

```

#1
for i in range(5):
    for i in range(5):
        print("*", end="")
    print()#satırı atla demek

"""
*****
*****
*****
*****
*****      """
print("-----")
#2
k=1
for i in range(5):
    print("*"*k, end="")
    k += 1
    print()

for i in range(1,6):
    for j in range(i):
        print("*", end="")
    print()
"""
*
**
***
****
*****      """

print("-----")
#3
k=5
for i in range(5):
    print("*"*k, end=" ")
    k -= 1
    print()

for i in range(1,5):
    for j in range(5-i):
        print("*", end="")
    print()
"""
*****
****
***
**
*

```

```

"""
print("-----")
#4

k=0
p=5
for i in range(0,6):
    print(" "*p, end="")
    p -= 1
    print(""*k)
    k += 1

n = 7
for i in range(1, n+1):
    for j in range(2*n - 2*i):
        print(end=" ")
    for j in range(0,i):
        print(end="* ")
    print()

n = 5
for i in range(i, n+1):
    print((2*n - 2*i)* " " + i*"* ")
"""
*
**
***
****
*****

      *
     * *
    * * *
   * * * *
  * * * * *
 * * * * *
* * * * *
"""

```

#5

```

i = 0
for i in range(1,6):
    print(" "*(2*i-2), end="")
    print("* "*(6-i))
    i += 1

n=5
for i in range(1, n+1):
    for j in range((2*i)-2):
        print(end=" ")

```

```

        for j in range(0,n+1-i):
            print(end="* ")
        print()

"""
* * * * *
* * * *
* * *
* *
*
* """

#6
n = 5
for i in range(0,n):
    for j in range(0,i+1):
        print("*", end=" ")
    print()
for i in range(n-1,0,-1):
    for j in range(i):
        print(end="* ")
    print()

n = 5
for i in range(1, 2*n):
    if i <= n:
        print(i*"* ")
    else:
        print((2*n-i)*"* ")

"""
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
* """

#7
n = int(input("lütfen bir sayı giriniz"))
for i in range(1,n):
    print(" "*n, end=" ")
    n -= 1
    print("* "*i)
n = 5
k = n+2
for i in range(n):
    for j in range(k):
        print(end=" ")

```

```
Process finished with exit code 0
    """
```

```
for i in range(1,n+1):
    for j in range(k):
        print(end=" ")
    k -= 1
    for j in range(i):
        print(end="* ")
    print()
for i in range(n-1,0,-1):
    for j in range(k+2):
        print(end=" ")
    k += 1
    for j in range(i):
        print(end="* ")
    print()
    """
```

```

#9
n = 5
char = 65
for i in range(n):
    ch = chr(char)
    char += 1
    for j in range(0,i+1):
        print(ch, end=" ")
    print()

```

```

"""
A
B B
C C C
D D D D
E E E E E
"""

```

```

#10
n = 6
k = n + 2
for i in range(n):
    print(end=" "*k)
    k -= 1
    x = 65
    for j in range(0,i+1):
        ch = chr(x)
        print(ch, end=" ")
        x += 1
    print()
for i in range(n, 1,-1):
    print(end=" "*(k+2))
    k += 1
    x = 65
    for j in range(0,i-1):
        ch = chr(x)
        print(ch, end=" ")
        x += 1
    print()
"""

```

```

    A
  A B
A B C
A B C D
A B C D E
A B C D E F
A B C D E
A B C D
A B C

```

A B

A

""

```

import array
sayılar = array.array("i", [-2, 2, 4, -7])
print(sayılar)
import array as arr
sayılar = arr.array("i", [-2, 2, 4, -7])
print(sayılar)

from array import *
sayılar = array("i", [-2, 2, 4, -7])
print(sayılar)
from array import *
sayılar = array("I", [9, 2, 4, 7]) # "I" ==> kesirli sayı ya da negatif
sayı eklenmez
print(sayılar)

print(sayılar[0])
print(sayılar.buffer_info()) #arrayın nerede tutulduğunu ve içerisinde kaç
tane eleman bulunduğunu verir.
print(sayılar.typecode) # generator tipini sorgular

for index in range(len(sayılar)):
    print(sayılar[index], end=" ")
harfler = array("u", ["f", "b", "s"]) # 9 2 4 7 array('u', 'fbs')
print(harfler)
print(harfler[0])

""" Generators """
from array import *
sayılar = array("i", [-2, 0, 2, 3, 4, 10]) #Generatörler sayesinde ram de
yer kaplamadan iş yapılabilir.
print([sayı for sayı in sayılar])
print((sayı for sayı in sayılar)) # generator obje oluşturuldu ramde yer
kaplamadı
print(list(sayı for sayı in sayılar))
print(set(sayı for sayı in sayılar))
print(tuple(sayı for sayı in sayılar))
print(sum(sayı for sayı in sayılar))
print(max(sayı for sayı in sayılar))

for sayı in (sayı*sayı for sayı in sayılar):
    print(sayı, end=" ") # Generatorler de bu şekilde kullanılabilir
print()
i = 0
while i < len(list(sayı*sayı for sayı in sayılar)):
    print(list(sayı*sayı for sayı in sayılar)[i], end=" ")
    i += 1
print()

sayılar3 = array(sayılar.typecode, (sayı for sayı in sayılar))
print(sayılar3)

```

```

for sayı in sayılar3:
    print(sayı, end=" ")
print()

i = 0
while i < len(sayılar3):
    print(sayılar[i], end=" ")
    i += 1
print()

generator = (sayı*sayı for sayı in sayılar)
i= 0
while i < len(sayılar3):
    print(sayılar[i], end=" ")
    i += 1
print()

""" Kullanıcıdan Array Elemanlarını Alma""" #arr.index()
from array import*
arr = array("i", [])
# printer(arr)
n = int(input("Diziye kaç tane eleman eklemek istiyorsunuz?"))
for i in range(n):
    sayı = int(input("lütfen eklemek istediğiniz sayıyı giriniz"))
    arr.append(sayı)
print(arr)

"""Array içerisindeki elemanı bulma"""
from array import array
arr = array("i", [2, 5, 8, -1, 0, 8, 76])
aranacakSayı = int(input("lütfen aramak istediğiniz sayıyı giriniz."))

i = 0
for sayı in arr:
    if sayı == akanacakSayı:
        print("{} değeri {}. indekste bulunmaktadır.".format(aranacakSayı,
i))
    else:
        print("aradığınız sayı bu array içerisinde bulunmamaktadır.")
        break # ilk bulunan indexte durmasını sağlar
    i += 1

aranacakSayı = int(input("lütfen aramak istediğiniz sayıyı giriniz."))
print("{} değeri {}. indekste bulunmaktadır.".format(aranacakSayı,
arr.index(aranacakSayı)))

""" Array İçerisindeki elemanı silme """
arr = array("i", [2, 5, 8, -1, 0, 75, 8, 75])
print(arr)

```



```

#Karşılaşılan ilk elemanı silme
arr.remove(8)
print(arr)
#Bütün değerleri silme
for i in arr:
    if i == 75:
        arr.remove(75)
print(arr)
arr = array("i", [2, 3, 4, 5, 6, 7, 2, 3, 4, 5, 2])
arr[5] = 100
print(arr)
for index, sayı in enumerate(arr):
    if sayı == 7:
        arr[index] = 100
print(arr)

```

""" Girilen değerin dizi içerisinde kaç defa geçtiğini bulmak """

```

from array import array
arr= array("i", [2, 3, 4, 5, 6, 7, 2, 3, 4, 5, 2, 7])
aranacakSayı = int(input("lütfen kaç tane geçtiğini bulmak istediğiniz sayıyı yazınız"))

```

```

#1.yol
count = 0

```

```

for i in arr:
    if i == akanacakSayı:
        count += 1
print("{} sayısı {} defa geçmektedir.".format(aranacakSayı, count))

```

```

#2.yol
aranacakSayıIndex = array("i", [])
for sayı in enumerate(arr):
    if sayı == akanacakSayı:
        akanacakSayıIndex.append(index)
print(aranacakSayıIndex)

```

```

#3.yol
print("count metodu: {} sayısı {} defa geçmektedir.".format(aranacakSayı, arr.count(aranacakSayı)))

```

ÖRNEK Ekleme, silme ve tersten sıralama işlemlerini metod kullanmadan uygulama.

1-) Boş bir dizi oluşturun. Bu diziye 7 tane eleman kullanıcı tarafından eklenecektir.

Ekleme işlemini append() metodu kullanmadan gerçekleştirin

2-) Bu dizide kullanıcının silmesini istediği değerlerin ailinmesini

sağlayın.

```
# Silme işlemini remove() metodunu kullanmadan gerçekleştirin.  
# 3-) Bu dizinin elemanlarının tersine çevrilmesini sağlayın.  
# Tersine çevirme işlemini reverse() metodunu kullanmadan  
gerçekleştirin.
```

```
# 1
```

```
from array import array  
dizi = array("i", [])  
for i in range(7):  
    sayı = int(input("lütfen eklemek istediğiniz sayıyı yazınız!"))  
    dizi += array("i", [sayı]) # iki array birbiriyle toplandı  
print(dizi)
```

```
delVal = int(input("lütfen silmek istediğiniz sayıyı giriniz"))  
degerSilinmişDizi = array("i", [])  
for i in dizi:  
    if i != delVal:  
        degerSilinmişDizi += array("i", [i])  
dizi = degerSilinmişDizi  
print(dizi)  
tersDizi = array(len(dizi))  
for i in range(len(dizi)-1, -1, -1):  
    tersDizi += array("i", [dizi[i]])  
dizi = tersDizi  
print(dizi)
```

```
""" Foksiyonların en önemli amacı aynı işlemi tekrar tekrar kodlamayı önlemektir"""
```

```
# fonksiyonlar tanımlanırken def anahtar kelimesi kullanılır.
```

```
def seslen():  
    print("sesleniyorum")  
    print("naber")  
seslen()
```

```
# fonksiyon çağırırken call anahtar kelimesi kullanılır.
```

```
""" Dönüş Değeri Olmayan Parametresiz Fonksiyonlar"""
```

```
def selamVer():  
    sonuc = 5 + 6  
    return sonuc # return(dönüş değeri) ortaya bir ürün çıkacak demektir.  
Ortaya sonuc ürünü çıkacak, return kullanılan fonksiyonlar tek başına kullanılarak yazdırılamaz.  
print(selamVer())
```

```
def selamVer():  
    sonuc = 5 + 6  
    print(sonuc) # dönüş değeri olmayan fonksiyonlar print içerisinde kullanılamaz.
```

```
selamVer()
```

```
def isimSoyle():  
    print("Mustafa Kemal ATATÜRK")
```

```
isimSoyle()
```

```
def isimSoyle2():  
    print("Berk Batuhan Devran")
```

```
def bilgiVer():  
    isimSoyle2() # fonksiyon içinde fonksiyon kullanılabilir  
    print("Yıldız Teknik Üniversitesi")  
    print("Matematik Mühendisi")  
    print("ANAMUR")
```

```
bilgiVer()
```

```
def topla(sayı1, sayı2): # parametrelili fonksiyon  
    print(sayı1 + sayı2)
```

```
topla(65,87)
```

```
def isimSoyle3(isim):
```

```

    print(isim)
isimSoyle3("berk")

from math import pi
print(pi)
def piSayısı():
    return pi
print(piSayısı())

def veriTabanınaBaglan():
    print("veri tabanına bağlanıyor")
    return "veri tabanına bağlandı"
print(veriTabanınaBaglan())

bilgi = veriTabanınaBaglan()
print(bilgi)

"""Dönüş Değeri Olan parametrelili(argümanlı) fonksiyonlar"""
print(abs(-9))
sayı = abs(-15)
print(sayı)

def carp(sayı1, sayı2):
    return sayı1 * sayı2
print(carp(4,6))
sonuc = carp(8, 9)
print(sonuc)

def mutlakDegerliCarpım(sayı1, sayı2):
    return abs(sayı1 * sayı2)
print(mutlakDegerliCarpım(-7, 9))

"""Birden Fazla Dönüş Değeri Olan Fonkşşyonlar"""
def ucgenKenarlarınıSoyle():
    return 3,4,5 # defaultta () isersen [] koyabilirsin...
birinciKenar = ucgenKenarlarınıSoyle()[0]
ikinciKenar = ucgenKenarlarınıSoyle()[1]
üçüncüKenar = ucgenKenarlarınıSoyle()[2]
print(ucgenKenarlarınıSoyle())
print(birinciKenar)
print(ikinciKenar)
print(üçüncüKenar)
from math import pi
def daireAlanVeCevreHesapla(r):
    return pi * (r**2), 2*pi*r
print(daireAlanVeCevreHesapla(100))
daireAlan = daireAlanVeCevreHesapla(100)[0]
daireCevre = daireAlanVeCevreHesapla(100)[1]

```

```
print(daireAlan)
print(daireCevre)
```

```
"""FONKSİYON ARGÜMANLARI"""
```

```
# Pass by value (Değer Yollama)
# Pass by reference (Adresi yollama)
```

```
def guncelle(sayi):
    print("Sayı id: ", id(sayi))
    sayi = 9
    print("Sayı id: ", id(sayi))
```

```
sayimiz = 10
guncelle(sayimiz)
print(sayimiz)
```

```
def listeGuncelle(liste):
    print("liste id:", id(liste))
    liste[0] = 100
    print("liste id:", id(liste))
```

```
listemiz = [0, 45, 75, 86, 98]
listeGuncelle(listemiz)
print(listemiz)
```

```
"""Types of Arguments (Argüman tipleri)"""
```

```
#Positional, keyword, default, variable, length
```

```
def cikar(a, b):
    sonuc = a - b
    print(sonuc)
    return sonuc
print(cikar(86, 54))
cikar(86, 54)
```

```
def ogrenciBilgileriSoyle(isim, numara, adres, sube=""):
    print("Öğrencinin ismi:", isim)
    print("Öğrencinin numarası:", numara)
    print("Öğrencinin şubesi:", adres)
    print("Öğrencinin adresi:", sube)
ogrenciBilgileriSoyle("Berk Batuhan Devran", "595", "Mersin/Anamur", "A")
# Eğer bir bilgi daha sonra girilecekse o bilgiyi default da boş olarak tanımlamalıyız
ogrenciBilgileriSoyle("Berk Batuhan Devran", "595", "Mersin/Anamur")
# ogrenciBilgileriSoyle("Berk Batuhan Devran", "Mersin/Anamur") # default da boş olmayan değer
```

için atama yapılmadan fonksiyon çalıştırıldığında program hata vriyor

```
def ogrenciBilgileriSoyle(isim, numara, sube, *adres):
    print("Öğrencinin ismi:", isim)
    print("Öğrencinin numarası:", numara)
    print("Öğrencinin şubesi:", sube)
    print("Öğrencinin adresi:", adres)
ogrenciBilgileriSoyle("Berk", "595", "A", "Mersin/ANAMUR",
"Mersin/TARSUS", "Mersin/AKDENİZ")
# yaptığımız *... değişikliği sonucu adres değeri sonsuz değer alabilir
```

```
def ogrenciBilgileriSoyle(isim, numara, sube, *adres):
    print("Öğrencinin ismi:", isim)
    print("Öğrencinin numarası:", numara)
    print("Öğrencinin şubesi:", sube)
    print("Öğrencinin adresi:", adres)
ogrenciBilgileriSoyle("Berk", "595", "A")
# *adres e değer yazmadığımız zaman program boş bir tuple gönderiyor.
```

```
def ogrenciBilgileriSoyle(isim, numara, sube, *adres):
    print("Öğrencinin ismi:", isim)
    print("Öğrencinin numarası:", numara)
    print("Öğrencinin şubesi:", sube)
    if len(adres) != 0:
        print("Öğrencinin adresi:", adres)
    else:
        print("adres girilmemiştir")
```

```
def ogrenciBilgileriSoyle(isim, numara, sube, adres):
    print("Öğrencinin ismi:", isim)
    print("Öğrencinin numarası:", numara)
    print("Öğrencinin şubesi:", sube)
    if len(adres) != 0:
        print("Öğrencinin adresi:", adres)
    else:
        print("adres girilmemiştir")
ogrenciBilgileriSoyle("Berk", "595", "A", "anamur")
ogrenciBilgileriSoyle(sube="b", isim="batuhan", adres="anamur",
numara="45")
```

```
def ogrenciBilgileriSoyle(isim, numara, adres, sube=""):
    print("Öğrencinin ismi:", isim)
    print("Öğrencinin numarası:", numara)
    print("Öğrencinin şubesi:", adres)
    print("Öğrencinin adresi:", sube)

ogrenciBilgileriSoyle(sube="b", isim="batuhan", adres="anamur",
numara="45")
#Bu kodun doğru çalışabilmesi için default değeri değiştirilen değerler
sube gibi sona alınmalıdır.
```

```
"""variable length"""
def mySum(*sayılar):
    sonuc = 0
    for sayi in sayılar:
        sonuc += sayi
    print(sonuc)
```

```
mySum(4, 5, 6, 7)
```

```
def ogretmenBilgileriSoyle(tcKimlikNo, **data):
    # isim
    # maaş
    # brans
    print(tcKimlikNo, data)
```

```
ogretmenBilgileriSoyle(11111111, isim = "Berk Batuhan", soyad = "DEVİRAN",
maas = 4354, brans = "Matematik")
# çıktı:
# 11111111 {'isim': 'Berk Batuhan', 'soyad': 'DEVİRAN', 'maas': 4354,
'brans': 'Matematik'}
```

```
def ogretmenBilgileriSoyle(tcKimlikNo, **data):
    # isim
    # maaş
    # brans
    print(data["isim"])
    print(data["soyad"])
    print(data["maas"])
    print(data["brans"])
    print(data.keys())
    print(data.values())
    print(data.items())
ogretmenBilgileriSoyle(11111111, isim = "Berk Batuhan", soyad = "DEVİRAN",
maas = 4354, brans = "Matematik")

#çıktı:
#Berk Batuhan
```

```

#DEVİRAN
#4354
#Matematik
#dict_keys(['isim', 'soyad', 'maas', 'brans'])
#dict_values(['Berk Batuhan', 'DEVİRAN', 4354, 'Matematik'])
#dict_items([('isim', 'Berk Batuhan'), ('soyad', 'DEVİRAN'), ('maas',
4354), ('brans', 'Matematik')])

""" değer önüne tek yıldız koyduğün zaman keywordsüz sonsuz eleman
alabilirsin
iki yıldız koyarsan keywordlü sonsuz eleman alabilirsin """

"""def toplamaYadaÇarpmaİşlemi(**sayılarVeİşlem):
    print(sayılarVeİşlem)
    if sayılarVeİşlem['isim'] == "toplama":
        sonuc = 0
        for sayı in sayılarVeİşlem['sayılar']:
            sonuc += sayı
            print(sonuc)
    elif sayılarVeİşlem["isim"] == "çarpma":
        sonuc = 1
        for sayı in sayılarVeİşlem['sayılar']:
            sonuc *= sayı
            print(sonuc)
    else:
        print("lütfeñ toplama veya çarpma işlem bilgileri giriniz ")
toplamaYadaÇarpmaİşlemi(sayılar = [4, 5, 5, 6, 7, 8], isim = "toplama")
toplamaYadaÇarpmaİşlemi(sayılar = [4, 5, 5, 6, 7, 8], isim = "çarpma")
"""

"""Varianle length arguments"""

def bilgileriGoster(*isimler, **yaslar):
    for isim in isimler:
        print(isim)
    for yas in yaslar:
        print(yas)
    for key in yaslar.keys():
        print(key)
    for value in yaslar.values():
        print(value)
    for item in yaslar.items():
        print(item)
bilgileriGoster("batuhan", "berk", "DEVİRAN", fatih = 28, kaan = 35, aysel
= 67)
# positional argümanlar keyword argümanlardan daha sonra yazılamaz.

def deneme(*args, isim):#positional argumanlar sonsuz değer alabilen
ifadelerden sonra gelirse mutlaka key kullanılarak değer atanmalı.

```



```

    print(args)
    print(isim)
deneme("fat", "sadsa", "sfds", isim="dasdass")
#Örnek1 Bir kullanıcının ismini, adını, yaşını ve cinsiyetini(opsiyonel)
olarak gösteren kodu yazınız.
#Kullanıcının cinsiyet bilgisi olmadığında konsolda cinsiyete dair
herhangi bir bilgilendirme olmamalıdır.
def showUserInfo(name, username, age, gender=None):
    if gender!=None:
        print(name, username, age, gender)#ctrl+d ile çoğalt alt+shift ve
yön tuşları ile aşağı taşı
    else:
        print(name, username, age)
showUserInfo("Berk", "Batuhan33", 26, "Heriff")
showUserInfo("batuhan", "berk3", 22)

```

```

def showUserInfo(name, username, age, gender=None):
    if gender!=None:
        print(name, username, age, gender)#ctrl+d ile çoğalt alt+shift ve
yön tuşları ile aşağı taşı
    else:
        print(name, username, age)
showUserInfo("Berk", "Batuhan33", 26, "Heriff")
showUserInfo("batuhan", "berk3", 22)

```

```

def showUserInfo(name, username, age, gender=None):
    if gender!=None:
        print(name, username, age, gender)#ctrl+d ile çoğalt alt+shift ve
yön tuşları ile aşağı taşı
    else:
        print(name, username, age)
showUserInfo("Berk", "Batuhan33", 26, "Heriff")
showUserInfo("batuhan", "berk3", 22, gender="herif")

```

#Örnek2 Sınırsız sayıda sayının karekökünü hesaplayan fonksiyon.

```

from math import sqrt
def calcSqrt(*args):
    Liste = []
    for i in args:
        Liste.append(round(sqrt(i),4))

```

```

print(calcSqrt(4, 6, 67, 89))

```

Örnek3 Bir websitenin veri tabanında kullanıcılar kayıtlıdır. Ve bu kullanıcıların id değerleri vardır.

Yazılımcımız bu id ler arasında en küçük id yi 1 yapmak istemektedir. Ve 1 yapılan en küçük id nin azaldığı değer kadar

diğer id leri azaltmak istemektedir.

Id lerin azaltılmış haline ulaştıktan sonra kullanıcıların her birini

isim ve is değeri ile birlikte konsolda görmek istemektedir.
İlgili fonksiyonu yazınız.
Not:Fonksiyonda parametre olarak sadece keyword arguments alınacaktır.
Output: [('Berk', 1), ('Batuhan', 2)...]

```
def reduceIDandShowINFO(**kwargs):
    minVal = min(kwargs['ids'])
    reducedIDs = []
    for id in kwargs['ids']:
        reducedIDs.append(id - minVal + 1)
    infos = []
    for i in range(len(reducedIDs)):
        infos.append((kwargs['names'][i], reducedIDs[i]))
    return infos

print(reduceIDandShowINFO(ids=[100, 200, 300], names=["berk", "batuhan",
"devran"]))
```

#Örnek4

#Bir inşaat firmasında bir projenin başlangıç ve çalışan maliyetleri vardır. Bunun yanı sıra bilinmeyen maliyetler ve blinen extra # maliyetler de firma tarafından tutulmaktadır.
#Firmanın başlangıç, çalışan, bilinmeyen ve bilinen extra maliyetlerinin toplamını hesaplayan fonksiyonu yazınız.

```
def totalCost(initCost, employeCost, *unknownCosts, **extraKnownCosts):
    totalCostValue = initCost + employeCost
    for unknownCost in unknownCosts:
        totalCostValue += unknownCost
    for extraKnownCost in extraKnownCosts.values():
        totalCostValue += extraKnownCost
    print(totalCostValue)
```

```
totalCost(5, 10, 15, 45, 55, walls=100, doors=99)
```

"""LOCAL VARIABLE vs GLOBAL VARIABLE"""

```
a = 10 #global variable
```

```
def fonk1():
    a = 15 #local variable
    print(a)
fonk1()
def fonk2():
    global a
    a = 20
    print("fonk2 a değeri:", a, "adres:",id(a))
print("global a:", a, "Adres:", id(a))
```

```

fonk2()
print("global a:", a, "Adres:", id(a))

print("-----")
if True:
    y = 10
print(y)
while True:
    z = 20
    break

print(z)

for i in range(1):
    t = 30
print(t)
print("-----")
print(globals())#####

x = 100
print(x, id(x))
def fonk3():
    global x
    y = x
    y = 500
globals()['x']
fonk3()
print(x, id(x))
def fonk3():
    y = globals()['x']
    y = 500
fonk3()
print(x, id(x))

""" FONKSİYONLARDA VE DÖNGÜLERDE SONSUZLUK """
"""
import sys
sys.setrecursionlimit(200)# recursion özyineleme(kendi kendini çağırma)
demektir
print(sys.getrecursionlimit())

while True:
    print("while döngüsü")
def sonsuz():
    print("sonsuz gidiyoruz")
    sonsuz()
sonsuz()"""

```

#Örnek- Bir listeniİN içindeki tek ve çift sayıların kaç tane olduğunu

hesaplayan fonksiyonu yazınız.

```
def countOddAndEvenNumber(l):
    countEven = 0
    countOdd = 0
    for i in l:
        if i % 2 == 0:
            countEven += 1
            continue
        countOdd += 1
    return countEven, countOdd
oddEvenList = [2, 4, 6, 78, 978, 56, 4674, 74, -66]
oddEvenCount = countOddAndEvenNumber(oddEvenList)
print(oddEvenCount)
print("Count of odd number in list:", oddEvenCount[0], "Count of even
number in list", oddEvenCount[1])
oddCount, evenCount = oddEvenCount
print(oddCount)
print(evenCount)
```

ÖRNEK: Fibonacci serisini istenilen eleman sayısına göre bastıran programı yazınız.

```
"""n = int(input("Fibonacci serisini kaçınıcı elemana kadar görüntülemek
istiyorsunuz?"))
first = 0
second = 1
third = first + second
print(first, second, third, end=" ")
for i in range(0, x-3):
    first = second
    second = third
    third = second + first
    print(third, end=" ")
"""
def fibo(n):
    if type(n) == int:
        if n <= 0:
            print("lütfeñ pozitif bir tamsayı giriniz")
        else:
            first = 0
            if n == 1:
                print(first)
                return # return döñgülerdeki break işlemini sağlar
            second = 1
            print(first, second, end=" ")
            for i in range(n - 2):
                third = first + second
                first = second
                second = third
                print(third, end=" ")
```

```

        print()
    else:
        print("l\u00fctfen pozitif bir tamsayı giriniz")
        print()

fibonacci(9)
fibonacci(45)

```

#\u00d6rnek: faktoriyel hesaplama

#iterative(tekrarlanan)

```

def faktoriyel(n):
    if type(n) == int and n>=0:
        sonuc = 1
        for i in range(2, n+1):
            sonuc *= i
        print(sonuc)
    else:
        return "l\u00fctfen do\u011fal sayı giriniz"
faktoriyel(9)

```

#recursive(öz yinelenen)

```

def recfaktoriyel(n):
    if type(n) == int and n >= 0:
        if n==0:
            return 1
        return n * recfaktoriyel(n-1)
    else:
        return "l\u00fctfen do\u011fal sayı giriniz"
print(recfaktoriyel(9))

```

"""Lambda anonymous function (ismi olmayan fonksiyonlar)"""

```

def kareAl(sayi):
    return sayi * sayi
print(type(kareAl))
print(kareAl(5))

```

```

lambdakareAl = lambda sayi : sayi * sayi
print(lambdakareAl)
print(lambdakareAl(8))

```

```

lambdatopla = lambda sayi1, sayi2 : sayi1 + sayi2
print(lambdatopla)
print(lambdatopla(7, 87))

```

```

kareAlFonksiyonu = kareAl
print(kareAlFonksiyonu(9))
print(type(kareAlFonksiyonu))

```

#Örnek1 Bir sayının küpünü alan bir fonksiyon yazın.

```
kupAl = lambda sayi: sayi*sayi*sayi
```

```
print(kupAl(9))
```

#Örnek2 Bir stringi tersten yazan anonymous fonksiyon yazın.

```
tersYaz = lambda str: str[::-1]
```

```
print(tersYaz("BERK BATUHAN DEVRAN"))
```

```
"""ÖZEL FONKSİYONLAR(map, filter, reduce)"""
```

```
#map() return değeri ne ise o elemanlar maplenmiş listeye eklenir. Normal liste ile aynı boyuttadır.
```

```
#map = eşlemek planlamak
```

```
sayilar = [8, 9, 23, 24, 13, 67, -8, 76, -78, -66]
```

```
def ikiIlecarp(liste):
```

```
    sonuc= []
```

```
    for sayi in liste:
```

```
        sonuc.append(sayi*2)
```

```
    return sonuc
```

```
print(ikiIlecarp(sayilar))
```

```
print("-----")
```

```
def ikiIleCarp(sayi):
```

```
    return 2*sayi
```

```
sayilar2 = list(map(ikiIleCarp,sayilar))
```

```
print(sayilar2)
```

```
sayilarınİkiKati = list(map(lambda sayi: sayi*2, sayilar))
```

```
print(sayilarınİkiKati)
```

```
#filter() filtrelenen verşler döner. (return bool olduğunda True ise
```

```
filrelenmiş listeye değeri ekler)
```

```
#filter=filtrelemek
```

```
sayilar = [2, 3, 4, 5, -6, -7, -9]
```

```
def pozitifMi(sayi):
```

```
    return sayi>0 #return burada if görevi yapar true dönerselisteğe ekleme yapılır.
```

```
pozitifSayilar = filter(pozitifMi, sayilar)
```

```
print(pozitifSayilar)
```

```
print(list(pozitifSayilar))
```

```
pozitifSayilar = list(filter(pozitifMi, sayilar))
```

```
negatifSayilar = [sayi for sayi in sayilar if sayi not in pozitifSayilar]
```

```
negatifSayilar2 = list(filter(lambda sayi: sayi < 0, sayilar))
```

```
print(negatifSayilar)
```

```
print(negatifSayilar2)
```

```
negatifSayilar.sort()#küçükten büyüğe sıralandı
```

```
print(negatifSayilar)
```

```
negatifSayilar.sort(reverse=True)#büyükten küçüğe sıralandı
```

```
print(negatifSayilar)
```

```
print("-----")
```

```

sayilar = [8, 9, 23, 24, 13, 67, -8, 76, -78, -66]
def cifSayi(sayi):
    return sayi % 2 == 0
çiftSayilar = list(filter(cifSayi, sayilar))
print(çiftSayilar)

```

```

tekSayilar = list(filter(lambda sayi: sayi % 2 != 0, sayilar))
print(tekSayilar)

```

reduce() bir listedeki elemanları dönüş değeriyle beraber gezmeyi sağlar
 # return edilen değer ve sıradaki eleman parametre olarak kullanılır.

```

# reduce = indirgemek
sayilar = [3, 2, 4, 6, -7]
sayilar2 = [3.2, 5.4, 7.8, 9.8]

```

```

def sayilarTopla(liste):
    sonuc = 0
    for sayi in liste:
        sonuc += sayi
    return sonuc
print(sayilarTopla(sayilar))
print(sayilarTopla(sayilar2))

```

```

from functools import reduce
toplama = reduce(lambda a, b : a + b, sayilar)
print(toplama)
def toplama(sayi1, sayi2):
    return sayi1 + sayi2
toplama3 = reduce(toplama, sayilar)
print(toplama3)

```

```

liste = [1, 2, 3, 4]
sonuc = reduce(lambda a,b : a*b, liste)
print(sonuc)

```

```

n = int(input("lütfen faktoriyelini öğrenmek istediğiniz sayıyı giriniz"))
liste = list(range(1, n+1))
sonuc = reduce(lambda a,b : a*b, liste)
print(sonuc)

```

#ÖRNEK

Bir liste içerisindeki en küçük ve en büyük sayıyı reduce fonksiyonunu kullanarak bulunuz. Bir de lambda kullanarak bulunuz.

```

def enBüyükSayi(a,b):
    if a>b:
        return a

```

```

        else:
            return b

liste = [4, 5, 6, -5, -7, 0, 99]
print(reduce(enBüyükSayi, liste))

print(reduce(lambda a, b: a if a>b else b, liste))

def enKüçükSayi(a,b):
    if a<b:
        return a
    else:
        return b
print(reduce(enKüçükSayi, liste))
print(reduce(lambda a, b: a if a<b else b, liste))

#ÖRNEK
#Notların katsayısını ve notları ayrı ayrı tutan 2 liste bulunmaktadır.
#Bu 2 listeyi kullanarak her bir not için dönem sonu notunu hesaplayın.
# NOT: Dönem sonu notlarının toplamı 50 üzerindeyse öğrenci geçmiş
altındaysa kalmış kabul edilmektedir.
katsayilar = [0.15, 0.25, 0.6]
notlar = [50, 70, 90]
def katsayıÇarp(katsayi,_not):
    return katsayi*_not
dönemSonuNotlar = list(map(katsayıÇarp, katsayilar, notlar))
dönemOrtalaması = reduce(lambda a,b: a+b, dönemSonuNotlar)
print(dönemOrtalaması)
if dönemOrtalaması>=50:
    print("geçtiniz")
else:
    print("kaldınız")

"""ZIP FONKSİYONU"""
numaralar = [123, 234, 456, 6577]
isimler = ["berk", "batuhan", "devran", "bbd"]
bilgiler = zip(numaralar, isimler)
print(bilgiler)
print(list(bilgiler))

yaslar = (23, 32, 12, 45)
bilgiler2 = zip(numaralar, isimler, yaslar)
#print(list(bilgiler2))# fazla gelen eleman eşlenmez

for bilgi in bilgiler2:
    print(bilgi)

bilgiler3 = zip(numaralar, isimler, yaslar)
for numara, isim, yas in bilgiler3:

```



```

    print("öğrenci no {} isim {} yaş {}".format(numara, isim, yas))

#ziplenerek oluşturulan bir liste sadece bir kez kullanılabilir

bilgilerSet = set(zip(numaralar, isimler, yaslar))
print(bilgilerSet)

"""bilgilerDict = dict(zip(numaralar, isimler, yaslar))
print(bilgilerDict)""" # sözlükler içli olarak dict edilemez

"""ENUMERATE FONKSİYONU"""

sayilar = (1, 2, 3, 4, 5)
print(sayilar)
print(list(enumerate(sayilar)))

enumerateSayılar = enumerate(sayilar)
print(enumerateSayılar)
print(list(enumerateSayılar))

enumerateSayılar = enumerate(sayilar)
print(tuple(enumerateSayılar))

def myEnumrate(list):
    count = 0
    newList = []
    for i in list:
        newList.append((count, i))
        count += 1
    return newList
print("myEnumerated list :", myEnumrate(sayilar))
print("myEnumerated list :", tuple(myEnumrate(sayilar)))

rehber = {'berk':5, 'batuhan':7}
print(list(enumerate(rehber)))# anahtar kelimeler tutulur değerler kabulur
print(list(enumerate(rehber.values()))) # bu şekilde de değerler üzerinden
bir numaralandırma işlemi yapılabilir

"""all() ve any() fonksiyonu"""
# all() bütün değerler için True ise True any() en az 1 değer için True
ise True döndürür.
print(all([True, True, False]))
print(all((True, True, False)))
print(all((True, True, True)))

print(any([True, True, False]))

```

```
print(any((True, True, False)))
print(any((True, True, True)))
```

```
def myAny(liste):
    for i in liste:
        if i == True:
            return True
    return False
```

```
def myAll(liste):
    for i in liste:
        if i == False:
            return False
    return True
print(myAll([True, True, False]))
print(myAll((True, True, False)))
print(myAll((True, True, True)))
```

```
print(myAny([True, True, False]))
print(myAny((True, True, False)))
print(myAny((True, True, True)))
```

```
"""Bir Fonksiyonun Fonksiyon döndürmesi"""
```

```
def bilgiVer(func):
    print("bilgi verildi")
    return func
def kullanıcıyıGoruntule():
    return "Berk Batuhan Devran"
```

```
print(bilgiVer(kullanıcıyıGoruntule))# bu şekilde kullanıcıyı görğntğle
fonksiyonunun sadece adres bilgisi alınabilir
print(bilgiVer(kullanıcıyıGoruntule()))
```

```
"""KKONU - DECORATORS (İstedğimiz şekilde fonksiyonları süslememizi
sağlar)"""
```

```
def funcInfo(func):
    func()
def soruSor():
    print("fonksiyonun çalışması başladı")
    print("Soru sordum")
    print("fonksiyonun çalışması bitti")
def cevapVer():
    print("fonksiyonun çalışması başladı")
    print("cevap verdim")
    print("fonksiyonun çalışması bitti")
```

```
soruSor()
cevapVer()
```

```

def funcInfo(func):
    print("fonksiyonun çalışması başladı")
    func()
    print("fonksiyonun çalışması bitti")
@funcInfo
def soruSor():
    print("Soru sordum")

def cevapVer():
    print("cevap verdim")

"""soruSor()
cevapVer()
funcInfo(soruSor)
funcInfo(cevapVer)"""
print(soruSor)#None ifadesi geldi--- funncInfo fonksiyonunun return
değerini verir. Fonksiyonların return değeri varsayılanda none dir.
print(type(soruSor))# type = str
print(cevapVer)#<function cevapVer at 0x000001F1BA5B5A60> ifadesi geldi

```

```

print("-----")
print("-----")
print("-----")

```

```

def funcInfo(func):
    print("fonksiyonun çalışması başladı")
    func()
    print("fonksiyonun çalışması bitti")
@funcInfo
def soruSor():
    print("Soru sordum")
@funcInfo
def cevapVer():
    print("cevap verdim")
# bu haliyle program çalıştırıldığı zaman biz çağırmasak da fonksiyonlar
çalışıyor

```

```

print("-----")

```

```

def funcInfo(func):
    def bilgiVer(): #inner fonksiyon
        print("fonksiyonun çalışması başladı")
        func()
        print("fonksiyonun çalışması bitti")
        return "fwfs"

```

```

        return bilgiVer
@funcInfo
def soruSor():
    print("Soru sordum")
    return "akfsdşjşvs"
@funcInfo
def cevapVer():
    print("cevap verdim")
# artık fonksiyonlar kendi kendine çağırılmıyor. :)

funcInfo(soruSor)()
funcInfo(cevapVer)()

print("-----")

print(soruSor) # <function funcInfo.<locals>.bilgiVer at
0x000001D68B705E50>
print(soruSor())
"""
fonksiyonun çalışması başladı
Soru sordum
fonksiyonun çalışması bitti
fwfs"""
soruSor()
"""

fonksiyonun çalışması başladı
Soru sordum
fonksiyonun çalışması bitti
"""

cevapVer()
"""

fonksiyonun çalışması başladı
cevap verdim
fonksiyonun çalışması bitti"""

#Not:funcInfo(soruSor)=soruSor()

print("-----")

```

#ÖRNEK

```

def funcInfo(func):
    def inner():
        print("fonksiyonun çalışması başladı")
        func()
        print("fonksiyonun çalışması bitti")
    return inner #iner fonksiyonun çağırılmamasına() dikkat ediniz
def soruSor(isim, söz):
    print(isim, ":", söz)

```

```

def cevapVer(isim, söz):
    print(isim,":", söz)
soruSor("Batuhan", "Nasılsın?")
cevapVer("Kaan","İyiym, sen nasılsın?")

print("-----")

def funcInfo(func):
    def inner(isim, söz):
        print("Konuşma çalışması başladı")
        func(isim, söz)
        print("Konuşma çalışması bitti")
    return inner    #iner fonksiyonun çağırılmamasına() dikkat ediniz

def soruSor(isim, söz):
    print(isim,":", söz)

def cevapVer(isim, söz):
    print(isim,":", söz)

funcInfo(soruSor)# Bu ifade inner fonksiyonu tmsil eder ve burada inner fonksiyon çağırılmadığı() için program bir bilgi döndürmez
funcInfo(soruSor)("Batuhan", "Nasılsın?")

print("-----")
def funcInfo(func):
    def inner(*args, **kwargs):
        print("Konuşma çalışması başladı")
        func(*args, **kwargs)
        print("Konuşma çalışması bitti")
    return inner    #iner fonksiyonun çağırılmamasına() dikkat ediniz

def soruSor(isim, söz,yas):
    print(isim,":", söz, "yaşı", yas)

def cevapVer(isim, söz):
    print(isim,":", söz)

funcInfo(soruSor)# Bu ifade inner fonksiyonu tmsil eder ve burada inner fonksiyon çağırılmadığı() için program bir bilgi döndürmez
funcInfo(soruSor)("Batuhan", "Nasılsın?",yas=22)

# Örnek: Aşağıdaki çıktıyı veren programı dekoratör kullanarak yazınız.
"""
*****
%%%%%%%%%%%%%%
BERK BATUHAN DEVRAN

```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
*****
"""
```

```
def yüzdelikBas(func):
    def inner(*args, **kwargs):
        print("%"*30)
        func(*args, **kwargs)
        print("%"*30)
    return inner
```

```
def yıldızBas(func):
    def inner(*args, **kwargs):
        print("%"*30)
        func(*args, **kwargs)
        print("%"*30)
    return inner
```

```
def yazdir(isim):
    print(isim)
```

```
yıldızBas(yüzdelikBas(yazdir))("BERK BATUHAN DEVRAN")
```

```
print("-----")
```

```
def yüzdelikBas(func):
    def inner(*args, **kwargs):
        print("%"*30)
        func(*args, **kwargs)
        print("%"*30)
    return inner
```

```
def yıldızBas(func):
    def inner(*args, **kwargs):
        print("%"*30)
        func(*args, **kwargs)
        print("%"*30)
    return inner
```

```
@yüzdelikBas
@yıldızBas
def yazdir(isim):
    print(isim)
```

```
yazdir("BERK BATUHAN DEVRAN")
```

Örnek: Liste içerisindeki sayıların karesini ve karekökünü bulan 2 ayrı
fonksiyonun çalışma sürelerini
hesaplayan programı yazınız.

```
#import math
from math import sqrt
import time
print(time.time(), "saniye")# 1 ocak 1970 ten beri geçen saniye sayısı
print(time.time()/60, "dakika")
```

```
def calismaSuresiHesapla(func):
    def inner(*args, **kwargs):
        x = time.time()
        func(*args, **kwargs)
        y = time.time()
        print(y-x,"saniyede çalışmıştır")
    return inner
```

```
@calismaSuresiHesapla
def karekökAl(sayilar):
    sayilar = [sqrt(sayi) for sayi in sayilar]
    print(sayilar)
sayilar = list(range(0,1000))
karekökAl(sayilar)
print("-----")
@calismaSuresiHesapla
def kareAl(sayilar):
    sayilar = [sayi**2 for sayi in sayilar]
    print(sayilar)
sayilar = list(range(0,1000))
kareAl(sayilar)
```

#Örnek: İki sayıyı bölme işleminde otomatik olarak büyük olanı küçük olana bölen programı yazınız.

```
def buyuguBol(sayi1, sayi2):
    if sayi1 < sayi2:
        sayi1, sayi2 = sayi2, sayi1
    print(sayi1/ sayi2)
```

```
buyuguBol(12,2)
buyuguBol(2,12)
```

```
print("-----")
```

```
def form_bol(func):
    def inner(*args, **kwargs):
        sayi1 = args[0]
        sayi2 = args[1]
        if sayi1 < sayi2:
            sayi1, sayi2 = sayi2, sayi1
```

```

        func(sayi1, sayi2)
    return inner
@form_bol
def bol(sayi1, sayi2):
    print(sayi1/sayi2)
bol(3,6)
bol(6,3)

print("-----")
#Bir kelimenin palindrome olup olmadığını gösteren programı yazınız
"KABAK=KABAK"
def isPalindrome(s):
    return s == s[::-1]
kelime = "kabak"
sonuc = isPalindrome(kelime)
if sonuc: #sonuc == True: yazmanın kısaltması
    print(kelime, "palindromdur")
else:
    print(kelime, "palindrom değildir")
print(isPalindrome("kabak"))
print("-----")
#ÖRNEK: Bir kullanıcının isim, doğum tarihi, email ve şifre bilgisi
bulunmaktadır.
#     Bu isim, doğum tarihi, email ve şifre bilgisine "isim", "doğum
tarihi" ve "sifre" anahtar kelimeleriyle
#     ulaşılmak istenmektedir.
#     Bu kullanıcıların yaş ortalamasının üzerinde kalan kullanıcıyı
filtreleyen programı yazınız.

user1 = {"isim": "fatih", "doğumTarihi": 1990, "email": "ssgdsgs@gmail.com",
"sifre": 1234}
user2 = {"isim": "aysel", "doğumTarihi": 1992, "email": "fgrhr@gmail.com",
"sifre": 1235}
user3 = {"isim": "kaan", "doğumTarihi": 1994, "email": "kjhkgs@gmail.com",
"sifre": 1236}

users = [user1, user2, user3]
print(users)

def ortBul(users):
    toplam = 0
    for user in users:
        toplam += 2021 - user['doğumTarihi']
    return toplam/len(users)
usersYasOrt = ortBul(users)
def yasFiltrele(user):
    return 2021-user['doğumTarihi'] > usersYasOrt

ortUstuUsers = filter(yasFiltrele, users)
print(list(ortUstuUsers))

```



```
print("-----")
#ÖRNEK:Kullanıcıların bilgilerinin içerisinde "doğumTarihi" bilgisinşn
yanı sıra bir de "guncelYas"
# bilgisi tutulmak istenmektedir. Bunu sağlayan programı yazınız(map
fonksiyonu kullanınız).
#Yeni kullanıcılar listesi ile ilk önceki kullanıcılar listesi ayrı ayrı
tutulmaktadır. Var olan liste değişmemelidir.
#İpucu: python deep copy(araştırın).
```

```
user1 = {"isim":"fatih", "doğumTarihi":1990, "email":"ssgdsgs@gmail.com",
"sifre":1234}
user2 = {"isim":"aysel", "doğumTarihi":1992, "email":"fgrhr@gmail.com",
"sifre":1235}
user3 = {"isim":"kaan", "doğumTarihi":1994, "email":"kjhkgs@gmail.com",
"sifre":1236}
```

```
users = [user1, user2, user3]
```

```
"""# SÖZLÜĞE EKLEME YAPMAK
user1['guncelYas'] = 2021 - user1['doğumTarihi']
print(users)"""
```

```
#attribute=özellik
def attrEkle(user):
    user['guncelYas'] = 2021 - user['doğumTarihi']
    return user
"""
```

```
# shallow copy (yeni liste kopyalanan listenin adresi üzerinden
oluşturuluyor, yani newUsers ile users aynı adresi paylaşır.)
newUsers = list(map(attrEkle, users))
print(newUsers)
print(users)#istemese de users listesi de güncellendi
```

```
# deep copy (yeni liste kopyalanan listeden farklı bir adreste depoanır.)
# deep copy copy modülü kullanılarak gerçekleştirilir.
"""
```

```
import copy
newUsers = copy.deepcopy(users)
newUsers = list(map(attrEkle, users))
print(newUsers)
print(users)
print("-----")
#Örnek: Bir listede 0-50 arasındaki sayılar tutulmaktadır, bu sayıların
toplamını reduce fonksiyonunu kullanarak hesaplayınız
```

```
from functools import reduce
```

```

sayilar = [range(0,51)]
print(sayilar)
sayilar = [*range(0,51)]
print(sayilar)
def toplam(s1, s2):
    return s1 + s2
sayılarınToplamı = reduce(toplam, sayilar)
print(sayılarınToplamı)

print("-----")
#Örnek: bir sistem girişinde log kaydı tutan bir program yazılacaktır.
#İpucu: şu anki tarih ve saat bilgilerine erişmek için datetime modülünün
datetime sınıfı import edilmelidir.
"""OUTPUT - ÇIKTI
admin başarılı bir şekilde sisteme girdi. Tarih: 2021-09-11 03.11.33
asadmin adında bir giriş denendi.
Tarih: 2021-09-11 03.11.38 admin hatalı şifre girdi. Tarih:2021-09-11
03.11.46 """
from datetime import datetime
print(datetime.now())

def log(func):
    def inner(*args):
        message = func(*args)
        with open("Dosyalar/logs.txt", "a", encoding="utf-8") as dosya:
            dosya.write(message+ "Tarih: "+ str(datetime.now())+"\n")
        return inner

_username = "admin"
_password = 1234

@log
def sistemGiris(username, password):
    if username != _username:
        print("Böyle bir kullanıcı bulunmamaktadır")
        return username + "adında bir giriş denendi"
    elif password != _password:
        print("Hatalı şifre girdiniz")
        return username + "hatalı şifre girdi"
    else:
        print("Sisteme hoşgeldiniz")
        return username + "sisteme başarılı bir şekilde girdi"

sistemGiris("madmin",4321)
sistemGiris("admin", 1234)

```

```
"""import math
print(math.sqrt(16))
```

```
# as = alias
import math as m
print(m.sqrt(16))
```

```
from math import * #artık sadece fonksiyon ismi tazılarak kullanılabilir
print(sqrt(16))
```

```
from math import sqrt
print(sqrt(16))
```

```
from math import sqrt, factorial, floor, ceil
print(factorial(5))
print(floor(8,9))
print(ceil(9,1))"""
```

```
from math import sqrt
def sqrt(*args):
    return "Bu fonksiyon override edilmiştir"
print(sqrt(64))
```

```
import math
help(math)
print(dir(math))
```

```
import hesap
toplam = hesap.topla(5,6)
print(toplam)
daireCevre = hesap.daireCevreHesapla(8)
print(daireCevre)
daireAlan = hesap.daireCevreHesapla(7)
print(daireAlan)
print("-----")
#Konu: Special Variable olan __name__ değişkeninin işlevi
#     __name__ ana modülde kullanıldığında __main__ değerine sahiptir
#     Fakat başka modül import edildiğinde ve o modülde __name__
kullanıldıysa o zaman
#     kullanılan modülün ismine sahiptir.
```

```
#HATIRLATMA: Bir modül import edildiğinde edinilen modüldeki kodlar
kullanıldığı modülü içerisine yerleştirilir.
```

```
import hesap as h
print(h.__name__)
print(__name__)
print("-----")
#ÖRNEK: Sayı tahmini oyunu/Bilgisayar 1-50 arasında 1 sayı tutacak ve
```

```
kullanıcı bu sayıyı
# tahmin etmeye çalışacak
# Kullanıcının tahmin etmesi için sadece 5 hakkı vardır
# Program oyun başlamadan önce 5 saniye beklemede kalmalıdır.(sleep)
#1-) time modülünün içerisinde bulunan sleep fonksiyonundan
    faydalanılacaktır.
# sleep metodunun ne işe yaradığını ve nasıl çalıştığını kontrol edin.
#2-) random modülünün randint fonksiyonundan faydalanılacaktır.
# randint fonksiyonun ne işe yaradığını ve nasıl çalıştığını kontrol edin.
```

```
from time import sleep
from random import randint
print("Sayı tahmin oyunu başlıyor, lütfen bekleyiniz")
sleep(2)
sayi = randint(0,50)
print(sayi)
tahminHakkı = 5
while True:
    if tahminHakkı > 0:
        tahmin = int(input("lütfen tahmininizi giriniz"))
        tahminHakkı -=1

        if tahmin < sayi:
            print("lütfen sayıyı arttırınız")
        elif tahmin > 0:
            if tahminHakkı > 0:
                print("lütfen sayıyı azaltınız")
            else:
                print("kazandınız")
        else:
            print("tahmin hakkınız kalmamıştır")
            break
```