SAYISAL ANALİZ YÖNTEMLERİ FONKSİYON KÜTÜPHANESİ

```
import numpy as np

def hilal():
    print("hilal")

def deneme(katsayi):

    print("dizi: ",katsayi)


def katsayilar(katsayi, derece):
   print("\n\nUYARI ! GIRECEGIN DENKLEMLER ASAGIDAKI GIBI
OLMALIDIR !\n")
   print("\n*** aX^3 + bX^2 + cX + d = 0 *** \n")
   print("\n\nLutfen Fonksiyonun Katsayilarini Giriniz...\n")
   i=0
   for i in range(derece,-1,-1):
        katsayi[i] = float(input("..x^:"))


def denklemSonuc(x,maxDerece,katsayi):

    sonuc = 0
    t = 1
    i=0
    print("max derece:",maxDerece)
    for i in range(maxDerece):
        print("i: ",katsayi[i])
    # x Noktasinda Fonksiyonun Goruntusunu Hesaplar
    for i in range(maxDerece):
        sonuc += katsayi[i] * t
        t = t * x
    print("sonuc: ", sonuc)
    return sonuc

def grafikYontemi(maxDerece,katsayi,xSifir,deltaX,epsilon):
    while(deltaX > epsilon) :
        while ( denklemSonuc(xSifir, maxDerece, katsayi) *
denklemSonuc((xSifir + deltaX), maxDerece, katsayi) >= 0 ):
            xSifir = xSifir + deltaX
        deltaX = deltaX / 2
        print("deltaX: ",deltaX)
    print("\n\nBULUNAN YAKLASIK KOK : ", xSifir)



def BisectionYontemi(derece,katsayi,a,b,epsilon) :

    c = (a + b) / 2
```

```
    flag = 1

    while ( abs(denklemSonuc(c, derece, katsayi)) > epsilon and
flag==1):
        if (denklemSonuc(c, derece, katsayi) * denklemSonuc(a,
derece, katsayi) < 0) :
            b=c
        elif( denklemSonuc(c,derece,katsayi) *
denklemSonuc(b,derece,katsayi) < 0 ):
            a=c
        else :
            print("\n! Bu aralikta kok yok.Baska bir (a,b) araligi
giriniz !\n")
            flag = 0
        c = (a + b) / 2
        print("c=  \n", c)
    print("\n BULUNAN YAKLASIK KOK :  ", c)



def regulaFalse(derece,katsayi,a,b,epsilon) :

    c = (b * denklemSonuc(a, derece, katsayi) - a *
denklemSonuc(b, derece, katsayi)) / (
                denklemSonuc(a, derece, katsayi) - denklemSonuc(b,
derece, katsayi))
    flag = 1
    while (abs(denklemSonuc(c, derece, katsayi)) > epsilon and
flag == 1):
        if (denklemSonuc(c, derece, katsayi) * denklemSonuc(a,
derece, katsayi) < 0):
            b = c
        elif( denklemSonuc(c, derece, katsayi) * denklemSonuc(b,
derece, katsayi) < 0 ):
            a = c
        else :
            print("\n! Bu aralikta kok yok. Baska bir (a,b)
araligi giriniz !\n\n")
            flag = 0
        c = (b * denklemSonuc(a, derece, katsayi) - a *
denklemSonuc(b, derece, katsayi)) / (
                    denklemSonuc(a, derece, katsayi) -
denklemSonuc(b, derece, katsayi))

        print("c= %f \n", c)

    print("\n\n BULUNAN YAKLASIK KOK : %lf \n", c)



def newtonRaphson(derece,katsayi,x,epsilon):
```

```
    guncelX = x
    x = x - denklemSonuc(x, derece, katsayi) / turev(x, derece,
katsayi)

    while(abs( denklemSonuc(x,derece,katsayi) -
denklemSonuc(guncelX,derece,katsayi) ) > epsilon ):
        guncelX = x
        x = x - denklemSonuc(x, derece, katsayi) / turev(x,
derece, katsayi)
    print("\n\n BULUNAN YAKLASIK KOK :   ",x)




def trapez(a, b, n, derece, katsayilar) :

    h = (b - a) / n
    sigma = 0

    for x in range(a+h , b-h-1 , h ):
        sigma += denklemSonuc(x, derece, katsayilar)

    integral = h * ((denklemSonuc(a, derece, katsayilar) +
denklemSonuc(b, derece, katsayilar)) / 2 + sigma)
    return(abs(integral))


def simpson(a, b, n, derece, katsayi) :

    h = (b - a) / n
    sigmaTek = 0
    sigmaCift = 0

    for x in range(a + h, b - h-1, 2*h):
        sigmaTek += denklemSonuc(x, derece, katsayi)

    for x in range(a + 2*h, b - h-1, 2*h):
        sigmaCift += denklemSonuc(x, derece, katsayi)

    integral = (h / 3) * ((denklemSonuc(a, derece, katsayi) +
denklemSonuc(b, derece, katsayi)) + 4 * sigmaTek + 2 * sigmaCift)


    return(abs(integral))


def turev(x,derece,katsayi) :
    turev  = np.zeros(50)
    sonuc = 0
    for i in range(derece) :
        turev[i] = katsayi[i + 1] * (i + 1)
```

```
    for i in range(derece) :
        sonuc += turev[i] * pow(x,i)
    print("türev sonucu:", sonuc)
    return sonuc



def ileriFarkTurev(x, deltaX, derece, katsayi):
    turev = (denklemSonuc((x + deltaX), derece, katsayi) -
denklemSonuc(x, derece, katsayi)) / deltaX
    print("\n* Denklemin", x, "noktasindaki ileri fark turevi =
\n", turev)


def geriFarkTurev(x, deltaX, derece, katsayi):
    turev = ( denklemSonuc( x, derece, katsayi )- denklemSonuc((x-
deltaX), derece, katsayi) ) / deltaX
    print("\n* Denklemin", x, "noktasindaki geri fark turevi =
\n", turev)


def merkeziFarkTurev(x, deltaX, derece, katsayi):
    turev = (denklemSonuc((x + deltaX), derece, katsayi) -
denklemSonuc((x - deltaX), derece, katsayi)) / (2 * deltaX)
    print("\n* Denklemin", x, "noktasindaki merkezi fark turevi =
\n", turev)


def GaussJordanYontemi(denklemler,dsayisi):

    for k in range(dsayisi):
        for i in range(dsayisi):
            a = denklemler[i][k]
            for j in range(dsayisi):
                if (i != k) :
                    denklemler[i][j] = (denklemler[k][k] *
denklemler[i][j]) - (a * denklemler[k][j])


    print( "\nSONUCLAR :\n" )
    for i in range(dsayisi):
        print("\nx",i + 1," = ",denklemler[i][dsayisi] /
denklemler[i][i] ,"\n")
```