM DURUMUNU KONTROL ETMEDEN DİREKT OLARAK DURUMU YAZDIRMA
print("Bu kitap indirimde!" if kitap_indirimde_mi else "Bu kitap maalesef indirimde değildir.")

print(f"Kütüphanemizde toplam{kutuphane_kitap_sayisi} kitap bulunmaktadır.")
print(f"Kitap fiyatı:{kitap_fiyati} TL")
print(f"Popüler kitaplar arasında {populer_kitaplar} bulunmaktadır.")
```

```
Bu kitap indirimde!
Kütüphanemizde toplam100 kitap bulunmaktadır.
Kitap fiyatı:29.99 TL
Popüler kitaplar arasında {'Yapay Zeka', 'Veri Bilimi', 'Python Programlama'} bulunmaktadır.
```

2.4 PYTHON FONKSİYONLAR

Fonksiyonların temel görevi, anlaşılması zor işlemleri bir bütün haline getirerek, bu işlemleri tek bir işlemle yapmamızı sağlamaktır. Fonksiyonlar çoğunlukla, yapmak istediğimiz işlemler için bir yol haritası gibidir. Fonksiyonları kullanarak, bir ya da birden fazla adımdan var olan işlemleri tek bir başlık altında görüntüleyebiliriz. Python'da 'fonksiyon' kavramı diğer yazılım dillerinde 'rutin' ya da 'prosedür' olarak tanımlanır. Bu tanıma bakınca görüyoruz ki fonksiyonlar kavramı rutin olarak tekrar edilen işlemleri tek bir başlık altında toplayan yardımcılardır.

Bloklar süslü parantez olarak bildiğimiz '{ }' işareti ile belirtilen, bu işaretler içinde yazılan kodlardır. Bu kodlara kod bloğu ve blok denmektedir. Python'da fonksiyon, verilen görevi yapan bloktur. Fonksiyonlar kod bloklarının daha kolay ve kullanışlı hale getirmek için kullanılan kavramlardır. "def" kelimesi, Python programlama dilinde bir fonksiyon tanımlamak için kullanılan bir anahtar kelimedir. "def" kelimesi, "define" (tanımla) kelimesinin kısaltmasıdır ve bir fonksiyonun başlangıcını gösterir.

Python'da fonksiyon tanımı şu şekildedir:

Her fonksiyonun bir adı bulunur ve fonksiyonlar sahip oldukları bu adlarla anılır. (print fonksiyonu, open fonksiyonu, vb.) Şekil olarak, her fonksiyonun isminin yanında birer parantez işareti bulunur. (open(), print() vb.) Bu parantez işaretlerinin içine, fonksiyonlara işlevsellik kazandıran bazı parametreler yazılır. (open(dosya_adı), print("Merhaba Dünya!"), len("İstanbul") vb.)

Fonksiyonlar farklı sayıda parametre alabilir. Örneğin print() fonksiyonu toplam 256 adet parametre alabilirken, input() fonksiyonu yalnızca tek bir parametre alır.

Fonksiyonların isimli ve isimsiz parametreleri vardır. print() fonksiyonundaki sep, end ve file parametreleri isimli parametrelere örnekken, mesela print("Merhaba Dünya!") kodunda Merhaba Dünya! parametresi isimsiz bir parametredir. Aynı şekilde input("Adınız: ") gibi bir kodda Adınız: parametresi isimsiz bir parametredir.

Eğer bir parametrenin varsayılan bir değeri varsa, o parametreye herhangi bir değer vermeden de fonksiyonu kullanabiliriz. Python bu parametrelere, belirli değerleri ön tanımlı olarak kendisi atayacaktır. Varsayılan değerli parametrelere başka birtakım değerler de verilebilir. [43,31]

Aşağıdaki örnekte def (tanımlamak) kelimesi kullanılarak bir fonksiyona başlanır. Daha sonra fonksiyon tanımlanır ve print() yardımı ile fonksiyon çağırılır. Örneğin devamında aynı yol haritası izlenerek parametreli bir fonksiyon çağrılır. Son olarak tanımlanan değerler ve fonksiyonlar birleştirilerek $a + b$ işleminin toplamı elde edilir.

```
def merhaba_de():
    print("Merhaba!")
    #Fonksiyonu çağırmak
    merhaba_de()
```

```
#Parametreli fonksiyon çağırma
def toplama(a,b):
    sonuc = a+b
    return sonuc
```

```
#Fonksiyonu çağırmak ve dönen değeri kullanmak
toplam = toplama(3,4)
print(toplam)
```

```
7
```

Temel Fonksiyon Tipleri

A. İçerik Üreten Fonksiyon Tipleri

- **print()** fonksiyonu ekrana çıktı almamızı sağlayan bir fonksiyondur. Tek tırnak (‘), çift tırnak (“”), üç tırnak (“”” “”””) şeklinde gösterimi mevcuttur.

ÖRNEK: Print() fonksiyonu içerisine ekran çıktısı alınmak istenen metin çift tırnak içine yazılarak ‘Merhaba, Beykent’ ekran çıktısı alınır.

```
print("Merhaba, Beykent")
```

```
Merhaba, Beykent
```

- **len()** fonksiyonu, bir nesnenin (örneğin, bir dize, liste, tuple vb.) uzunluğunu ölçmek için kullanılır. Uzunluk, nesnedeki elemanların veya karakterlerin sayısını temsil eder**.**

ÖRNEK: Beş elemanlı bir liste oluşturulmuş ve len() yardımı ile bu listenin uzunluğu bulunur.

```
liste = [1,2,3,4,5]
uzunluk = len(liste)
print("Liste uzunlığu:", uzunluk)
```

```
Liste uzunlığu: 5
```

- **type()** fonksiyonu içine yazılan değişkenin ya da değerin tipini verir. Kullanımı type(kelime) veya type("Merhaba") şeklindedir.

ÖRNEK: Aşağıdaki kod diziliminde sırasıyla sayi,ondalik_sayi, metin, liste, sözlük değer atamaları yapılır. Daha sonra print() yardımıyla bu atamaların type olarak öğrendiğimiz tiplerinin ne olduğunu gösteren çıktılar alınır.

```
sayi = 42
ondalik_sayi = 3.14
metin = "Merhaba,Beykent!"
liste = [1,2,3]
sozluk = {"anahtar": "değer"}

print("sayi değişkeninin türü", type(sayi))
print("ondalik_sayi değişkeninin türü:", type(ondalik_sayi))
print("metin değişkeninin türü:", type(metin))
print("liste değişkeninin türü:", type(liste))
print("sozluk değişkeninin tüeü:", type(sozluk))
```

```
sayi değişkeninin türü <class 'int'>
ondalik_sayi değişkeninin türü: <class 'float'>
metin değişkeninin türü: <class 'str'>
liste değişkeninin türü: <class 'list'>
sozluk değişkeninin tüeü: <class 'dict'>
```

- **range()** fonksiyonu belirli aralıkta bulunan sayıları göstermek için kullanılır. Genelde for döngüsü ile kullanılır.

ÖRNEK: Aşağıdaki örnekte for döngüsü kullanılarak 5 sayısına kadar olan sayıların sıralanışının gösteriminin çıktısı alınır.

```
for i in range(5):
    print(i)
```

```
0
1
2
3
4
```

- **sum()** bir dizi içerisindeki değerlerin toplamını alır.

ÖRNEK: Aşağıdaki örnekte beş elemanlı numbers kümesi oluşturulmuş ve daha sonra bu küme elemanlarının toplamını bulmamızı sağşayan sum() fonksiyon tipi kullanılarak bu sayıların toplamının çıktısı alınır.

```
numbers = [1,2,3,4,5]
toplam = sum(numbers)
print("Listedeki sayıların toplamı:", toplam)
```

```
Listedeki sayıların toplamı: 15
```

- **max()** bir grup içindeki ondalıklı (veya tam sayı) sayıların en büyüğünün çıktısını alan bir fonksiyondur.

ÖRNEK: Aşağıdaki kod örneğinde karışık terimli altı elemanlı bir numbers kümesi oluşturulur. Daha sonra bu küme içindeki en büyük değerli terimin çıktısı alan max() fonksiyonu kullanılarak en büyük değere ulaşılır.

```
numbers = [3,1,4,5,9,2]
en_buyuk = max(numbers)
print("Listedeki En Büyük Sayı:", en_buyuk)
```

```
Listedeki En Büyük Sayı: 9
```

- **min()** bir grup ifadedeki toplanmış verilerin en düşük sayısal değerinin çıktısını alan fonksiyondur.

ÖRNEK: Aşağıdaki kod örneğinde karışık terimli altı elemanlı bir numbers kümesi oluşturulur. Daha sonra bu küme içindeki en küçük değerli terimin çıktısını alan min() fonksiyonu kullanılarak kümedeki en küçük değere ulaşılır.

```
numbers = [3,1,4,5,9,2]
en_kucuk = min(numbers)
print("Listedeki En Küçük Sayı:", en_kucuk)
```

```
Listedeki En Küçük Sayı: 1
```

B. Matematiksel Fonksiyonlar

- **abs()** fonksiyonu, bir sayının mutlak değerini döndürmek için kullanılır. Döndürülecek değer integer, float ve karmaşık sayı olabilir.

ÖRNEK: Aşağıdaki kod örneğinde öncelikle negatif_sayi değeri atanmış daha sonra bu sayının mutlak değerini gösteren abs() fonksiyonu kullanılarak , print() yardımıyla ekrana tanımlanan negatif sayının mutlak değeri yazdırılır.

```
negatif_sayi = -10
mutlak_deger = abs(negatif_sayi)
print(f"{negatif_sayi} sayısının mutlak değeri: {mutlak_deger}")
```

```
-10 sayısının mutlak değeri: 10
```

- **round()** fonksiyonu bir sayıyı belli ölçütlere göre yukarı veya aşağı doğru yuvarlamamızı sağlar.

ÖRNEK: Aşağıdaki örnekte negatif_sayi değeri atanmış ve round() fonksiyonu kullanılarak bu atanan değerin en yakın değere yuvarlanması sağlanır.

```
ondalik_sayi = 3.14159
yuvarlanmis_sayi = round(ondalik_sayi,2)
print(f"{ondalik_sayi} sayısı {yuvarlanmis_sayi}'ye yuvarlandı.")
```

```
3.14159 sayısı 3.14'ye yuvarlandı.
```

- **pow() (veya ** operatörü)** Bu fonksiyon herhangi bir sayının üssünü almak için kullanılır.

ÖRNEK: Aşağıdaki kod örneğinde taban ve üs değerleri atanarak bu değerlerin pow() fonksiyonu ile üslü ifade edilmesi sağlanır.

```
taban = 2
us = 3
sonuc = pow(taban, us)
print(f"{taban} üzeri {us} = {sonuc}")
```

```
2 üzeri 3 = 8
```

C. Kullanıcı Giriş Fonksiyonu

- **input()** fonksiyonu kullanıcıdan gelen girişi yazdırır. Çıktı alınan kullanıcının girdiği parametredir.

ÖRNEK: Aşağıdaki örnekte input fonksiyonu kullanılarak kullanıcıdan giriş alınması gereken bir kutu oluşturulmuştur. Daha sonrasında bu kutuya girilecek olan isim ekrana çıktı olarak alınacaktır.

```
isim = input("Lütfen adınızı girin")
print("Merhaba," "+ isim +" "!")
```

```
 Merhaba,+ isim +!
Lütfen adınızı girin
```

Belirtilen fonksiyon tiplerinden daha farklı ve çok çeşitli fonksiyonlarda vardır. Burada en temel fonksiyonlar ele alınmıştır.

2.5 PYTHON KOŞULLAR VE DÖNGÜLER

Çalışma prensibini anlamak için çeşitli matematiksel koşul ifadelerini tanımamız gerekiyor. Bunlar aşağıda belirtildiği gibidir.

$x == y$ *Eğer x ve y birbirine eşitse doğrudur, değilse yanlıştır*
$x < y$ *Eğer x, y'den küçükse doğrudur, değilse yanlıştır*
$x <= y$ *Eğer x, y'den küçük ya da eşitse doğrudur, değilse yanlıştır*

$x > y$ Eğer x, y'den büyükse doğrudur, değilse yanlıştır
$x >= y$ Eğer x, y'den büyük ya da eşitse doğrudur, değilse yanlıştır.
$x != y$ Eğer x, y'den farklı ise doğrudur, değilse yanlıştır.

Koşul

İf-Else Yapısı: Belirli bir koşulun doğruluğunu kontrol eder ve koşul sağlanıyorsa belirli bir bloğu çalıştırır. Eğer koşul sağlanmıyorsa, başka bir koşulu kontrol edebilir. **elif (else if) blokları**, birden fazla koşulu kontrol etmek için kullanılır. İlk başta bir koşul belirlenir; koşul sağlanırsa belirlenen işlem yapılır. Bu koşul sağlanmazsa başka bir işlem devreye girerek o bölüm çalışmaya başlayacaktır.

ÖRNEK: Aşağıdaki kod örneğinde kullanıcıdan input() fonksiyonu kullanılarak bir integer sayı birimi yaş girişi yapılması istenir. Daha sonrasında eğer bu sayı girişi integer değilse ekrana alınması gereken çıktılar print() yardımıyla koda aktarılır. Kod diziliminin devamında if koşulu kullanılarak bir sınır belirlenir. $Yas < 16$ olan bu sınır eğer 16 altında bir integer değer girilirse ekrana 'Henüz reşit değilsiniz.' Yazısını yazdıracaktır. Sonrasında kullanılan else fonksiyonu ise bu bahsedilen if koşulunun karşılığının olması durumda ekrana 'Reşitsiniz.' yazısını yazdıracaktır.

```
#Kullanıcıdan yaş girişi almak
try:
    yas= int(input("Yaşınızı girin:"))
except ValueError:
    print("Geçerli bir yaş girişi yapmadınız.")
#Yaşa göre mesaj vermek
    if yas<16:
        print("Henüz reşit değilsiniz.")
    else:
        print("Reşitsiniz.")
```

ÖRNEK: Aşağıdaki kod örneğinde atutar değeri input() yardımıyla değer girişi alınarak belirlenir. Daha sonrasında if kavramı ile $atutar < 50$ olması gerektiği belirtilir. Bu koşulun sağlanmaması durumda ekrana 'Kargo ücreti ile toplam tutar:' print() yardımı ile $atutar + 7$ değerinin müşterinin ödemesi gereken toplam tutarı verdiği belirtilir. Else kavramı kullanılarak da bu koşul sağlanmazsa ekrana 'Kargo bedava. Toplam tutar: yani $atutar$ ' değerinin yazdırılması sağlanır.

```
atutar= int(input("Alışveriş tutarını giriniz="))
if atutar<50:
 print("Kargo ücreti ile toplam tutar:", atutar+7)
else:
 print("Kargo bedava. Toplam tutar:",atutar)
```

ÖRNEK: Aşağıdaki kod örneğinde input() fonksiyonu kullanılarak kullanıcıdan bir sayı girilmesi istenir. Girilen değerin sayı olup olmadığını kontrol etmek için isdigit() yardımıyla bir if koşulu oluşturulur. Kodlamanın devamında bu sayının pozitif, negatif veya sıfır olma koşulunun kontrol edildiği koşullar ve bu koşulların sağlanması durumunda ekrana alınacak çıktılar print() yardımı ile yazılır.

Not: isdigit() metodu, bir karakter dizisinin (string) tamamen sayısal karakterlerden oluşup oluşmadığını kontrol eder. Eğer karakter dizisi tamamen sayısal ise, True değerini döndürür; aksi halde False döndürür.

```
#Kullanıcıdan bir sayı girişi almak
sayi_girisi = input("Bir sayı giriniz:")
```

```
Bir sayı giriniz:
```

```
#Girişin sayı olup olmadığını kontrol etmek
if sayi_girisi_isdigit():
    sayi = int(sayi_girisi)

#Sayının pozitif,negatif veya sıfır olma durumunu kontrol etmek
if sayi>0:
    print("Girilen sayı pozitiftir.")
elif sayi<0:
    print("Girilen sayı negatiftir.")
else:
    print("Girilen sayı sıfırdır.")
else:
print("Lütfen geçerli bir sayı girin.")
```

Döngüler

A. **For Döngüsü:** Belirli bir aralıktaki (örneğin, liste, demet, dize) elemanları üzerinde döner. Python programlama dilindeki "for" döngüsü, belirli bir sayıda yinelemek amacıyla kullanılan bir kontrol yapısıdır ve genellikle bir dizi üzerinde iterasyon (tekrarlama)yapmak için tercih edilir. Her iterasyonda döngü, belirtilen dizinin elemanlarını tek tek ele alarak tanımlanan işlemleri uygular. Bu, veri koleksiyonları üzerinde gezinme ve her eleman üzerinde belirli bir işlemi gerçekleştirme ihtiyacı olan durumları etkili bir şekilde çözmek için kullanılır. [38]

Aşağıdaki örnekte for döngüsünün bir liste üzerindeki döngü çıktısı görülmektedir.

```
sayilar = [1,2,3,4,5]
for sayi in sayilar:
   print(sayi)
```

Ekran çıktısı = “1 2 3 4 5 “olacaktır.

ÖRNEK: Aşağıdaki kod örneğinde sayılar isimli karışık elemanlı bir liste oluşturulur. Birinci örnekte bu kümenin içindeki hangi sayının üçün katı olduğunu gösteren $sayi\%3 == 0$ for döngüsü oluşturulur. Bu döngünün çıktısı $3,9,12,21$ olacaktır. İkinci örnekte bu listedeki sayıların toplamını veren $+=$ for döngüsü oluşturulur. Üçüncü örnekte ise listedeki tek sayıların karesini alan matematiksel for döngüsü olan $sayi\%2 == 1$ döngüsü oluşturulur. Kod dizileri çalıştırıldığında bu soruların cevabını liste içinden çekerek çıktı olarak alınır.

sayilar = [1,3,5,7,9,12,19,21]

1- Sayilar listesindeki hangi sayılar 3' ün katıdır ?

```
for sayi in sayilar:
   if (sayi%3==0):
       print(sayi)
```

2- Sayilar listesinde sayıların toplamı kaçtır ?

```
toplam = 0
for sayi in sayilar:
    toplam += sayi
print('toplam:',toplam)
```

3- Sayilar listesindeki tek sayıların karesini alınız.

```
for sayi in sayilar:
   if (sayi % 2 == 1):
      print(sayi ** 2)
```

B. **While Döngüsü:** Bir koşulun sağlandığı sürece bir dizi ifadeyi tekrar tekrar çalıştırmak için kullanılan bir kontrol yapıdır. Genelde kaç kere tekrar edileceği önceden bilinmeyen durumlarda tercih edilir. Koşulun doğru olduğu sürece döngü sürekli devam eder, fakat koşul yanlış olduğunda döngü sona erer. [39]

ÖRNEK:Aşağıdaki örnekte bir sayı listesi oluşturulmuş ve bu liste üzerinden üç farklı durum incelenir. Birinci örnekte while döngüsü kullanılarak bu listedeki elemanların ekrana çıktısı alınan kod dizilimi görülür. İkinci örnekte başlangıç ve bitiş değerlerinin kullanıcıdan giriş alınması durumunda ekrana girilen değer aralığındaki tüm tek sayıların ekrana yazdırılmasının sonucunu veren $i < len(sayilar)$ while döngüsü oluşturulur. Üçüncü örnekte $i = 100$ olması durumunda $i > 0$ olacak şekilde azalarak sıralama yapan kod dizilimi görülür.

sayilar = [1,3,5,7,9,12,19,21]

1- Sayilar listesini while ile ekrana yazdırınız.

```
i = 0
while (i < len(sayilar)):
    print(sayilar[i])
    i += 1
```

2- Başlangıç ve bitiş değerlerini kullanıcıdan alıp aradaki tüm tek sayıları ekrana yazdırınız.

```
baslangic = int(input('başlangıç: '))
bitis = int(input('bitiş: '))

i = baslangic

while i < bitis:
    i += 1
    if (i % 2 == 1):
        print(i)
```

3- 1-100 arasındaki sayıları azalan şekilde yazdırınız.

```
i = 100
while i > 0:
    print(i)
    i -= 1
```

C. **Break ve Continue:** Döngülerde özel durumları kontrol etmek için kullanılır. break, döngüyü tamamen sonlandırırken, continue döngünün geri kalanını atlayarak bir sonraki iterasyona geçer. [37]

Break kullanımı: Bulunduğu döngüyü sonlandırılmasını sağlar. Programın kontrolü döngünün merkezinden sonra gelen ifadeye geçer. İç içe çalışan döngülerde yer alıyorsa, bulunduğu döngüyü sonlandırma işlemini gerçekleştirir.

```
i = 0
while (i<20):
    if(i==5):
        break
        print("i:",i)
        i+=1
```

Ekran çıktısı:

i: 0

i: 1

i: 2

i: 3

i: 4

Continue kullanımı: Sadece geçerli tekrarlama için bir döngüdeki kodun geride kalan kısmını atlamak için kullanılır. Döngü bitmez ama sonraki tekrarlamayla devam eder.

Aşağıdaki örnekte $1-10$ arası bir liste oluşturulmuş ve bu liste range() kullanılarak sıralanır. Ardından i'den i'ye bir for döngüsü oluşturulur. Bu döngü aksi söylenene kadar 1'den 10'a kadar sayıların sıralanmasının çıktısının alınması gerektiğini ifade eder. En son kod bloğunda oluşturulan if döngüsü ile belirtilen koşul sağlandığı sürece döngünün başa sarması gerektiği aradaki kodların atlanması gerektiği komutu verilir.

```
"""
İçinde 0,1,2,3,4,5,6,7,8,9,10 sayıları olan liste adında bir liste oluşturalım
"""
liste=list(range(0,11))
print(liste)

"""
Çıktı:

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
"""

# continue ifadesi kullanmadan listenin içinde dolaşalım
for i in liste:
    print("i: ",i)

"""
Çıktı:

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
"""


for i in liste:
    if (i==3 or i==5):
        continue # koşul sağlanıp if bloğuna girdiğinde aşağıdaki print'i çalıştırma döngünün en başına git demek
    print("i: ",i)
```

Ekran çıktısı:

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
i:  0
i:  1
i:  2
i:  3
i:  4
i:  5
i:  6
i:  7
i:  8
i:  9
i:  10
i:  0
i:  1
i:  2
i:  4
i:  6
i:  7
i:  8
i:  9
i:  10
>
```

D. **Range():** Belirli bir aralıktaki sayıları üretmek için kullanılır. [32]

ÖRNEK: range(5) fonksiyonu, 5'e kadar olan sayıları yani 0,1,2,3,4 değerlerini ifade eder.

```
for i in range(5):
    print(i)

#range(5) 0'dan başlayıp 5'e kadar olan sayıları(5 hariç) ifade eder

"""
Bu programın çıktısı
0
1
2
3
4
"""
```

ÖRNEK: range(2,5) fonksiyonu 2'den başlayıp 5'e kadar olan sayıları 2,3,4 değerlerini ifade eder.

```
for i in range(2,5):
    print(i)

#range(2,5) 2'den başlayıp 5'e kadar olan sayıları(5 hariç) ifade eder

"""
Bu programın çıktısı

2
3
4
"""
```

3. PYTHON'DA KULLANILAN MATEMATİK KÜTÜPHANELERİ VE LİNEER CEBİRİN PYTHON'LA İLİŞKİSİ

3.1. NumPy Kütüphanesi

Numerical Python (Sayısal Python), çok boyutlu dizileri işlemek için kullanılan popüler bir veri bilimi kütüphanesidir. Jim Hugunin tarafından geliştirilen Numeric ve Numarray kütüphanelerinin özellikleri kullanılarak 2005 yılında Travis Oliphant tarafından geliştirilmiştir. Açık kaynak kod olduğu için birçok katılımcısı vardır ve gelişmeye devam etmektedir. NumPy, diziler üzerine matematiksel ve mantıksal işlemler yapılır. Hesaplamaları daha kolay, hızlı ve esnek bir şekilde yapmamızı sağlar.[14] NumPy kütüphanesini kullanmak için import etmemiz gerekiyor, import etmek için kullandığımız IDE'ye:

```
In [1]: import numpy as np
```

yazılıp çalışıldı. 'np' bundan sonra NumPy kullanmak istediğimiz yerlerde kısaca kullanmak için verdiğimiz isimdir. NumPy array'i Python'nın veri tiplerinden olan listeye oldukça benzer yapıdadır. Aralarındaki fark NumPy array'i verileri sabit tip (fix type) veri ile tutar. NumPy dizilerinin veri türü ndarray'dir.

3.1.1. NumPy Array'i Oluşturma

Tek boyutlu NumPy array'i oluşturma işlemi için np.array metodu kullanılır, içerisine liste veri tipinde değişken içermelidir. Aşağıdaki örnekte görüldüğü üzere np.array yazdıktan sonra normal parantez içerisine liste veri tipindeki değişken yazılarak bir NumPy array elde edebiliriz.

```
In [8]: x  = np.array([0,1,2,3,4,5])

In [9]: x
Out[9]: array([0, 1, 2, 3, 4, 5])
```

İki boyutlu NumPy array'i oluşturma işlemi için birden fazla listeye ihtiyacımız var, listelerin hepsini ortak bir liste içerisine alarak iki boyutlu NumPy array'i elde edilebilir. Aşağıda örnek olarak verilmiştir.

```
In [12]: y = np.array([[1,0,0],[0,2,0],[0,0,2]])

In [13]: y
Out[13]: array([[1, 0, 0],
                [0, 2, 0],
                [0, 0, 2]])
```

3.1.2. NumPy'da Eleman Seçme İşlemi

NumPy array indeksleri sıfırdan başlar. Tersten saymaya başlarsak yani sağdan sola doğru indeksler -1 'den başlayarak devam eder, negatif indeks olarak adlandırılır.
Tek boyutlu NumPy array içinden eleman seçme işlemi için atama yaptığımız değişkenin adı yanına köşeli parantez içerisinde hangi elemana erişmek isteniyorsa onun indeks numarasını yazılması yeterlidir.[41] Aşağıda örnek olarak verilen ilk kod satırında da görüldüğü üzere ilk önce 2.indeksteki eleman seçilir, ikinci kod satırında ise –2.indeksteki eleman seçilir.

```
In [37]: x[2]
         #Tek boyutlu numpy array'inde eleman seçme
Out[37]: 2

In [28]: x[-2]
Out[28]: 4
```

İki boyutlu NumPy array liste içerisinde liste içerir bundan dolayı eleman içerisinde eleman içerir burada eleman seçme işlemini yaparken öncelikle seçmek istenilen elemanın listesi seçilmeli yani eleman hangi indeksteki liste içerisinde ise o liste seçildikten sonra elemanın indeksi seçilir. Aşağıda örnek olarak verilen koda da görüldüğü üzere ilk önce 0.indeksteki eleman yani liste seçilir daha sonra seçilen liste içerisindeki 2.indeksteki eleman seçilir.

```
In [19]: y[0]
Out[19]: array([1, 0, 0])

In [20]: y[0][2]
Out[20]: 0
```

3.1.3. NumPy'daki Veri Tipleri ve Veri Tipi Değiştirme İşlemi

Python'da var olan veri tipleri; strings , integer, float, boolean, complex'ti. NumPy array'lerde bazı ekstra veri tipleri vardır. Bunlar; tam sayı (i), boolean (b) , işaretsiz tam sayı (u), ondalıklı sayı (f), kompleks sayı (c), saat (zaman) veri tipi (m), tarih veri tipi (M), nesne (O), string (S), unicode string (U).[40]

NumPy'da veri tipini değiştirmek için NumPy array'in sonuna 'dtype' argümanını girerek array'i istediğimiz veri tipine dönüştürebiliriz.[40] Aşağıdaki örnekte NumPy array'inin veri tipinin float yapmak istersek dtype = float şeklinde Numpy array'inin içine yazılır, böylece NumPy array'inin içerisindeki tüm elemanlar float veri tipine dönüşür.

```
In [60]: x = np.array([0,1,2,3,4,5], dtype = float)

In [61]: x
Out[61]: array([0., 1., 2., 3., 4., 5.])
```

3.1.4. NumPy Kütüphanesinde Kullanılan Bazı Metotlar

NumPy array içerisindeki toplam eleman sayısına erişmek için .size() metodunu kullanılır.[14] Aşağıdaki örnekte de verildiği üzere bir NumPy array'inin sonuna .size eklenir ve böylece NumPy array'inin toplam eleman sayısı elde edilir.

```
In [15]: x.size
Out[15]: 6
```

NumPy array'inin boyut sayısına erişmek için .ndim() metodunu kullanılır. [14] Aşağıdaki örnekte de verildiği üzere bir NumPy array'inin sonuna .ndim eklenir ve böylece NumPy array'inin boyut sayısını elde edilir.

```
In [17]: x.ndim
Out[17]: 1
```

NumPy array'inin boyut bilgisine erişmek için .shape() metodunu kullanılır. [14] Aşağıdaki örnekte de verildiği üzere bir NumPy array'inin sonuna .shape eklenir ve böylece NumPy array'inin boyut bilgisi elde edilir.

```
In [10]: x.shape
Out[10]: (6,)
```

NumPy array'nin transpozuna erişmek için .T metodunu kullanılır.[14] Aşağıdaki örnekte de verildiği üzere bir NumPy array'inin sonuna .T eklenir ve böylece NumPy array'inin transpozu elde edilir.

```
In [20]: z = np.array([[1,2,4],[5,3,9],[6,0,7]])

In [21]: z
Out[21]: array([[1, 2, 4],
                [5, 3, 9],
                [6, 0, 7]])

In [22]: z.T
Out[22]: array([[1, 5, 6],
                [2, 3, 0],
                [4, 9, 7]])
```

NumPy array'nin boyutunu değiştirmek için .reshape() metodunu kullanılır.[14] Burada dikkat edilmesi gerekilen şey reshape içine yazdığımız sayıların çarpımı eleman sayısına eşit olmalıdır.[14] Aşağıdaki örnekte de verildiği üzere bir NumPy array'inin sonuna .reshape eklenir ve içerisine istenilen boyut bilgisi girilir burada 9 satırlık 1 sütunluk bir NumPy array'i elde edilir. İkinci kod satırında ise 1 satırlık 9 sütunluk bir NumPy array'i elde edilir.

```
In [28]: y.reshape(9,1)
Out[28]: array([[1],
                [0],
                [0],
                [0],
                [2],
                [0],
                [0],
                [0],
                [2]])

In [29]: y.reshape(1,9)
Out[29]: array([[1, 0, 0, 0, 2, 0, 0, 0, 2]])
```

Belirttiğimiz sayı kadar sıfırlardan oluşan matris oluşturmak için .zeros() metodunu kullanılır.[14] 'dtype' ön tanımlı olarak ondalıklı (float) sayıdır.[14] Aşağıdaki örnekte de verildiği üzere bir np.zeros yazılır içerisine 10 sayısı girildiğinde on sütunluk bir satırlık ve tipi integer olan elemanları sıfır olan bir NumPy array'i elde edilir. İkinci kod satırında ise np.zeros yazılır içerisindeki (5,4) beş satırlık dört sütunluk ve veri tipi integer olan elemanları sıfır olan bir NumPy array'i elde edilir.

```
In [25]: np.zeros(10 , dtype = int)
Out[25]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0])

In [14]: np.zeros((5,4) , dtype = int)
         #Çok boyutlu sıfırlardan oluşan numpy array'i oluşur.
Out[14]: array([[0, 0, 0, 0],
                [0, 0, 0, 0],
                [0, 0, 0, 0],
                [0, 0, 0, 0],
                [0, 0, 0, 0]])
```

Belirttiğimiz sayı kadar birlerden oluşan matris oluşturmak için .ones() metodunu kullanılır.'dtype' ön tanımlı olarak ondalıklı (float) sayıdır.[14] Aşağıdaki örnekte de verildiği

üzere bir np.ones yazılır içerisine 3 sayısı girildiğinde üç sütunluk bir satırlık ve tipi integer olan elemanları bir olan bir NumPy array'i elde edilir. İkinci kod satırında ise np.ones yazılır içerisindeki (3,3) üç satırlık üç sütunluk ve veri tipi integer olan elemanları bir olan bir NumPy array'i elde edilir.

```
In [11]: np.ones(3 , dtype = int)
Out[11]: array([1, 1, 1])

In [18]: np.ones((3,3), dtype = int)
         #Çok boyutlu birlerden oluşan numpy array'i oluşur.
Out[18]: array([[1, 1, 1],
                [1, 1, 1],
                [1, 1, 1]])
```

'.random' metodunun birçok kullanım çeşidi vardır. Biz burada sadece iki tanesini göstereceğiz.

Tam sayılardan oluşan random matris oluşturmak için .random.randint() metodunu kullanılır. Belli bir aralık ve kaç elemandan oluşturulması gerektiğini belirtmeliyiz.[14] Aşağıdaki örnekte de verildiği üzere np.random.randint yazılır içerisindeki sıfır başlangıç sayımız on bitiş sayımızdır size=9 ise 9 elemandan oluşmasını ifade eder, böylece 0 ile 10 arasında rastgele 9 elemandan oluşan veri tipi integer olan bir NumPy array'i elde edilir.

```
In [27]: np.random.randint(0, 10, size = 9) # 0 ile 10 arasında 9 elemanlı matris oluşur
Out[27]: array([9, 3, 9, 8, 7, 3, 5, 7, 1])
```

Birim matris oluşturmak için .eye() metodunu kullanılır.[14] Aşağıdaki örnekte de verildiği üzere np.eye içerisine beş yazılır böylece beş satırlı beş sütunlu veri tipi integer olan bir NumPy array'i elde edilir.

```
In [9]: np.eye(5 , dtype = int)
Out[9]: array([[1, 0, 0, 0, 0],
               [0, 1, 0, 0, 0],
               [0, 0, 1, 0, 0],
               [0, 0, 0, 1, 0],
               [0, 0, 0, 0, 1]])
```

Belirli bir aralıkta doğrusal olarak aralıklandırılmış değerlere sahip biri NumPy array'i oluşturmak için .linspace() metodunu kullanılır. Bu metod üç argüman girmemizi ister. İlk argümana başlangıç değerimizi, ikinci argümana bitiş değerimizi gireriz. Üçüncü argümanımıza başlangıç ve bitiş değerleri arasındaki sayıları kaça bölmek istediğimizi gireriz, üçüncü argümana göre aralıkları eşit parçalara böler.[14] Aşağıdaki örnekte de verildiği üzere np.linspace içerisine yazılan iki başlangıç değerimizi, üç bitiş değerimizi, beş eleman sayımızı ifade eder, böylece iki ile üç arasında beş tane veri tipi ondalıklı sayı (float) olan bir NumPy array'i elde ederiz.

```
In [69]: np.linspace(2,3,5)
Out[69]: array([2.  , 2.25, 2.5 , 2.75, 3.  ])
```

Girilen sayı ve verilen boyut bilgisine göre NumPy array'i oluşturmak için .full() metodunu kullanılır.[14] Aşağıdaki örnekte de verildiği üzere np.full içerisine yazılan üç boyut bilgisini, beş eleman değerini ifade eder, böylece üç satırlı bir sütunlu elemanları beş olan bir NumPy array'i elde edilir. İkinci kod satırında ise np.full içerisine yazılan (2,3) boyut bilgisini, beş

eleman değerini ifade eder, böylece iki satırlı üç sütunlu elemanları beş olan bir NumPy array'i elde edilir.

```
In [70]: np.full(3,5)
         #Tek boyutlu array

Out[70]: array([5, 5, 5])

In [71]: np.full((2,3),5)
         #İki boyutlu array

Out[71]: array([[5, 5, 5],
                [5, 5, 5]])
```

İki tane tek boyutlu NumPy array'ini iki boyutlu iki dizi yapmak için .meshgrid() metodu kullanılır.[14] Aşağıdaki örnekte x ve y NumPy array'leri oluşturulmuştur, np.meshgrid içerisine yazılır. Bu metot bize iki tane NumPy array'i oluşturacağı için isimlendirme işlemi için de iki tane değer girilmelidir. Bu örnekte X ve Y olarak adlandırılmıştır.

```
In [35]: x = np.linspace(-10,10,9)

In [36]: y = np.linspace(10,20,9)

In [37]: X,Y = np.meshgrid(x,y)

In [38]: X

Out[38]: array([[-10. ,  -7.5,  -5. ,  -2.5,   0. ,   2.5,   5. ,   7.5,  10. ],
                [-10. ,  -7.5,  -5. ,  -2.5,   0. ,   2.5,   5. ,   7.5,  10. ],
                [-10. ,  -7.5,  -5. ,  -2.5,   0. ,   2.5,   5. ,   7.5,  10. ],
                [-10. ,  -7.5,  -5. ,  -2.5,   0. ,   2.5,   5. ,   7.5,  10. ],
                [-10. ,  -7.5,  -5. ,  -2.5,   0. ,   2.5,   5. ,   7.5,  10. ],
                [-10. ,  -7.5,  -5. ,  -2.5,   0. ,   2.5,   5. ,   7.5,  10. ],
                [-10. ,  -7.5,  -5. ,  -2.5,   0. ,   2.5,   5. ,   7.5,  10. ],
                [-10. ,  -7.5,  -5. ,  -2.5,   0. ,   2.5,   5. ,   7.5,  10. ],
                [-10. ,  -7.5,  -5. ,  -2.5,   0. ,   2.5,   5. ,   7.5,  10. ]])

In [39]: Y

Out[39]: array([[10.  , 10.  , 10.  , 10.  , 10.  , 10.  , 10.  , 10.  , 10.  ],
                [11.25, 11.25, 11.25, 11.25, 11.25, 11.25, 11.25, 11.25, 11.25],
                [12.5 , 12.5 , 12.5 , 12.5 , 12.5 , 12.5 , 12.5 , 12.5 , 12.5 ],
                [13.75, 13.75, 13.75, 13.75, 13.75, 13.75, 13.75, 13.75, 13.75],
                [15.  , 15.  , 15.  , 15.  , 15.  , 15.  , 15.  , 15.  , 15.  ],
                [16.25, 16.25, 16.25, 16.25, 16.25, 16.25, 16.25, 16.25, 16.25],
                [17.5 , 17.5 , 17.5 , 17.5 , 17.5 , 17.5 , 17.5 , 17.5 , 17.5 ],
                [18.75, 18.75, 18.75, 18.75, 18.75, 18.75, 18.75, 18.75, 18.75],
                [20.  , 20.  , 20.  , 20.  , 20.  , 20.  , 20.  , 20.  , 20.  ]])
```

pi sayısına ulaşmak için np.pi() metodu kullanılır.[14] Aşağıdaki örnekte de verildiği üzere np.pi yazılır ve böylece pi sayısı elde edilir.

```
In [2]: np.pi

Out[2]: 3.141592653589793
```

3.1.5. NumPy'da Matematiksel İşlemler

NumPy array'ler çok boyutlu dizi olması bize birçok matematiksel işlemler yapmamızı sağlar. Bu kısımda aritmetik işlemler, trigonometrik işlemler, bazı istatistiksel işlemleri ele alacağız. Bunlar gibi birçok matematiksel, istatistiksel, polinom, işlemleri gerçekleştirilebilir.

3.1.5.1.Basit Aritmetik İşlemler

NumPy ‘da basit aritmetik işlemler olarak ; toplama, çıkarma, çarpma, bölme ve üs alma işlemlerini ele alacağız. NumPy’da yaptığımız tüm basit aritmetik işlemler matrisler üzerinde karşılık gelen yöndeş elemanlar üzerinden yapılır.

3.1.5.1.1. Toplama : NumPy’da toplama işlemini iki şekilde gerçekleştirebiliriz; aritmetik operatör olan ’+’ ile matris-matris toplamı ve matris-skaler toplamı yapmamızı sağlar. Bir diğer yöntem ise np.add() fonksiyonunu kullanmaktır. Aşağıdaki örnekte de verildiği üzere .reshape metodu yardımıyla iki satırlık, iki sütunluk x ve y NumPy array’i oluşturulur. Beşinci kod satırında x NumPy array’ini bir ile yani bir skaler ile toplama işlemi yapılır ve yeni bir NumPy array'i elde edilir. Altıncı kod satırında x ile y NumPy array’lerinin toplama işlemi yapılır ve yeni bir NumPy array'i elde edilir. Yedinci kod satırında np.add metodu ile x ve y NumPy array’lerinin toplama işlemi yapılır ve yeni bir NumPy array'i elde edilir. Sekizinci kod satırında np.add metodu ile x NumPy array’inin iki ile yani bir skaler ile toplama işlemi yapılır ve yeni bir NumPy array'i elde edilir.

```
In [4]: x = np.array([1,2,3,4]).reshape(2,2)

In [5]: x
Out[5]: array([[1, 2],
               [3, 4]])

In [6]: y = np.array([5,6,7,8]).reshape(2,2)

In [7]: y
Out[7]: array([[5, 6],
               [7, 8]])

In [8]: x + 1
Out[8]: array([[2, 3],
               [4, 5]])

In [9]: x + y
Out[9]: array([[ 6,  8],
               [10, 12]])

In [10]: np.add(x, y)
Out[10]: array([[ 6,  8],
                [10, 12]])

In [16]: np.add(x, 2)
Out[16]: array([[3, 4],
                [5, 6]])
```

3.1.5.1.2. Çıkarma : NumPy’da çıkarma işlemini iki şekilde gerçekleştirebiliriz; aritmetik operatör olan ’ -’ ile matris-matris farkı ve matris-skaler farkını almamızı sağlar. Bir diğer yöntem ise np.subtract() fonksiyonunu kullanmaktır.[14] Aşağıdaki örnekte de verildiği üzere birinci kod satırında x NumPy array’ini iki ile yani bir skaler ile çıkarma işlemi yapılır ve yeni bir NumPy array'i elde edilir. İkinci kod satırında y ile x NumPy array’lerinin çıkarma işlemi yapılır ve yeni bir NumPy array’i elde edilir. Üçüncü kod satırında np.subtract metodu ile y ve x NumPy array’lerinin çıkarma işlemi yapılır ve yeni bir NumPy array’i elde edilir. Dördüncü kod satırında np.subtract metodu ile x NumPy array’inin iki ile yani bir skaler ile çıkarma işlemi yapılır ve yeni bir NumPy array’i elde edilir.

```
In [11]: x - 2
Out[11]: array([[-1,  0],
                [ 1,  2]])

In [12]: y - x
Out[12]: array([[4, 4],
                [4, 4]])

In [15]: np.subtract(y, x)
Out[15]: array([[-4, -4],
                [-4, -4]])

In [17]: np.subtract(x, 2)
Out[17]: array([[-1,  0],
                [ 1,  2]])
```

3.1.5.1.3. Çarpma : NumPy'da çarpma işlemini iki şekilde gerçekleştirebiliriz; aritmetik operatör olan '*' ile matris-matris çarpımını ve matris-skaler çarpımını yapmamızı sağlar. Bir diğer yöntem ise np.multiply() fonksiyonunu kullanmaktır.[14] Aşağıdaki örnekte de verildiği üzere birinci kod satırında x NumPy array'ini üç ile yani bir skaler ile çarpma işlemi yapılır ve yeni bir NumPy array'i elde edilir. İkinci kod satırında x ile y NumPy array'lerinin çarpma işlemi yapılı ve yeni bir NumPy array'i elde edilir. Üçüncü kod satırında np.multiply metodu ile x NumPy array'inin üç ile yani bir skaler ile çarpma işlemi yapılır ve yeni bir NumPy array'i elde edilir. Dördüncü kod satırında np.multiply metodu ile x ve y NumPy array'lerinin çarpma işlemi yapılır ve yeni bir NumPy array'i elde edilir.

```
In [18]: x * 3
Out[18]: array([[ 3,  6],
                [ 9, 12]])

In [19]: x * y
Out[19]: array([[ 5, 12],
                [21, 32]])

In [20]: np.multiply(x , 3)
Out[20]: array([[ 3,  6],
                [ 9, 12]])

In [21]: np.multiply(x , y)
Out[21]: array([[ 5, 12],
                [21, 32]])
```

3.1.5.1.4. Bölme : NumPy'da bölme işlemini iki şekilde gerçekleştirebiliriz; aritmetik operatör olan '/' ile matris-matris ve matris-skaler bölme işlemini yapmamızı sağlar. Bir diğer yöntem ise np.divide() fonksiyonunu kullanmaktır.[14] Aşağıdaki örnekte de verildiği üzere birinci kod satırında x NumPy array'ini iki ile yani bir skaler ile bölme işlemi yapılır ve yeni bir NumPy array'i elde edilir. İkinci kod satırında x ile y NumPy array'lerinin bölme işlemi yapılır ve yeni bir NumPy array'i elde edilir. Üçüncü kod satırında np.divide metodu ile x NumPy array'inin iki ile yani bir skaler ile bölme işlemi yapılır ve yeni bir NumPy array'i elde edilir. Dördüncü kod satırında np.divide metodu ile x ve y NumPy array'lerinin bölme işlemi yapılır ve yeni bir NumPy array'i elde edilir.

```
In [24]: x / 2
Out[24]: array([[0.5, 1. ],
                [1.5, 2. ]])

In [25]: x / y
Out[25]: array([[0.2       , 0.33333333],
                [0.42857143, 0.5       ]])

In [26]: np.divide(x, 2)
Out[26]: array([[0.5, 1. ],
                [1.5, 2. ]])

In [27]: np.divide ( y, x)
Out[27]: array([[5.        , 3.        ],
                [2.33333333, 2.        ]])
```

3.1.5.1.5. Üs Alma : NumPy'da üs işlemini iki şekilde gerçekleştirebiliriz; aritmetik operatör olan ' ** ' ile matris-matris ve matris-skaler üssünü almamızı sağlar. Bir diğer yöntem ise np.power() fonksiyonunu kullanmaktır.[14] Aşağıdaki örnekte de verildiği üzere birinci kod satırında x NumPy array'ini üçüncü kuvveti yani bir skaler ile kuvvet alma işlemi yapılır ve yeni bir NumPy array'i elde edilir. İkinci kod np.power metodu ile x NumPy array'inin üç ile yani bir skaler ile kuvvet alma işlemi yapılır ve yeni bir NumPy array'i elde edilir.

```
In [28]: x ** 3
Out[28]: array([[ 1,  8],
                [27, 64]], dtype=int32)

In [29]: x ** y
Out[29]: array([[    1,    64],
                [ 2187, 65536]])

In [30]: np.power(x, 3)
Out[30]: array([[ 1,  8],
                [27, 64]], dtype=int32)

In [31]: np.power(x,y)
Out[31]: array([[    1,    64],
                [ 2187, 65536]])
```

3.1.5.2.Trigonometrik İşlemler: NumPy, radyan cinsinden değerler alan ve sonuç olarak sin, cos ve tan değerlerini üreten ufunc sin(), cos(), tan(), sinh(), cosh(), tanh() sağlar.

3.1.5.2.1. sin() ve sinh() : np.sin metodu girilen değerin sin değerini verir. np.sinh metodu girilen değerin sinh değerini verir.[14] Aşağıdaki birinci kod satırında da verildiği üzere np.sin metodunun içerisine np.pi yani pi sayısı yazılır böylece pi sayısının sin değeri elde edilir. İkinci kod satırında ise np.sinh içerisine np.pi/2 yazılır böylece pi sayısının yarısının sinh değeri elde edilir.

```
In [52]: np.sin(np.pi)
Out[52]: 1.2246467991473532e-16
In [53]: np.sinh(np.pi/2)
Out[53]: 2.3012989023072947
```

3.1.5.2.2. cos() ve cosh() : np.cos metodu girilen değerin cos değerini verir. np.cosh metodu girilen değerin cosh değerini verir.[14] Aşağıdaki birinci kod satırında da verildiği üzere np.cos metodunun içerisine np.pi/3 yazılır böylece pi sayısının üçe bölümünün cos değeri elde edilir. İkinci kod satırında ise np.cosh içerisine np.pi/4 yazılır böylece pi sayısının dörde bölümünün cosh değeri elde edilir.

```
In [54]: np.cos(np.pi/3)
Out[54]: 0.5000000000000001
In [55]: np.cosh(np.pi/4)
Out[55]: 1.324609089252006
```

3.1.5.2.3. tan() ve tanh() : np.tan metodu girilen değerin tan değerini verir. np.tanh metodu girilen değerin tanh değerini verir.[14] Aşağıdaki birinci kod satırında da verildiği üzere np.sin metodunun içerisine np.pi/5 yazılır böylece pi sayısının beşe bölümünün tan değeri elde edilir. İkinci kod satırında ise np.tanh içerisine np.pi yazılır böylece pi sayısının tanh değeri elde edilir.

```
In [56]: np.tan(np.pi/5)
Out[56]: 0.7265425280053608
In [57]: np.tanh(np.pi)
Out[57]: 0.9962720762207499
```

3.2. Matplotlib Kütüphanesi

Matplotlib, Python'da görselleştirme için birçok araç içeren bir kütüphanedir. Matplotlib kütüphanesini kullanarak iki boyutlu ve üç boyutlu grafikler oluşturabiliriz. Genellikle Python'da grafik çizdirmek için matplotlib kütüphanesinde yer alan ve matplotlib'in Matlab stilinde grafik üretmesini sağlayan komutların modülü matplotlib.pyplot modülünü kullanacağız.[14] Matplotlib kütüphanesini kullanmak için import etmemiz lazım, import etmek için kullandığımız IDE'ye:

```
In [1]: import matplotlib.pyplot as plt
```

yazıp çalıştırıyoruz. 'plt' bundan sonra pyplot modülünü kullanmak istediğimiz yerlerde kısaca kullanmak için verdiğimiz isimdir.

3.2.1. Matplotlib Kütüphanesinde Kullanılan Bazı Metotlar

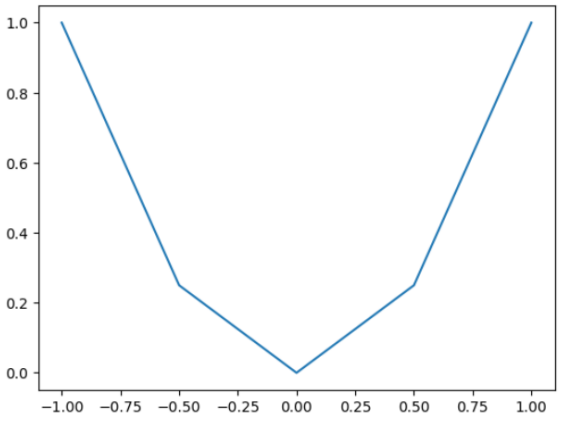
Matplotlib kütüphanesini görselleştirme amacı ile kullanılır. Görselleştirme işlemlerini çeşitli metotlar kullanarak geliştirilebilir ya da özelleştirilebilir. Aşağıda sık kullanılan bazı matplotlib kütüphanesinin metotlarına yer verilmiştir.

Grafiği çizdirmek için plt.show() metodu kullanılır. Kullandığınız IDE'ye göre bu kod satırı girilmediği takdirde grafik çizimi gerçekleşmez yani ekrana bir çıktı gelmez. [9]

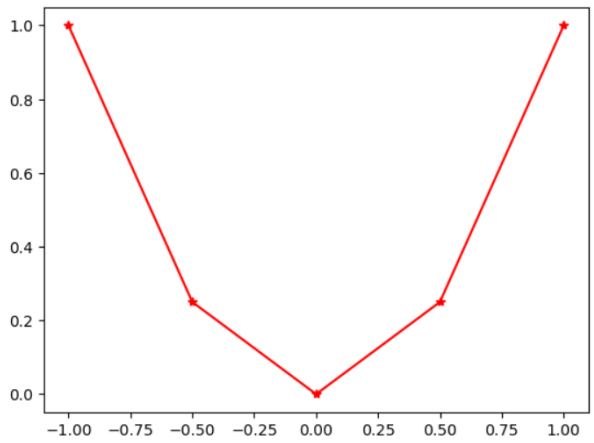
Verilen noktaları birleştirerek bir grafik elde etmek için plt.plot() metodu kullanılır. İlk iki argümana list nesnelerimizi gireriz. Üçüncü bir argüman girildiğinde grafiğin çizgi formatını belirtir iki ya da üç karakterli string olarak girilir. Bunlardan ilki rengi belirtir, ikincisi

karakter işaretleyici (marker) belirtir, üçüncüsü ise çizgi stilini belirtir. Bunları string ifade içerisine almadan da gösterebiliriz her biri için ayrı argüman girmemiz gerekir. Renk belirtmek için ‘color’ ya da ‘c’, karakter işaretleyici (marker) için ‘marker’ ve çizgi stili için ‘linestyle’ ya da ‘ls’ ek olarak çizgi kalınlığını belirtmek istersek ‘linewidth’ ya da ‘lw’ kullanılır ve float olarak girmemiz gerekir. Bunlar dışında grafiği özelleştirmek için bir çok argüman vardır.[9] Aşağıdaki örnekte x ve y listeleri oluşturulur. x ve y listelerini plt.plot içerisine yazılır böylece x ve y listeleri sırası ile girildiğinde x listesi x eksenini y listesi y eksenini ifade eden bir grafik elde edilir. İkinci kod satırında color = red grafiğin çizgisinin kırmızı olması gerektiğini belirtir. marker = * grafiğin her kırılımında * olması gerektiğini belirtir. linestle = - grafiğin çizgi stilinin - olması gerektiğini belirtir.

```
In [62]: x = [-1.0 , -0.5, 0.0, 0.5, 1.0]
In [63]: y = [1.0, 0.25, 0.0, 0.25, 1.0]
In [64]: plt.plot(x, y)
         plt.show()
```




```
In [37]: plt.plot(x,y, color = 'red' , marker ='*', linestyle = '-')
         plt.show()
```



Matplotlib’teki grafiklerde kullanılabilecek renkler: [3]

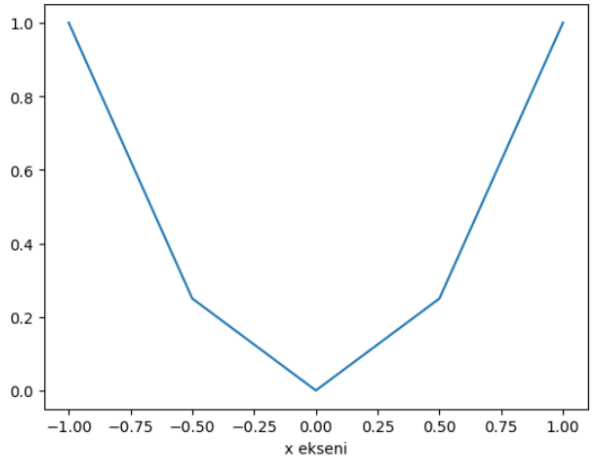
Matplotlib'teki grafiklerde kullanılabilecek markerlar: [3]

Karakter	Tanım	Karakter	Tanım
'.'	point marker	's'	square marker
','	pixel marker	'p'	pentagon marker
'o'	circle marker	'*'	star marker
'v'	triangle_down marker	'h'	hexagon1 marker
'^'	triangle_up marker	'H'	hexagon2 marker
'<'	triangle_left marker	'+'	plus marker
'>'	triangle_right marker	'x'	x marker
'1'	tri_down marker	'D'	diamond marker
'2'	tri_up marker	'd'	thin_diamond marker
'3'	tri_left marker	'|'	vline marker
'4'	tri_right marker	'_'	hline marker

x ekseninin başlığını oluşturmak için plt.xlabel() ya da .set_xlabel() metodu kullanılır.[9] Aşağıdaki örnekte daha önceden tanımladığımız x ve y listeleri kullanılır. plt.xlabel içerisine

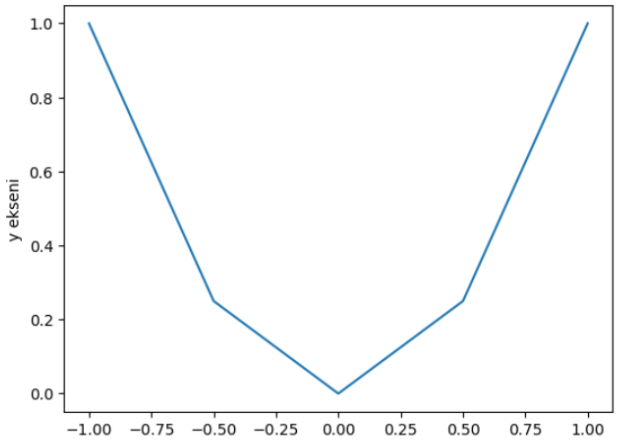
yazdığımız x ekseni yazısı x ekseninin başlığını oluşturur.

```
In [78]: plt.plot(x, y)
         plt.xlabel("x ekseni")
         plt.show()
```



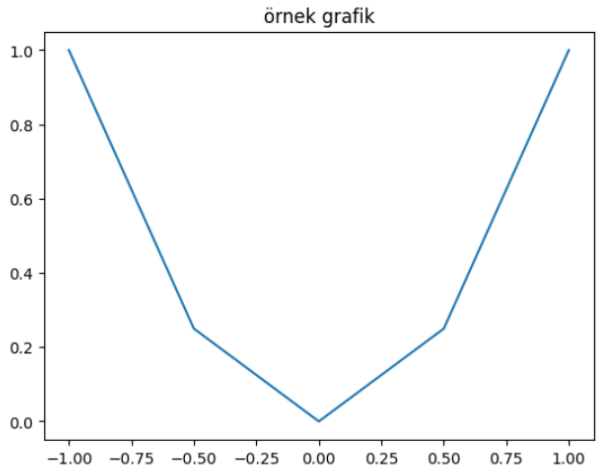
y ekseninin başlığını oluşturmak için plt.ylabel() ya da .set_ylabel() metodu kullanılır.[9] Aşağıdaki örnekte daha önceden tanımladığımız x ve y listeleri kullanılır. plt.ylabel içerisine yazdığımız y ekseni yazısı y ekseninin başlığını oluşturur.

```
In [79]: plt.plot(x,y)
         plt.ylabel("y ekseni")
         plt.show()
```



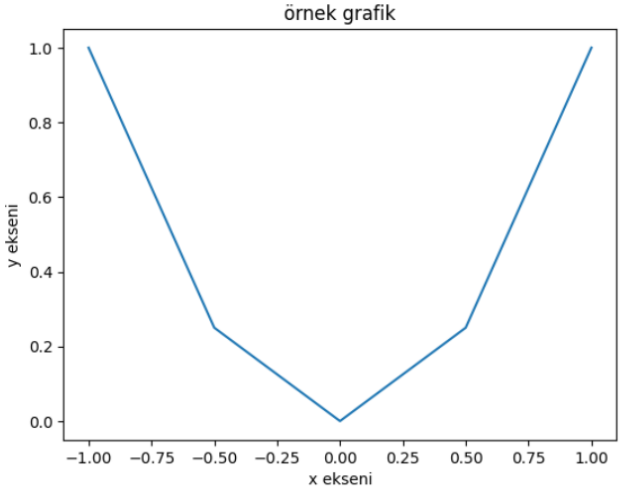
Grafiğin ana başlığını oluşturmak için plt.title() metodu kullanılır. [9] Aşağıdaki örnekte daha önceden tanımladığımız x ve y listeleri kullanılır. plt.title içerisine yazdığımız örnek grafik yazısı grafiğin genel başlığını oluşturur.

In [80]:
```
plt.plot(x,y)
plt.title("örnek grafik")
plt.show()
```



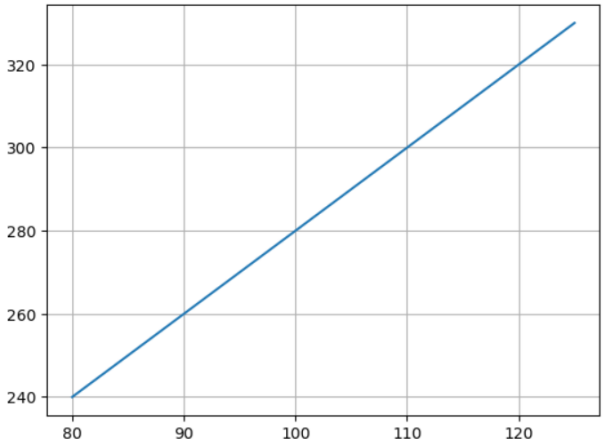
Hepsini bir arada görmek istersek;

In [81]:
```
plt.plot(x,y)
plt.xlabel("x ekseni")
plt.ylabel("y ekseni")
plt.title("örnek grafik")
plt.show()
```



Grafiğin arka kısmına ızgara deseni eklemek için plt.grid() metodu kullanılır.[9] Aşağıdaki örnekte x ve y NumPy array'leri oluşturulur. plt.plot metodunun içerisine x ve y NumPy array'leri sırasıyla yazılır. x NumPy array'i x eksenini y NumPy array'i y eksenini oluşturulur, daha sonrasından plt.grid metodu eklenir böylece grafiğin arka kısmına ızgara eklenir.

```
In [51]: x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])
         y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

         plt.plot(x, y)
         plt.grid()
         plt.show()
```



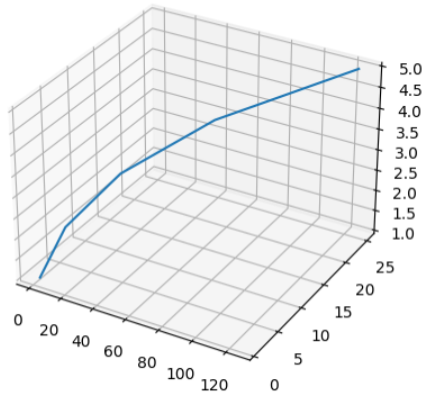
3.2.2. Matplotlib Kütüphanesinde Üç Boyutlu Grafikler

Matplotlib'de üç boyutlu grafikler çizdirmek için çalışmamıza 'from mpl_toolkits.mplot3d import Axes3D' eklememiz gerekmektedir. İki boyutlu grafik çizimlerinin metodu dışında burda bilmemiz gereken birkaç tane metod vardır.

Grafiğe eksenler eklemek için plt.axes() metodu kullanılır. Ayrıca 'projection' parametresi '3d' olarak belirtilmelidir, tüm üç boyutlu grafiklerde yapılmalıdır.[9] Aşağıdaki örnekte üç boyutlu grafiğin oluşturulması gösterilmiştir.

```
In [5]: x = [1, 2, 3, 4, 5]
        y = [1, 4, 9, 16, 25]
        z = [1, 8, 27, 64, 125]

        ax = plt.axes(projection = '3d')
        ax.plot3D(z, y, x)
```

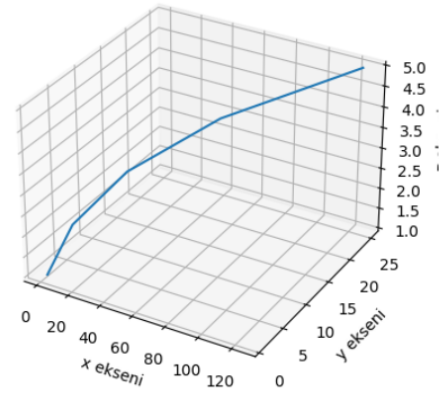
```
Out[5]: [<mpl_toolkits.mplot3d.art3d.Line3D at 0x132d91fce50>]
```



z ekseninin başlığını oluşturmak için plt.zlabel() ya da .set_zlabel() metodu kullanılır.[9] Aşağıdaki örnekte tanımladığımız x, y, z listeleri kullanılır. plt.zlabel içerisine yazdığımız z ekseni yazısı z ekseninin başlığını oluşturur.

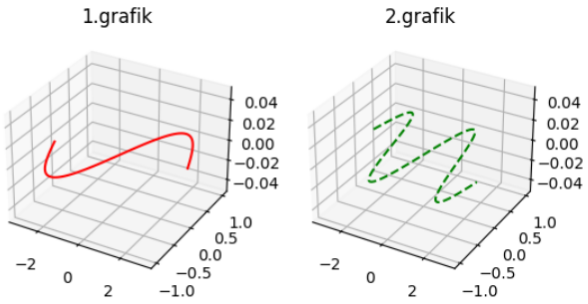
```
In [6]: x = [1, 2, 3, 4, 5]
        y = [1, 4, 9, 16, 25]
        z = [1, 8, 27, 64, 125]

        ax = plt.axes(projection = '3d')
        ax.plot3D(z, y, x)
        ax.set_xlabel("x ekseni")
        ax.set_ylabel("y ekseni")
        ax.set_zlabel("z ekseni")
```

```
Out[6]: Text(0.5, 0, 'z ekseni')
```



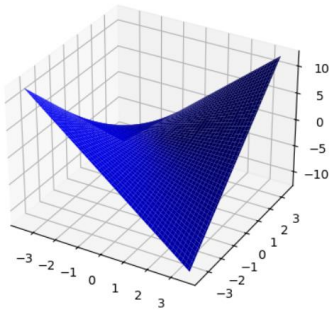
Üç boyutlu bir figure nesnesi içinde birden fazla plot nesnesi oluşturmak için .add_subplot() metodu kullanılır.[9] Örneğin iki grafik eklemek istiyorsak ilk grafik için x.subplot(121) ve ikinci grafik için ax.subplot(122) şeklinde kullanılabilir. Aşağıdaki örnekte tanımlanan d, f ve g'yi kullanarak iki tane plot nesnesini tek bir figure nesnesinde çıktısı alınması gösterilmiştir.

```
In [48]: d= np. linspace(-3, 3, 101)
         f = np.sin(d)
         g = np.sin(2*d)

         fig = plt.figure()
         ax = fig.add_subplot(121, projection = '3d')
         plt.plot(d, f, 'r-')
         plt.title("1.grafik")
         ax = fig.add_subplot(122, projection ='3d')
         plt.plot(d, g, 'g--')
         plt.title("2.grafik")
         plt.show()
```



Grafiğe renkli bir yüzey çizdirmek için .plot_surface() metodu kullanılır. Renklendirmeyi iki şekilde yapabiliriz. Birincisi color argümanını kullanmaktır. İkincisi, renk haritaları (color map) kullanmak, renk haritaları grafiğin yüksekliği değiştikçe rengide değişir. Renk haritalarını kullanmak için matplotlib kütüphanesi içerisinden ‘cm’ çalışmamıza eklememiz gerekmektedir. Ve ‘cm’ içerisinden ‘cmap’ argümanını ekleyerek üç boyutlu grafiğimizi renklendirebiliriz.[9] Aşağıdaki örnekte x ve y NumPy array’lerini np.meshgrid fonksiyonu kullanarak NumPy array’ini iki boyutlu iki dizi yapar ve bunlara X, Y isimlendirmesi yapılır. plt.axes metodu ile üç boyutlu bir grafik oluşturulur. Üç boyutlu grafiğin içerisine X, Y ve üçüncü argüman olarak X ve Y’nin çarpımını yazılır, color = ‘b’ grafiğimizin rengini ifade eder ‘b’ mavi rengini temsil eder.

```
In [50]: x = y = np.linspace(-3.5, 3.5, 100)
         X, Y = np.meshgrid(x, y, indexing='ij')

         fig = plt.figure()
         ax =  plt.axes(projection = '3d')
         ax.plot_surface(X, Y, X*Y, color = 'b')
         plt.show()
```



3.3. Sympy Kütüphanesi

SymPy kütüphanesi sembolik matematik işlemleri için kullanılan bir Python kütüphanesidir. SymPy kütüphanesi sayesinde birçok matematiksel işlemi kolayca yapmamızı sağlar.[11] SymPy kütüphanesini kullanmak için import etmemiz gerekiyor, import etmek için kullandığımız IDE'ye:

```
In [1]: import sympy as sp
```

yazıp çalışıldı. 'sp' bundan sonra SymPy kullanmak istediğimiz yerlerde kısaca kullanmak için verdiğimiz isimdir. Matematikte sembolik ifadeleri kullanmak için kullandığımız IDE'ye aşağıdaki kod satırını yazılır ve çalıştırılır. Bu sayede sembolik ifadeleri kullanılabilir hale gelir.

```
In [4]: from sympy import *
```

SymPy kütüphanesini kullanarak ifadeleri denklemdeki ifadeleri sembolleştirme işlemi için symbols metodu kullanılır ve aşağıda verilen şekilde yapılır.[8]

```
In [13]: x = symbols("x")
In [14]: x
Out[14]: x
```

Bir denklemi eşitlemek istediğimiz bir ifade varsa bunun için sp.Eq metodu kullanılır. İki argümandan oluşur ilk argümana denklem girilir, girilen denklem parantez içerisinde yazılır. İkinci argümana eşitlemek istediğimiz ifade girilir.[8]

```
In [54]: denklem = x**2 + 4*x + 4
In [55]: denklem
Out[55]: x^2 + 4x + 4
In [57]: denklem = sp.Eq((denklem), 0)
In [58]: denklem
Out[58]: x^2 + 4x + 4 = 0
```

Bir denklemi çözmek için sp.solve metodu kullanılır. İlk argümana denklem, ikinci argümana denklemde çözülmesini istediğimiz değişkeni girmeliyiz.[8]

```
In [64]: denklem = x**2 + 4*x + 4
In [65]: denklem
Out[65]: x^2 + 4x + 4
In [11]: denklem = sp.Eq((denklem), 0)
In [12]: denklem
Out[12]: x^2 + 4x + 4 = 0
In [13]: sp.solve(denklem,x)
Out[13]: [-2]
```

Burada tanımlanan denklem ve çözülmesi istenilen değişken sp.solve metodunun içerisine yazılır ve denklemin çözümü, kökleri elde edilir. [8]

Bir denklemi karekök içerisinde yazılması sembolleştirmek için sp.sqrt metodu kullanılır.[8]

```
In [20]: sp.sqrt(x)
Out[20]: √x
```

Burada sp.sqrt içerisine girilen denklemi matematiksel ifadesiyle elde edilir.
Limit ifadesinin sembolleştirmek için sp.Limit metodu kullanılır. İlk argümana denklem, ikinci argüman değişken, üçüncü argümana yaklaşılacak değer girilir.[8]

```
In [74]: sp.Limit(x**2, x,1)
Out[74]: lim_{x→1⁺} x²
```

Limit hesaplama işlemini gerçekleştirmek için sp.limit metodu kullanılır. İlk argümana denklem, ikinci argümana değişken, üçüncü argümana yaklaşılacak değer girilir.[8]

```
In [75]: sp.limit(x**2,x,1)
Out[75]: 1
```

Türev ifadesini sembolleştirmek için np.Derivative metodu kullanılır. İlk argümana denklem, ikinci argümana değişken, üçüncü argümana türevin derecesi girilir. [8]

```
In [89]: sp.Derivative(x**2, x,2)
Out[89]: d²/dx² x²
```

Türev alma işlemini gerçekleştirmek için sp.diff metodu kullanılır. İlk argüman denklemi, ikinci argüman hangi değişkene göre alınacağını belirtilir.[8]

```
In [17]: sp.diff(sin(x), x)
Out[17]: cos(x)
```

Burada girilen fonksiyonunun belirttiğimiz değişkene göre türevi alınır. Birden fazla kez türev almak istenirse, kaç kere türev alınması isteniyorsa türevin alınması istenilen değişken o kadar kez girilmelidir.

```
In [83]: sp.diff(sin(x), x,x)
Out[83]: −sin(x)
```

Bir diğer yöntem olarak değişkenden sonra üçüncü argüman olarak kaç kere türev almak istediğimizi nümerik olarak yazılır.

```
In [87]: sp.diff(sin(x), x,3)
Out[87]: −cos(x)
```

Birden fazla değişkene göre türev almak istenirse ikinci argümana değişkenlerin hepsi girilmesi gerekir.[8] Aşağıdaki örnekte gösterilmiştir.

```
In [33]: sp.diff(x**3/y**2, x,y)
```

Out[33]: $$-\frac{6x^2}{y^3}$$

İntegral ifadesini sembolleştirmek için sp.Integral metodu kullanılır. Alt ve üst sınırlar belirtilmezse belirsiz integral olarak elde ederiz.[8]

```
In [19]: sp.Integral(sqrt(1/x),x)
```

Out[19]: $$\int \sqrt{\frac{1}{x}}\, dx$$

Burada sp.Integral içerisine girilen denklemi belirttiğimiz değişkene göre matematiksel ifadesiyle elde edilir. Belirli integral ifadesinin sembolleştirilmesi isteniyorsa sırasıyla değişkeni, alt sınırı, üst sınırı parantez içerisine yazılır.[8]

```
In [34]: sp.Integral(sqrt(1/x),(x,1,3))
```

Out[34]: $$\int_1^3 \sqrt{\frac{1}{x}}\, dx$$

Burada sp.Integral içerisine girilen denklemi belirttiğimiz değişkene, alt sınır ve üst sınıra göre matematiksel ifadesiyle elde edilir.[8] İntegralin alt ve üst sınırları -∞, +∞ aralığında olabilir. Sonsuz değerlerin gösterimi için kod satırında -oo ve +oo şeklinde yazılır.[11] Aşağıda örnek olarak verilmiştir.

```
In [37]: sp.Integral(sqrt(1/x),(x,-oo,oo))
```

Out[37]: $$\int_{-\infty}^{\infty} \sqrt{\frac{1}{x}}\, dx$$

İntegral alma işlemini gerçekleştirmek için sp.integrate metodu kullanılır. İlk argümana denklem ikinci argümana değişken girilir. Eğer ki belirli integral ise ilk argümana denklem ikinci argümana parantez içerisinde sırasıyla değişken, alt sınır, üst sınır girilir.[8] Aşağıdaki örnek kod satırında belirsiz integral ve belirli integral için örnek kod satırı vardır, sp.integrate kullanarak belirtilen değişkene göre integral hesabı yapılır.[8]

```
In [70]: sp.integrate(sp.sqrt(1/x),x)
```

Out[70]: $$2x\sqrt{\frac{1}{x}}$$

```
In [71]: sp.integrate(sp.sqrt(1/x),(x,1,3))
```

Out[71]: $$-2 + 2\sqrt{3}$$

3.4. Lineer Cebirin Python'la İlişkisi

Lineer cebirin Python'da önemli bir yere sahiptir. Yapay zeka, makine öğrenmesi gibi birçok alanda kullanılmaktadır. Konumuz kapsamında matrisler ve vektörler üzerinde yapabileceğimiz birçok işlemi NumPy kütüphanesi içerisinde bulunan metodlar yardımlarıyla gerçekleştirebiliriz.

3.4.1. NumPy'da Matrisler

Python NumPy kütüphanesi içerisinde bulunan linalg modülü lineer cebirde matris ve vektör çarpımları, matris özdeğerlerini bulma, matris normunu bulma, matrisini tersine çevirme ve matris denklemi çözme gibi işlemleri yapmamızı sağlar.

3.4.2. Python NumPy Kütüphanesinde Kullanılan Bazı Lineer Cebir Metotları

Bu kısımda Python kütüphanesi olan NumPy'ın lineer cebir için kullanılan bazı metotlarını ele alacağız.
Bir kare Numpy array'inin (matrisinin) özdeğerlerini ve özvektörlerinin hesaplamak için np.linalg.eig() metodu kullanılır.[14] Aşağıdaki örnekte de verildiği üzere a NumPy array'i oluşturulur. np.linalg.eig metodunun içerisine yazılır ve a NumPy array'inin özdeğer ve özvektörleri elde edilir.

```
In [54]: a = [[1,0] ,[0,1]]

In [55]: v,d =np.linalg.eig(a)

In [56]: print("özvektör")
         print(v)
         print("özdeğerler")
         print(d)

         özvektör
         [1. 1.]
         özdeğerler
         [[1. 0.]
          [0. 1.]]
```

Genel bir matrisin,NumPy array'inin özdeğerlerini hesaplamak için np.linalg.eigvals() metodu kullanılır.[14] Aşağıdaki örnekte de verildiği üzere daha önceden oluşturduğumuz a NumPy array'ini np.linalg.eigvals metodunun içerisine yazılır ve a NumPy array'inin özdeğerleri elde edilir. İkinci kod satırında ise aynı işlem daha önce oluşturduğumuz b NumPy array'i için yapılır.

```
In [42]: np.linalg.eigvals(a)
Out[42]: array([1., 1.])

In [43]: np.linalg.eigvals(b)
Out[43]: array([4.73205081, 1.26794919])
```

NumPy array'inin (matrisin) normuna ulaşmak için np.linalg.norm() metodu kullanılır. [14] Aşağıdaki örnekte de verildiği üzere daha önceden oluşturduğumuz a NumPy array'ini np.linalg.norm metodunun içerisine yazılır ve a NumPy array'inin normu elde edilir. İkinci kod satırında ise aynı işlem daha önce oluşturduğumuz b NumPy array'i için yapılır.

```
In [44]: np.linalg.norm(a)
Out[44]: 1.4142135623730951

In [52]: np.linalg.norm(b)
Out[52]: 5.0
```

NumPy array'inin (matrisin) determinantını hesaplamak için np.linalg.det() metodu kullanılır.[14] Aşağıdaki örnekte de verildiği üzere ilk kod satırında daha önceden oluşturduğumuz a NumPy array'ini np.linalg.det metodunun içerisine yazılır ve a NumPy

array'inin determinantı elde edilir. İkinci kod satırında ise aynı işlem daha önce oluşturduğumuz b NumPy array'i için yapılır.

```
In [47]: np.linalg.det(a)
Out[47]: 1.0

In [48]: np.linalg.det(b)
Out[48]: 6.0
```

Doğrusal bir NumPy array'inin (matrisin) denklemini veya doğrusal skaler denklem sistemini çözmek için np.linslg.solve() metodu kullanılır.[14]

```
In [49]: np.linalg.solve(a,b)
Out[49]: array([[4., 1.],
                [2., 2.]])
```

NumPy array'inin (matrisin) tersine erişmek için np.linalg.inv() metodu kullanılır. [14] Aşağıdaki örnekte de görüldüğü üzere daha önceden oluşturduğumuz iki satırlık iki sütunluk a NumPy array'inin (matrisini) np.linalg.inv metodunun içerisine yazılır böylece a NumPy array'inin tersi elde edilir.

```
In [50]: np.linalg.inv(a)
Out[50]: array([[1., 0.],
                [0., 1.]])
```

NumPy array'inin (matrisin) transpozuna erişmek için np.transpose() metodu kullanılır.[14] Aşağıdaki örnekte de görüldüğü üzere üç sütunluk dört satırdan oluşan bir NumPy array'inin (matrisin) np.transpose metodunun içerisine yazılır ve transpozu elde edilir.

```
In [137]: a = np.array([[1, 3, 7, 2],
          [5, 8, -9, 0],
          [6, -7, 11, 12]])

In [138]: np.transpose(a)
Out[138]: array([[ 1,  5,  6],
                 [ 3,  8, -7],
                 [ 7, -9, 11],
                 [ 2,  0, 12]])
```

4. PYTHON'DA ANALİTİK GEOMETRİ UYGULAMALARI

4.1. Python İle Bazı Kuadriklerin Koniklerin Çizimi

4.1.1 Kürenin Python İle Çizimi

İlk adımda, Matplotlib kütüphanesini projemize dahil edelim. Numpy, matematiksel işlemler için ve matplotlib.pyplot, grafik çizimi için kullanılırken mpl_toolkits.mplot3d, 3D çizimler için kullanılır.

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

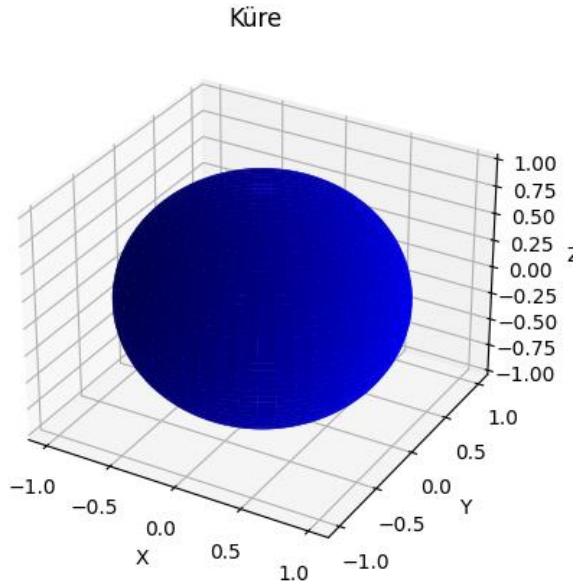
theta ve phi adında iki numPy dizisi oluşturalım. theta dizi, 0 ile 2*pi arasında 100 adet değeri içerirken phi dizi, 0 ile pi arasında 100 adet değeri içerir. Sonrasında bu hesaplamalar için uygun bir aralık seçelim. Kürenin matematiksel denklemini kullanarak, x, y ve z koordinatlarını hesaplayabiliriz.

```
theta = np.linspace(0, 2 * np.pi, 100)
phi = np.linspace(0, np.pi, 100)
theta, phi = np.meshgrid(theta, phi)
x = np.sin(phi) * np.cos(theta)
y = np.sin(phi) * np.sin(theta)
z = np.cos(phi)
```

plot_surface fonksiyonunu kullanarak x, y ve z koordinatlarıyla küreyi çizelim.Rengi mavi olarak belirleyelim.

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(x, y, z, color='b')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
plt.title('Küre')
plt.show()
```

Son olarak, grafiği görüntülemek için plt.show() fonksiyonunu kullanalım.



4.1.2 Elipsoidin Python İle Çizimi

İlk adımda, Matplotlib kütüphanesini projemize dahil edelim. Numpy, matematiksel işlemler için ve matplotlib.pyplot, grafik çizimi için kullanılırken mpl_toolkits.mplot3d, 3D çizimler için kullanılır.

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

Ardından, elipsoidin eksenlerini temsil eden a, b, c değerlerini belirleyelim.

```
a, b, c = 1.0, 2.0, 3.0
```

Daha sonra, u ve v adında iki adet array oluşturalım. u array'inde 0'dan 2*pi'ye kadar 100 adet nokta olacak şekilde değerler , v array'inde ise 0'dan pi'ye kadar 100 adet nokta olacak şekilde değerler elde ederiz.

Sonraki adımda, u ve v array'lerini kullanarak u-v düzleminde bir grid oluşturalım. Yani, her bir u ve v değeri için bir nokta belirlemiş oluyoruz.

```
u = np.linspace(0, 2 * np.pi, 100)
v = np.linspace(0, np.pi, 100)
u, v = np.meshgrid(u, v)
```

x, y, z adında üç adet array oluşturalım. Bu array'ler, elipsoidin 3D koordinatlarını temsil etmeli. x değeri a * cos(u) * sin(v), y değeri b * sin(u) * sin(v) ve z değeri c * cos(v) şeklinde hesaplanır.

```
x = a * np.cos(u) * np.sin(v)
y = b * np.sin(u) * np.sin(v)
z = c * np.cos(v)
```

Bir figure oluşturalım ve bu figure üzerinde 3D plot yapmak için bir subplot ekleyelim.

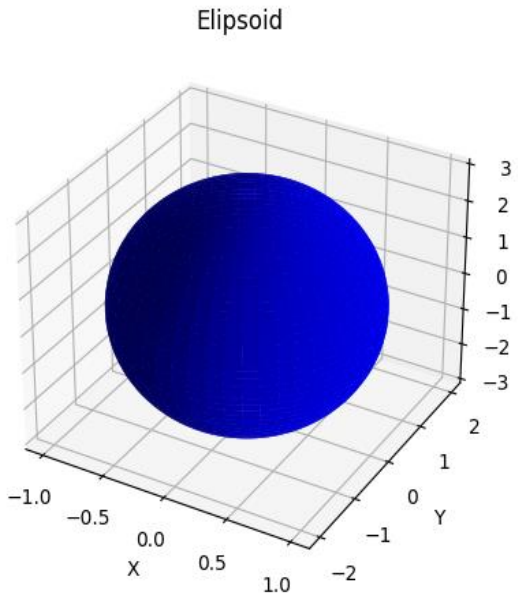
Oluşturulan 3D plot üzerinde x, y ve z array'lerini kullanarak bir surface plotu çizelim. Rengi mavi olarak belirleyelim.

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(x, y, z, color='b')
```

```
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')

plt.title('Elipsoid')
plt.show()
```

Plot'a x, y ve z eksenlerinin etiketlerini ekleyelim.Sonrasında Elipsoid'in çizildiği plot'un başlığını belirleyelim.

Son olarak,plot'u görüntüleyelim.



4.1.3 Hiperboloidin Python İle Çizimi

İlk olarak, numpy ve matplotlib.pyplot modüllerini import edelim. Bu modüller, sayısal hesaplama ve grafik çizimi için gerekli işlevleri sağlar.

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

Sonra, a, b, c gibi değişkenlere hiperboloidin parametrelerini atıyalım. Bu parametreler,

hiperboloidin boyutunu ve şeklini belirler.

```
a, b, c = 2.0, 1.0, 1.0
```

Sonraki adımda, u ve v adında iki adet numpy dizisi oluşturalım. Bu diziler, hiperboloidin 3D uzayındaki noktalarını oluşturmalı. u, 0 ile 2*pi arasında 100 eşit parçaya bölünmüş bir dizi, v ise -1 ile 1 arasında 100 eşit parçaya bölünmüş bir dizi olmalı.

```
u = np.linspace(0, 2 * np.pi, 100)
v = np.linspace(-1, 1, 100)
u, v = np.meshgrid(u, v)
```

Daha sonra, u ve v dizilerini kullanarak 3D uzayda hiperboloidin x, y ve z koordinatlarını hesaplıyalım. Bu hesaplama işlemleri, hiperboloidin denklemine dayanır.

```
x = a * np.cosh(v) * np.cos(u)
y = b * np.cosh(v) * np.sin(u)
z = c * np.sinh(v)
```

Şimdi, bir figür oluşturup 3D bir grafik çizmek için mpl_toolkits.mplot3d modülünden Axes3D sınıfını kullanıp,rengi mavi olarak belirtelim.

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(x, y, z, color='b')
```

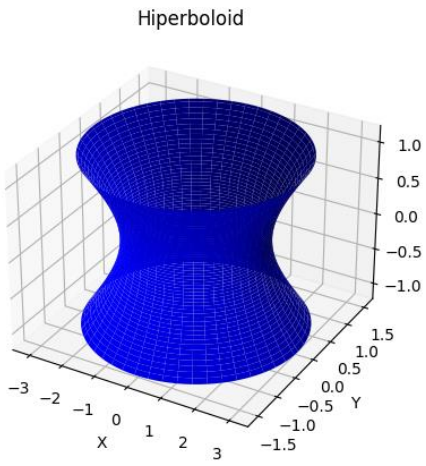
Oluşturduğumuz 3D grafiğe plot_surface işlevini kullanarak hiperboloidin yüzeyini çizelim. Bu işlev x, y ve z dizilerini alarak 3D uzayda bir yüzeyi grafik üzerinde temsil eder.

```
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
```

Grafiği daha anlaşılır hale getirmek için x, y ve z eksenlerine etiketler ekleyerek ve başlığı belirleyerek grafik üzerindeki açıklamaları ayarlayalım.

Son olarak, plt.show() işlevini kullanarak grafiği ekranda görüntüleyelim.

```
plt.title('Hiperboloid')
plt.show()
```



4.2 Python İle Genel Kuadriklerin Matris Formuyla Merkezil Hale Getirilmesi

Örnek 4.2.1.1

$\phi(x,y,z)=x^2+y^2+z^2+4xy+2y+4z+1=0$ kuadriğinin cinsini Python kullanarak belirleyiniz ve merkezil hale getiriniz.

Pythonla Çözümü:

In [1]:
```
import numpy as np
import sympy as sp
from sympy import *
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

In [2]:
```
# Kullanıcıdan kuadrk için katsayıları alalım :
print("Kullanıcıdan beklenen kuadrik formülü:Q(x,y,z)=(a11 )*x**+(a22)*y**+(a33)*z**+2*(a12)*xy+2*(a13)*xz+"
      "2*(a23)*yz+2*(a14)*x+2*(a24)*y+2*(a34)*z+(a44=0")

# Kullanıcıdan kuadrik denkleminin katsayılarını alalım :
a11 = int(input(" Kuadrik denklem  için a11 ifadesini giriniz: "))
a12 = int(input(" Kuadrik denklem  için a12 ifadesini giriniz: "))
a13 = int(input(" Kuadrik denklem  için a13 ifadesini giriniz: "))
a14 = int(input(" Kuadrik denklem  için a14 ifadesini giriniz: "))
a21 = int(input(" Kuadrik denklem  için a21 ifadesini giriniz: "))
a22 = int(input(" Kuadrik denklem  için a22 ifadesini giriniz: "))
a23 = int(input(" Kuadrik denklem  için a23 ifadesini giriniz: "))
a24 = int(input(" Kuadrik denklem  için a24 ifadesini giriniz: "))
a31 = int(input(" Kuadrik denklem  için a31 ifadesini giriniz: "))
a32 = int(input(" Kuadrik denklem  için a32 ifadesini giriniz: "))
a33 = int(input(" Kuadrik denklem  için a33 ifadesini giriniz: "))
a34 = int(input(" Kuadrik denklem  için a34 ifadesini giriniz: "))
a41 = int(input(" Kuadrik denklem  için a41 ifadesini giriniz: "))
a42 = int(input(" Kuadrik denklem  için a42 ifadesini giriniz: "))
a43 = int(input(" Kuadrik denklem  için a43 ifadesini giriniz: "))
a44 = int(input(" Kuadrik denklem  için a44 ifadesini giriniz: "))
```

```
Kullanıcıdan beklenen kuadrik formülü:Q(x,y,z)=(a11 )*x**+(a22)*y**+(a33)*z**+2*(a12)*xy+2*(a13)*xz+2*(a23)*yz+2*(a14)*x+2*(a2
4)*y+2*(a34)*z+(a44=0
 Kuadrik denklem  için a11 ifadesini giriniz: 1
 Kuadrik denklem  için a12 ifadesini giriniz: 4
 Kuadrik denklem  için a13 ifadesini giriniz: 0
 Kuadrik denklem  için a14 ifadesini giriniz: 0
 Kuadrik denklem  için a21 ifadesini giriniz: 0
 Kuadrik denklem  için a22 ifadesini giriniz: 1
 Kuadrik denklem  için a23 ifadesini giriniz: 0
 Kuadrik denklem  için a24 ifadesini giriniz: 2
 Kuadrik denklem  için a31 ifadesini giriniz: 0
 Kuadrik denklem  için a32 ifadesini giriniz: 0
 Kuadrik denklem  için a33 ifadesini giriniz: 1
 Kuadrik denklem  için a34 ifadesini giriniz: 4
 Kuadrik denklem  için a41 ifadesini giriniz: 0
 Kuadrik denklem  için a42 ifadesini giriniz: 0
 Kuadrik denklem  için a43 ifadesini giriniz: 0
 Kuadrik denklem  için a44 ifadesini giriniz: 1
```

In [3]:

```python
# Kuadratik denklemi yazdırma:
# Semboller:
x, y, z = sp.symbols('x y z ')
# Kuadratik denklemi tanımlama:
kuadrik_denklem = a11*x**2 + a12*x*y + a13*x*z + a14*x + a21*y*x + a22*y**2 + a23*y*z + a24*y + a31*z*x + a32*z*y + a33*z**2 + a3
# Sonucu yazdırma:
print("Kuadratik Denklemi:")
print(kuadrik_denklem)
# A matrisinin oluşturulması :
print("A matrisi: ")
A = np.array([ a11, a12/2, a13/2, a12/2, a22, a23/2, a13/2, a23/2, a33 ]).reshape(3,3)
print(A)
# Z matrisinin yazılması:
print("Z matrisi:")
Z = np.array([x,y,z]).reshape(3,1)
print(Z)
# B matrisinin yazılması :
print("B matrisi:")
B = np.array([2*a14 , 2*a24 , 2*a34]).reshape(3,1)
print(B)
# C matrisinin oluşturulması :
print("C matrisi : ")
C = np.array ([a11, a12/2, a13/2, a14/2, a21, a22, a23/2, a24/2, a31, a32, a33, a34/2, a41, a42, a43, a44]).reshape(4,4)
print(C)
```

```
#A matrisinin determinantı:
det_A = np.linalg.det(A)
print("A matrisinin determinantı:",det_A)
#C matrisinin determinantı :
det_C = np.linalg.det(C)
print("C matrisinin determinantı :",det_C)
# Kuadriğin sınıfını belirleme:
def kuadrik_siniflandirma(det_A, det_C):
    if det_C != 0 and det_A > 0:
        print ("Kuadrik, elipsoid sınıfındandır.")
    elif det_C != 0 and det_A < 0:
        print( "Kuadrik, hiperboloid sınıfındandır.")
    elif det_C != 0 and det_A == 0:
        print ("Kuadrik, paraboloid sınıfındandır.")
    elif det_C == 0 and det_A == 0:
        print( "Kuadrik, silindir sınıfındandır.")
    elif det_C == 0 and det_A != 0:
        print ("Kuadrik, koni sınıfındandır.")
    else:
        print ("Geçersiz değerler!")
kuadrik_siniflandirma(det_A, det_C)
```

```
Kuadratik Denklemi:
x**2 + 4*x*y + y**2 + 2*y + z**2 + 4*z + 1
A matrisi:
[[1. 2. 0.]
 [2. 1. 0.]
 [0. 0. 1.]]
Z matrisi:
[[x]
 [y]
 [z]]
B matrisi:
[[0]
 [4]
 [8]]
C matrisi :
[[1. 2. 0. 0.]
 [0. 1. 0. 1.]
 [0. 0. 1. 2.]
 [0. 0. 0. 1.]]
A matrisinin determinantı: -2.9999999999999996
C matrisinin determinantı : 1.0
Kuadrik, hiperboloid sınıfındandır.
```

In [4]:
```
print("Q(x,y,z)=0 kuadriğine x',y',z' terimlerini yok edecek şekilde :\n x=x'+x0 \n y=y'+y0 \n z=z'+z0 ötelemesi  uygulayalım.")
#Kuadrik denklemin x,y ve z'ye göre kısmi türevlerin alınması :
kuadrik_denlem_x_turev = sp.diff(kuadrik_denklem, x)
kuadrik_denlem_y_turev= sp.diff(kuadrik_denklem, y)
kuadrik_denlem_z_turev = sp.diff(kuadrik_denklem, z)
```

```
Q(x,y,z)=0 kuadriğine x',y',z' terimlerini yok edecek şekilde :
 x=x'+x0
 y=y'+y0
 z=z'+z0 ötelemesi  uygulayalım.
```

In [5]:
```
#Türevleri alınmış şekilde denklemleri yazdırma :
print("x'e göre türev alındığında oluşan denklem:")
print(kuadrik_denlem_x_turev)
print("y'e göre türev alındığında oluşan denklem:")
print(kuadrik_denlem_y_turev)
print("z'e göre türev alındığında oluşan denklem:")
print(kuadrik_denlem_z_turev)
print("Q = (x,y,z) olmak üzere:")
```

```
x'e göre türev alındığında oluşan denklem:
2*x + 4*y
y'e göre türev alındığında oluşan denklem:
4*x + 2*y + 2
z'e göre türev alındığında oluşan denklem:
2*z + 4
Q = (x,y,z) olmak üzere:
```

In [6]:
```
#Oluşan denklemleri sıfıra eşitleyelim:
denklem_x = sp.Eq(kuadrik_denlem_x_turev, 0)
denklem_y = sp.Eq(kuadrik_denlem_y_turev, 0)
denklem_z = sp.Eq(kuadrik_denlem_z_turev, 0)
print("x'e göre türev alındığında oluşan denklemin son hali:", denklem_x)
print("y'e göre türev alındığında oluşan denklemin son hali:",denklem_y)
print("z'e göre türev alındığında oluşan denklemin son hali:",denklem_z)
```

```
x'e göre türev alındığında oluşan denklemin son hali: Eq(2*x + 4*y, 0)
y'e göre türev alındığında oluşan denklemin son hali: Eq(4*x + 2*y + 2, 0)
z'e göre türev alındığında oluşan denklemin son hali: Eq(2*z + 4, 0)
```

In [7]:
```
p,q,r = sp.symbols('p q r ')
denklem_x = denklem_x.subs(x,p)
denklem_x = denklem_x.subs(y,q)
denklem_x = denklem_x.subs(z,r)
denklem_y = denklem_y.subs(x,p)
denklem_y = denklem_y.subs(y,q)
denklem_y = denklem_y.subs(z,r)
denklem_z = denklem_z.subs(x,p)
denklem_z = denklem_z.subs(y,q)
denklem_z = denklem_z.subs(z,r)
```

In [8]:
```
#Sıfıra eşitleyerek oluşturduğumuz denklemleri NumPy array'ine çevirelim:
denklem = [denklem_x, denklem_y, denklem_z]
print("Olusan denklem sistemi:")
print(denklem)
print("elde edilir. Buradan")
```

```
Olusan denklem sistemi:
[Eq(2*p + 4*q, 0), Eq(4*p + 2*q + 2, 0), Eq(2*r + 4, 0)]
elde edilir. Buradan
```

In [9]:
```
#NumPy array'ine dönüştürdüğümüz denklemleri çözelim:
cozum = sp.solve(denklem, (p,q,r))
print("Denklem sisteminin çözümü:", cozum)
print("x,y,z bulunur. Ayrıca")
```

```
Denklem sisteminin çözümü: {p: -2/3, q: 1/3, r: -2}
x,y,z bulunur. Ayrıca
```

In [10]:
```
cozum_p = cozum[p]
cozum_q = cozum[q]
cozum_r = cozum[r]
```

In [51]:
```
F1
```

Out[51]: -2.66666666666667

In [15]:
```
print("F' = " ,F1, "(F' ifadesini F1 olarak tanımlayacağız)" )
print(f"Böylece Q=(x,y,z)=0 kuadriğine;\n x=x'+{cozum_p} \n y=y'+{cozum_q} \n z=z'+{cozum_r}")
print(f" ötelemesi uygulanırsa:{a11}*x'**2 + {a22}*y'**2 + {a33}*z'**2 + {a12}*x'*y' + {a13}*x'*z' + {a23}*y'*z'+ {F1} = 0")
```

```
F' =  -2.66666666666667 (F' ifadesini F1 olarak tanımlayacağız)
Böylece Q=(x,y,z)=0 kuadriğine;
 x=x'+-2/3
 y=y'+1/3
 z=z'+-2
 ötelemesi uygulanırsa:1*x'**2 + 1*y'**2 + 1*z'**2 + 4*x'*y' + 0*x'*z' + 0*y'*z'+ -2.66666666666667 = 0
```

In [17]:
```
ozdegerler, ozvektorler = np.linalg.eig(A)
```

In [18]:
```
ozdegerler
```

Out[18]:
```
array([ 3., -1.,  1.])
```

In [19]:
```
ozvektorler
```

Out[19]:
```
array([[ 0.70710678, -0.70710678,  0.        ],
       [ 0.70710678,  0.70710678,  0.        ],
       [ 0.        ,  0.        ,  1.        ]])
```

In [79]:
```
#oluşması gereken P matirisi:
P_olması_gereken = np.array([0, 1/sp.sqrt(2) , 1/sp.sqrt(2) , 0, -1/sp.sqrt(2) , 1/sp.sqrt(2) , 1, 0, 0]).reshape(3,3)
```

In [80]:
```
P_olması_gereken
```

Out[80]:
```
array([[0, sqrt(2)/2, sqrt(2)/2],
       [0, -sqrt(2)/2, sqrt(2)/2],
       [1, 0, 0]], dtype=object)
```

In [20]:
```
ozvektorler.astype(int)
```

Out[20]:
```
array([[0, 0, 0],
       [0, 0, 0],
       [0, 0, 1]])
```

In [61]:
```
#P matrisini oluşturma
P = ozvektorler
print("P matrisi:")
print(P)
```

```
P matrisi:
[[ 0.70710678 -0.70710678  0.        ]
 [ 0.70710678  0.70710678  0.        ]
 [ 0.          0.          1.        ]]
```

In [62]:
```
#P matrisinin transpozesi
PT = P.T
print("P matrisinin transpozesini PT ile göstereceğiz:")
print(PT)
```

```
P matrisinin transpozesini PT ile göstereceğiz:
[[ 0.70710678  0.70710678  0.        ]
 [-0.70710678  0.70710678  0.        ]
 [ 0.          0.          1.        ]]
```

In [63]:
```
# p transpoze  *  a matrisi
D_PTA = PT.dot(A)
D_PTA
```

Out[63]:
```
array([[ 2.12132034,  2.12132034,  0.        ],
       [ 0.70710678, -0.70710678,  0.        ],
       [ 0.        ,  0.        ,  1.        ]])
```

In [88]:
```
#p transpoze * a matrisi * p
D_PTAP = D_PTA.dot(P)
D_PTAP
```

Out[88]:
```
array([[ 3.00000000e+00,  4.44089210e-16,  0.00000000e+00],
       [ 6.10622664e-16, -1.00000000e+00,  0.00000000e+00],
       [ 0.00000000e+00,  0.00000000e+00,  1.00000000e+00]])
```

In [89]:
```
#D (PT*A*P) matrisini hesaplama:
print("PT * A * P = D için D matrisi:")
print(D_PTAP)
```

```
PT * A * P = D için D matrisi:
[[ 3.00000000e+00  4.44089210e-16  0.00000000e+00]
 [ 6.10622664e-16 -1.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  1.00000000e+00]]
```

In [83]:
```
#D matrisi olması gereken :
D_olması_gereken = np.array([1,0,0,0,-1,0,0,0,3]).reshape(3,3)
D_olması_gereken
```

Out[83]:
```
array([[ 1,  0,  0],
       [ 0, -1,  0],
       [ 0,  0,  3]])
```

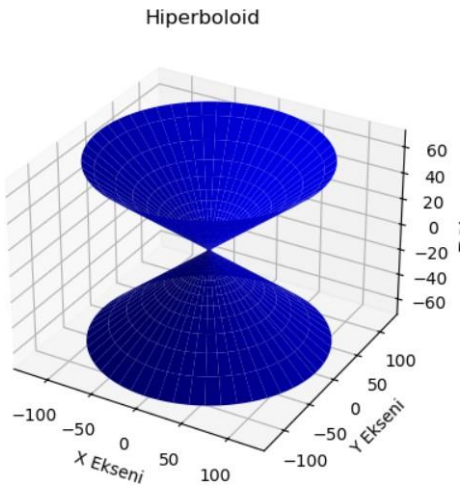
In [66]:
```
print("Q(x', y', z')= 0  kuadriğine")
Z1 = np.array(["x'","y'","z'"]).reshape(3,1)
Z2 = np.array(["x''","y''","z''"]).reshape(3,1)
print(f"Z' matrisi :{Z1} ")
print(f"Z'' matrisi :{Z2}")

print("Z' = PZ'' dönüşümü uygulanırsa ")
```

```
Q(x', y', z')= 0  kuadriğine
Z' matrisi :[["x'"]
 ["y'"]
 ["z'"]]
Z'' matrisi :[["x''"]
 ["y''"]
 ["z''"]]
Z' = PZ'' dönüşümü uygulanırsa
```

In [67]:
```
print( f"(x''**2/ {F1}) - (y''**2/{F1}) + (z''**2/{F1})" , " = 1  elde edilir. " )
```

```
(x''**2/ -2.66666666666667) - (y''**2/-2.66666666666667) + (z''**2/-2.66666666666667)  = 1  elde edilir.
```

In [92]:
```
#Hiperboloidin çizimi:
# Hiperboloid verileri
u = np.linspace(-5, 5, 100)
v = np.linspace(0, 2 * np.pi, 100)
u, v = np.meshgrid(u, v)
d1,d2  = np.meshgrid(u,v)
a = 1.63
b = 1.63
c = 0.94
x_hyperboloid = a * np.cosh(u) * np.cos(v)
y_hyperboloid = b * np.cosh(u) * np.sin(v)
z_hyperboloid = c * np.sinh(u)
# 3D grafik objesi oluşturma
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
# Hiperboloid çizimi
ax.plot_surface(x_hyperboloid, y_hyperboloid, z_hyperboloid,  color='b', label='Hiperboloid')
# Eksen etiketleri
ax.set_xlabel('X Ekseni')
ax.set_ylabel('Y Ekseni')
ax.set_zlabel('Z Ekseni')
plt.title('Hiperboloid')
# Grafik gösterimi
plt.show()
```



Bu kod, kullanıcıdan kuadrik denkleminin katsayılarını alarak verilen denkleme göre çeşitli hesaplamalar ve grafik çizimleri yapmaktadır. İşlem adımları şu şekildedir:

1. Kullanıcıdan kuadrik denkleminin katsayıları alınır.
2. Kuadrik denklemi sembollerle tanımlanır ve yazdırılır.
3. A matrisi oluşturulur ve yazdırılır.
4. Z matrisi oluşturulur ve yazdırılır.
5. B matrisi oluşturulur ve yazdırılır.
6. C matrisi oluşturulur ve yazdırılır.
7. A ve C matrislerinin determinantları hesaplanır ve yazdırılır.
8. Kuadriğin sınıfını belirleme işlemi yapılır ve sonuç yazdırılır.
9. Kuadrik denklemine öteleme uygulama işlemi gerçekleştirilir.
10. Kuadrik denklemin x, y ve z'ye göre kısmi türevleri alınır ve yazdırılır.
11. Denklem sistemini çözmek için denklemler oluşturulur ve çözümleri hesaplanır.
12. F1 ifadesi tanımlanır.
13. Kuadrik denklemin son hali elde edilir ve yazdırılır.
14. A matrisinin özdeğerlerini hesaplanır ve yazdırılır.
15. A matrisinin özdeğer ve özvektörlerini hesaplanır ve yazdırılır.
16. P matrisi oluşturulur ve yazdırılır.
17. P matrisinin transpozu P2 hesaplanır ve yazdırılır.

18. Karakteristik polinomun katsayıları ve polinomu hesaplanır ve yazdırılır.

19. P2 * A * P = D matrisi hesaplanır ve yazdırılır.

20. Kuadrik denklemin son hali D matrisine göre yeniden yazılır ve yazdırılır.

21. Hiperboloid ve doğru çizimleri yapılır ve grafik gösterilir.

Bu adımların her biri, kodun farklı bir parçasını oluşturmakta ve sonuçları ekrana yazdırmaktadır.

5.KAYNAKÇA

[1]Prof.Dr.Salim Yüce, Analitik Geometri, Pegem Akademi, Ankara, 4.Basım (2018)

[2]Uygulamalarla Temel Seviye Python -Enes Açıkgözlüoğlu & Ziya Dirlik (2021)

İnternet Kaynağı

[3]https://bayramadali.wordpress.com/matplotlib/

[4]https://bilginc.com/tr/blog/pythonin-yillar-icindeki-gelisimi-3417/

[5]https://blog.ozkula.com.tr/

[6]https://caylakyazilimci.com/post/python-temelleri

[7]https://docs.python.org/3/library/stdtypes.html

[8]https://docs.sympy.org/latest/index.html

[9]https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html

[10]https://medium.com/bili%C5%9Fim-hareketi/veri-bilimi-i%CC%87%C3%A7in-temel-python-k%C3%BCt%C3%BCphaneleri-1-numpy-750429a0d8e5#:~:text=np.zeros()%20%E2%86%92Bu,lardan%20olu%C5%9Fan%20bir%20matris%20d%C3%B6nd%C3%BCr%C3%BCr.&text=np.ones()%3A%20zeros(),lerden%20olu%C5%9Fan%20bir%20matris%20d%C3%B6nd%C3%BCr%C3%BCr.

[11]https://medium.com/kaveai/python%C4%B1n-sympy-k%C3%BCt%C3%BCphanesi-ve-sembolik-matematik-3cfd1dead4e7

[12]https://medium.com/mmmt-pergamon/pergamon-002-kuadratik-y%C3%BCzeyler-3f15eaae8fb8

[13]https://medium.com/machine-learning-t%C3%BCrkiye/neden-numpy-4130b36892ff

[14]https://numpy.org/doc/stable/reference/index.html

[15]https://pi404.com/pythonda-veri-tipleri/

[16]https://pi404.com/pythonda-veri-tipleri/#List_Veri_Tipi

[17]https://prezi.com/p/ei49mgezojyd/kuadratik-yuzeyler/

[18]https://prezi.com/p/wvp-5wclweuq/kuadratik-yuzeyler-sunum/?frame=6c4c783e92fc4122ce69a9d99348514096714b02

[19]https://python.sitesi.web.tr/python-bool.html

[20]https://python.sitesi.web.tr/python-degiskenler.html

[21]https://python-istihza.yazbel.com/kumeler_ve_dondurulmus_kumeler.html

[22]https://python-istihza.yazbel.com/listeler_ve_demetler.html

[23]https://python-istihza.yazbel.com/python_hakkinda.html#python-nedir

[24]https://sogrekci.com/ders-notu/python-ve-bilimsel-hesaplama/42-grafik-cizimi/

[25]https://www.ntnu.no/wiki/display/imtsoftware/Matrices+and+linear+algebra+iPython

[26]https://tanersayinnn.medium.com/sympy-1-e551f42a019b

[27] https://teknoloji.org/python-nedir

[28]https://tr.wikipedia.org/wiki/Koni

[29]https://tr.wikipedia.org/wiki/Sferoit

[30]https://www.academia.edu/53277411/PYTHON_%C4%B0LE_KODLAMA_II_PYTHON_B%C4%B0L%C4%B0M_PAKET%C4%B0_NUMPY_NUMERIC_SAYISAL_YTHON_SCIPY_SCIENTIFIC_B%C4%B0L%C4%B0MSEL_PYTHON_MATPLOTLIB_GRAF%C4%B0K_K%C3%9CT%C3%9CPHANES%C4%B0_PANDAS_VER%C4%B0_B%C4%B0L%C4%B0M%C4%B0_MOD%C3%9CL%C3C_

[31]https://www.ctrlbizde.com/index.php

[32]https://www.ctrlbizde.com/index.php/egitimler/python-dersleri/item/570-range-fonksiyonu-ve-fon-dongusunde-kullanimi-python3-dersleri-ders14

[33]https://www.halvorsen.blog/documents/programming/python/resources/powerpoints/Linea r%20Algebra%20i n%20Python.pdf

[34]https://www.mobilhanem.com/

[35]https://www.nkfu.com/kure-yuzeyinin-denklemi-nedir-hesaplanmasi-ornek-soru-ve-cozumleri/

[36]https://www.programiz.com/python-programming/online-compiler/

[37]https://www.pythontr.com/makale/python-break-ve-continue-komutlari-624

[38]https://www.sadikturan.com/python-donguler/python-for-dongusu/1388

[39]https://www.sadikturan.com/python-donguler/python-while-dongusu/1389

[40]https://www.w3schools.com/python/numpy/numpy_data_types.asp

[41]https://www.w3schools.com/python/numpy/numpy_intro.asp

[42]https://www.w3schools.com/python/python_intro.asp

[43]Yazbel Python Belgeleri-Fırat Özgül- yazbel.com- yayım 4.0.0

[44] https://www.nedemek.page/kavramlar/hiperboloid

[45]http://smhmatuyg2.ultimatefreehost.in/George%20B.Thomas,%20Jr.%20%26%20Di%C4%9Ferleri%20-%20Calculus%20Cilt%202/mobile/index.html?i=1#p=3

[46] https://avys.omu.edu.tr/storage/app/public/sbektas/71424/Jeodezi%202.pdf