```
""  Foksiyonların en önemli amacı aynı işlemi tekrar tekrar kodlamayı önlemektir"""
# fonksiynlar tanımlanırken def anahtar kelimesi kullanılır.

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

# fonksiyon çağırırken call anahtar kelimesş kullanılır.

""" Dönüş Değeri Olmayan Parametresiz Fonksşyonlar"""
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): # parametreli 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 parametreli(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 Fonkşyonlar"""
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("Sayi id: ", id(sayi))
    sayi = 9
    print("Sayi 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, veriable, 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 göderiyor.



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")
```

```python
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
    # branş
    print(tcKimlikNo, data)

ogretmenBilgileriSoyle(111111111, isim = "Berk Batuhan", soyad = "DEVRAN",
maas = 4354, brans = "Matematik")
# çıktı:
# 111111111 {'isim': 'Berk Batuhan', 'soyad': 'DEVRAN', 'maas': 4354,
'brans': 'Matematik'}

def ogretmenBilgileriSoyle(tcKimlikNo, **data):
    # isim
    # maaş
    # branş
    print(data["isim"])
    print(data["soyad"])
    print(data["maas"])
    print(data["brans"])
    print(data.keys())
    print(data.values())
    print(data.items())
ogretmenBilgileriSoyle(111111111, isim = "Berk Batuhan", soyad = "DEVRAN",
maas = 4354, brans = "Matematik")

#cıktı:
#Berk Batuhan
```

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


""" değer önüne tek yıldız koyduğun 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['sayilar']:
            sonuc += sayı
            print(sonuc)
    elif sayılarVeİşlem["isim"] == "çarpma":
            sonuc = 1
        for sayi in sayılarVeİşlem['sayilar']:
            sonuc *= sayi
            print(sonuc)
    else:
        print("lütfen toplama veya çarpma işlem bilgileri giriniz ")
toplamaYadaÇarpmaİşlemi(sayilar = [4, 5, 5, 6, 7, 8], isim = "toplama")
toplamaYadaÇarpmaİşlemi(sayilar = [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", "DEVRAN", fatih = 28, kaan = 35, aysel
= 67)
# positional argümanlar keyward 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", "sfsd", 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 Sınırsı sayıda sayınının karekökükü 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 keyward 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 exra
# 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 VARİABLE vs GLOBAL VARİABLE"""

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("sonsuza gidiyoruz")
    sonsuz()
sonsuz()"""

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

```
hesaplayan fonksşyonu 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 kaçıncı 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("lütfen pozitif bir tamsayı giriniz")
        else:
            first = 0
            if n == 1:
                print(first)
                return  # return döngülerdeki 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("lütfen pozitif bir tamsayı giriniz")
        print()

fibo(9)
fibo(45)

#Örnek: faktoriyel hesapplama
#ıterative(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 "lütfen doğal 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 "lütfen doğal 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)

sayilarınIkiKati = list(map(lambda sayi: sayi*2, sayilar))
print(sayilarınIkiKati)

#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 dönerselisteye
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("-----------------------------------------------")
```

```python
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 = indşrgemek
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
toplam = reduce(lambda a, b : a + b,  sayilar)
print(toplam)
def topla(sayi1, sayi2):
    return sayi1 + sayi2
toplam3 = reduce(topla, sayilar)
print(toplam3)

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 kullanrak 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 kullanrak 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")

"""ZİP 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 içlü 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 kabolur
print(list(enumerate(rehber.values()))) # bu şekilde de değerler üzerinden
bir numaralandırma işlemi yapılabiilir


"""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 (İstediğimiz 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()
```

```python
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","İyiyim, 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.
```

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

@calismaSüresiHesapla
def karekökAl(sayilar):
    sayilar = [sqrt(sayi) for sayi in sayilar]
    print(sayilar)
sayilar = list(range(0,1000))
karekökAl(sayilar)
print("------------------------")
@calismaSüresiHesapla
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 kısası
    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 kelimesiyle
#       ulaşılmak istenmektedir.
#       Bu kullanıların yaş ortalamasının üzerinde kalan kullanıcıyı
filitreleyen programı yazınız.

user1 = {"isim":"fatih", "doğumTarihi":1990, "email":"ssdgdsgs@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":"ssdgdsgs@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)#istemesek 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 datatime modülünün
datetime sınıfı import edilmelidir.
"""OUTPUT - ÇIKTI
admin başarılı bir şekilde sisteme girdi. Tarih: 2021-09-11 03.11.33
asdmin adında bir giriş denendi.
Tarih: 2021-09-11 03.11.38 admin hatalı şifre girdi. Tarih:021-09-11
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"
_passward = 1234

@log
def sistemGiris(username, password):
    if username != _username:
        print("Böyle bir kullanıcı bulunmamaktadır")
        return username + "adında bir giriş denendi"
    elif password != _passward:
        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)
```