Temel Programlama 1

Öğr. Gör. Furkan DURMUŞ

Samsun Üniversitesi
Teknik Bilimler
Meslek Yüksekokulu

Bu Haftanın Ders Kazanımları

Bu haftayı bitirdiğinizde,

{ 1 } Sayısal Veri Tipleri ve Bazı Matematiksel Fonksiyonlar

{ 2 } String Veri Tipi ve Bool Veri Tipi

{ 3 } Kullanıcıdan Veri Alma

{ 4 } Veri Tipi Dönüşüm İşlemler

{ 5 } listeler

{ 6 } Liste Operatör ve Fonksiyonları

öğrenmiş olacaksınız.

Sayısal
Veri Tipleri

{01} Sayısal Veri Tipleri

Numbers sayısal verileri tutan veri tiplerine verilen addır.

Python’da sayısal veri tipleri genel olarak int, float ve complex veri tipleridir.

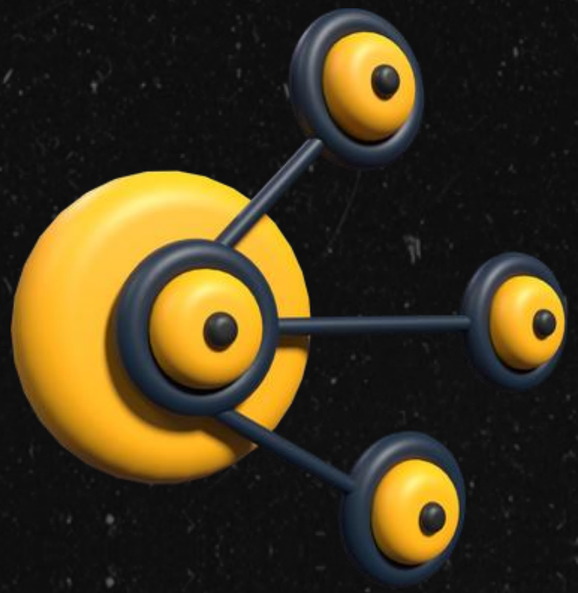
Bu derste int ve float veri tiplerinden bahsedilecektir.

{01} Sayısal Veri Tipleri

Int veri tipi tam sayı değerleri tutarken; float veri tipi ondalıklı değerleri tutar.

Bu noktada tüm tam sayıların da ondalıklı olarak ifade edilebileceğini unutmayınız. Örneğin 3 tam sayısı (normalde int) 3.00 şeklinde ifade edildiğinde float olarak da tanımlanabilir.

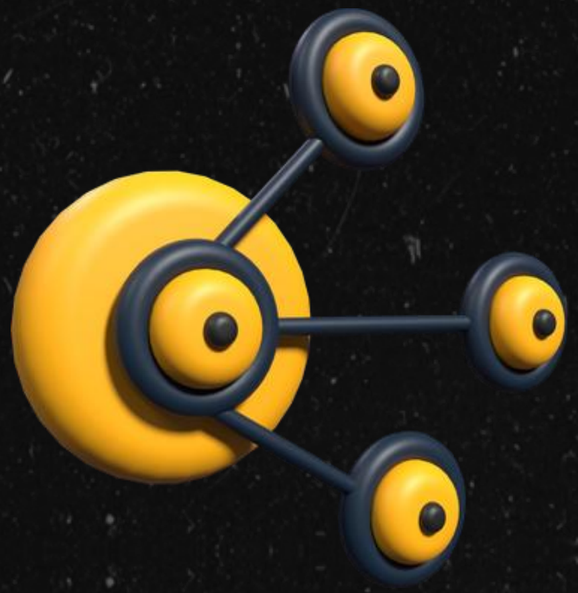
{01} Sayısal Veri Tipleri



```
PI_SAYISI = 3.14
sayi = 1919

print(PI_SAYISI)
print(sayi)

// ÇIKTI
3.14
1919
```

{01} Sayısal Veri Tipleri

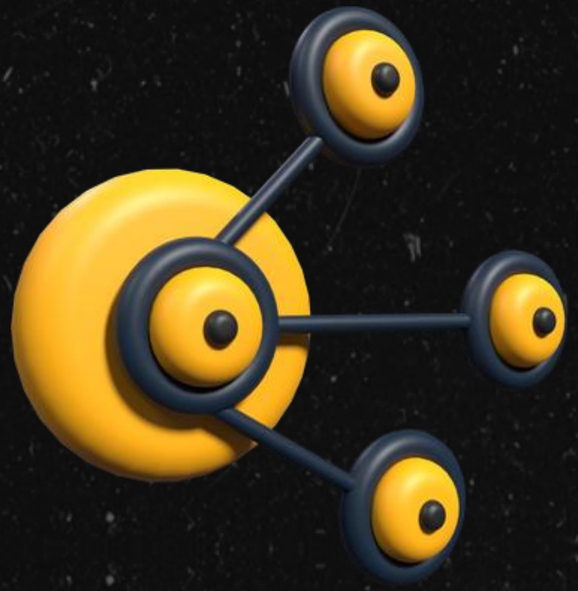
type() fonksiyonu ile değişkenlerin tip sınıfına erişilebilir.



```
PI_SAYISI = 3.14
sayi = 1919

print(type(PI_SAYISI))
print(type(sayi))
```

{01} Sayısal Veri Tipleri

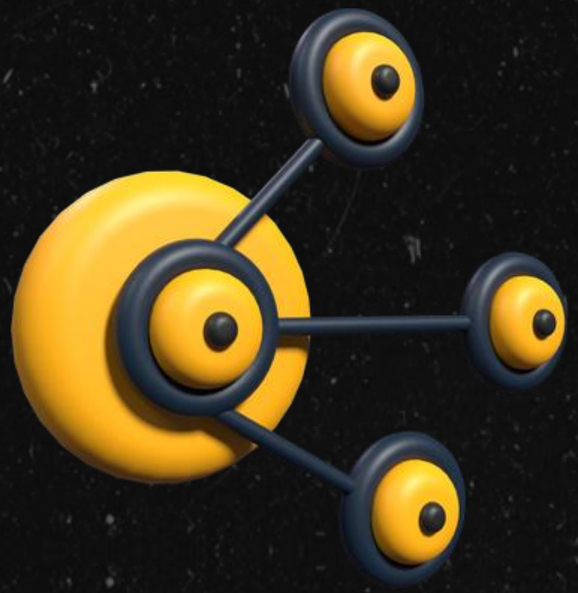
type() fonksiyonu ile değişkenlerin tip sınıfına erişilebilir.



```
PI_SAYISI = 3.14
sayi = 1919

print(type(PI_SAYISI))
print(type(sayi))

ÇIKTI:
<class 'float'>
<class 'int'>
```

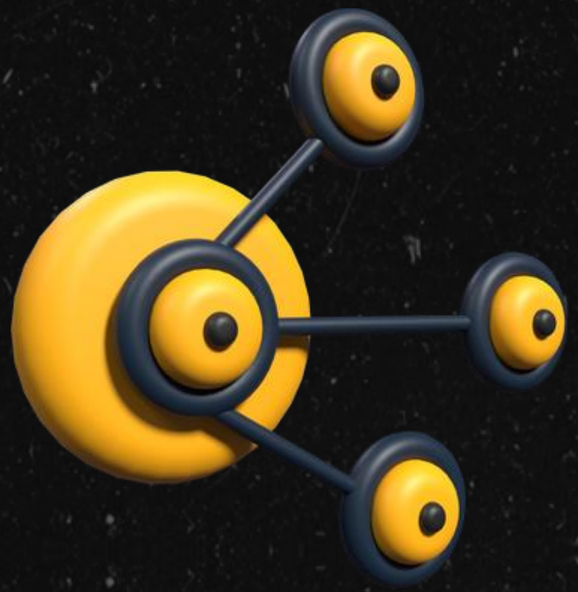
{01} Sayısal Veri Tipleri

19 ve 19.0 sayılarının eşit olup olmadığını kontrol eden kodu yazınız.



{01} Sayısal Veri Tipleri

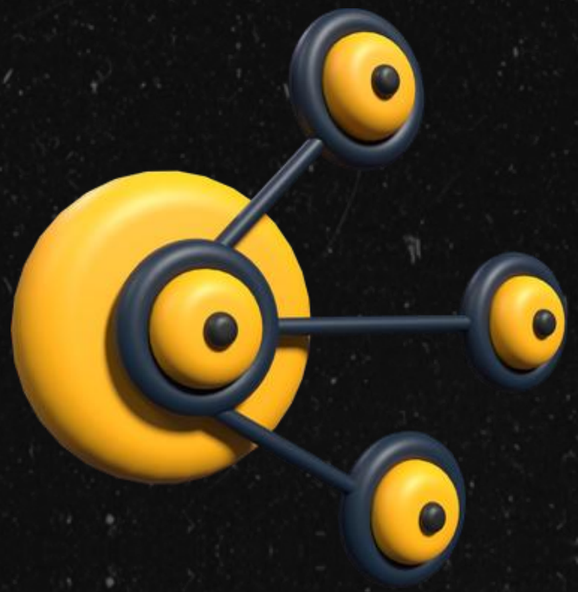
19 ve 19.0 sayılarının eşit olup olmadığını kontrol eden kodu yazınız.



```
sayi1 = 19
sayi2 = 19.0

print(sayi1 == sayi2)
```

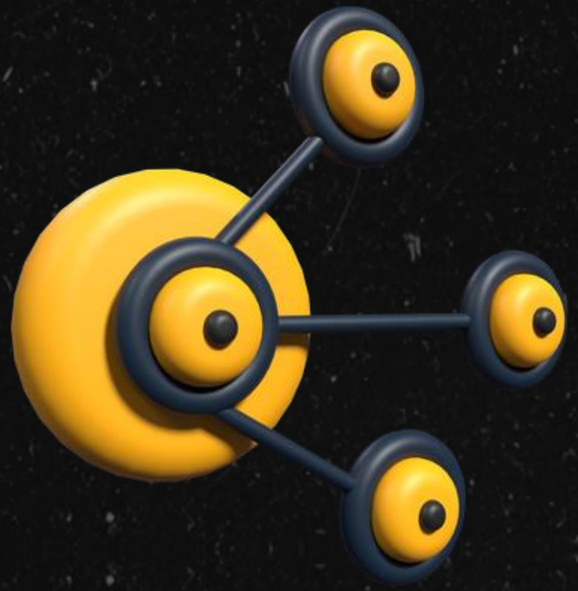
{01} Sayısal Veri Tipleri

19 ve 19.0 sayılarının tiplerinin eşit olup olmadığını kontrol eden kodu yazınız.



{01} Sayısal Veri Tipleri

19 ve 19.0 sayılarının tiplerinin eşit olup olmadığını kontrol eden kodu yazınız.



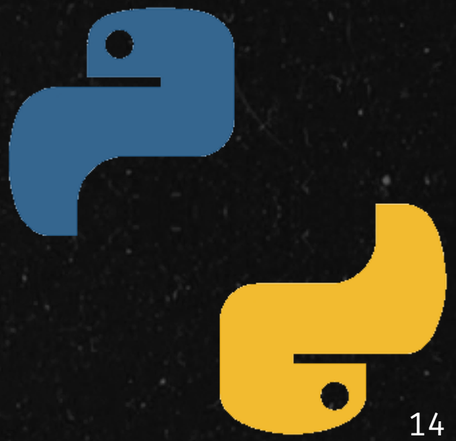
```
sayi1 = 19
sayi2 = 19.0

print(type(sayi1) == type(sayi2))
```

Bazı Matemaktiksel Fonksiyonlar

{01} Bazı Matemaktiksel Fonksiyonlar

{01} abs()

```
sayi = -5
print(abs(sayi))
```



Öğr. Gör. Furkan DURMUŞ

{01} Bazı Matemaktiksel Fonksiyonlar

{01} Mutlak Değer

abs()

```
sayi = -5
print(abs(sayi))
```

Öğr. Gör. Furkan DURMUŞ

{01} Bazı Matemaktiksel Fonksiyonlar

{02} pow()

```
sayi1 = 5
sayi2 = 3
print(pow(sayi1, sayi2))
```


Öğr. Gör. Furkan DURMUŞ

{01} Bazı Matemaktiksel Fonksiyonlar

{02} Üs Alma

pow()

```
sayi1 = 5
sayi2 = 3
print(pow(sayi1, sayi2))
```

Öğr. Gör. Furkan DURMUŞ

{01} Bazı Matemaktiksel Fonksiyonlar

{03} max()

```
sayi1 = 5
sayi2 = 3
sayi3 = 6
print(max(sayi1, sayi2, sayi3))
```

Öğr. Gör. Furkan DURMUŞ

{01} Bazı Matemaktiksel Fonksiyonlar

{03} En Büyük Sayı

max()

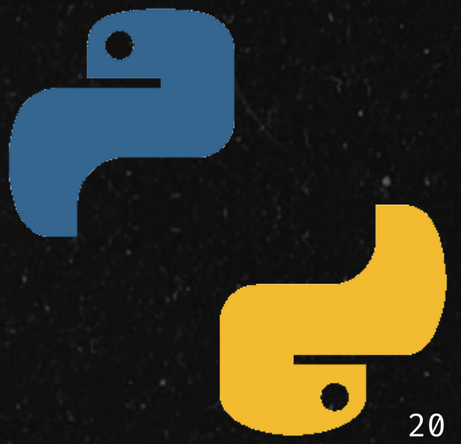
```
sayi1 = 5
sayi2 = 3
sayi3 = 6
print(max(sayi1, sayi2, sayi3))
```

Öğr. Gör. Furkan DURMUŞ

{01} Bazı Matemaktiksel Fonksiyonlar

{04} min()

```
sayi1 = 5
sayi2 = 3
sayi3 = 6
print(min(sayi1, sayi2, sayi3))
```



{01} Bazı Matemaktiksel Fonksiyonlar

{05} round()

```
sayi1 = 5.3
sayi2 = 3.5
sayi3 = 6.8
print(round(sayi1))
print(round(sayi2))
print(round(sayi3))
```

Öğr. Gör. Furkan DURMUŞ

{01} Bazı Matemaktiksel Fonksiyonlar

{05} Yuvarlama

round()

```
sayi1 = 5.3
sayi2 = 3.5
sayi3 = 6.8
print(round(sayi1))
print(round(sayi2))
print(round(sayi3))
```

{01} Bazı Matemaktiksel Fonksiyonlar

Python Built-in functions

İle diğer hazır fonksiyonları araştırabilirsiniz.

Öğr. Gör. Furkan DURMUŞ

String
Veri Tipi

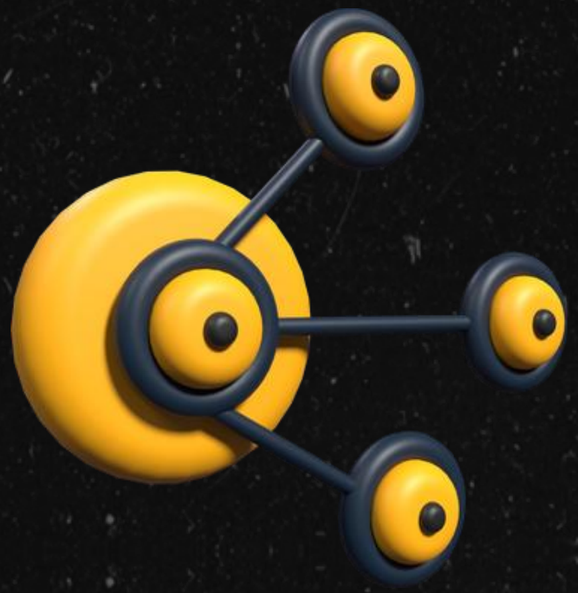
{02} String Veri Tipi

Python programlama dilinde metinsel ifadeler tek tırnak yada çift tırnak ile ifade edilebilir.

"merhaba dünya" == ‘merhaba dünya’

{02} String Veri Tipleri

Aşağıdaki uygulamayı bilgisayarınızda çalıştırınız.



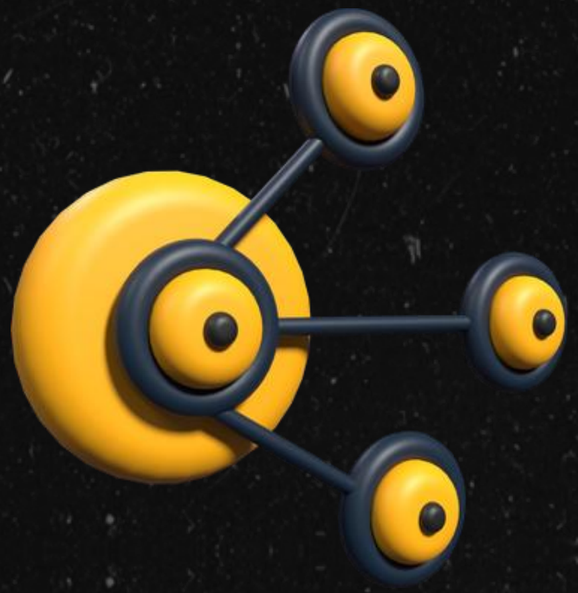
```
sayi1 = 19
sayi2 = 19.0
ders  = "Temel Programlama"

print(type(sayi1))
print(type(sayi2))
print(type(ders))
```

{02} Boolean Veri Tipleri

Python programlama dilinde doğru/yanlış (True/False) değerlerini tutan ifadelere boolean ifadeler (bool türü) denir.

{02} Boolean Veri Tipleri

Aşağıdaki uygulamayı bilgisayarınızda çalıştırınız.



```
sayi1 = 19
sayi2 = 19.0
ders  = "Temel Programlama"
dersi_gecti  = True

print(type(sayi1))
print(type(sayi2))
print(type(ders))
print(type(dersi_gecti))
```

Kullanıcıdan Veri Alma

{03} Kullanıcıdan Veri Alma

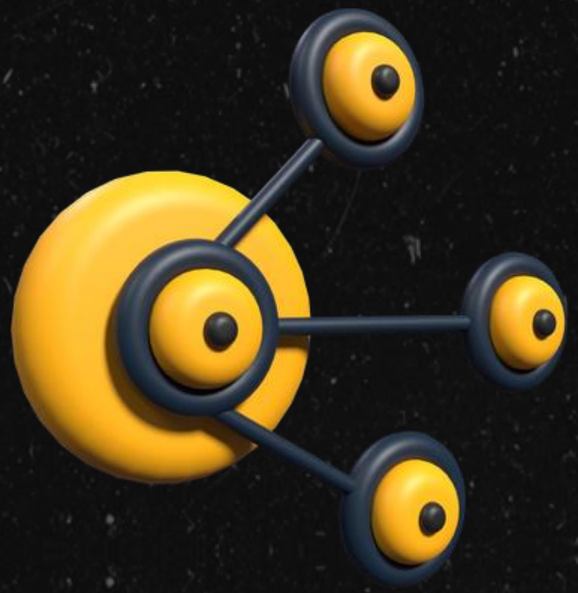
Python'da kullanıcıdan veri almak için

```
input()
```

fonksiyonu kullanılır.

{03} Kullanıcıdan Veri Alma

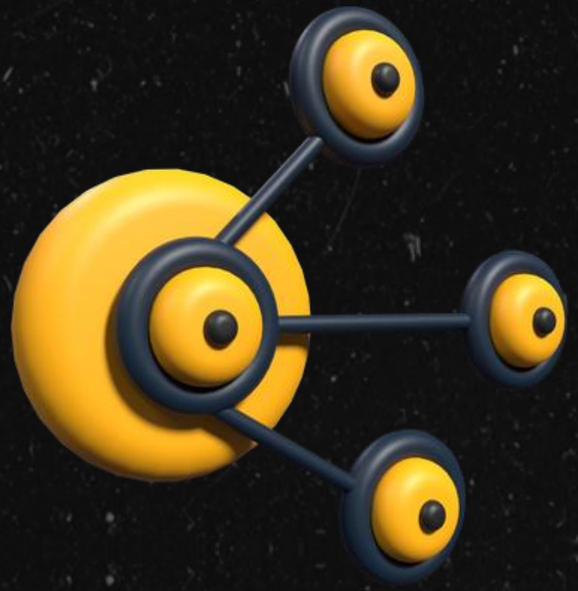
Kullanıcıdan ismini alma işlemi



```
print("İsminiz :")

isim = input()

print("Merhaba, " + isim)
```

{03} Kullanıcıdan Veri Alma

Kullanıcıdan ismini alma işlemi



```
isim = input("isminiz : ")

print("Merhaba, " + isim)
```

{03} Kullanıcıdan Veri Alma

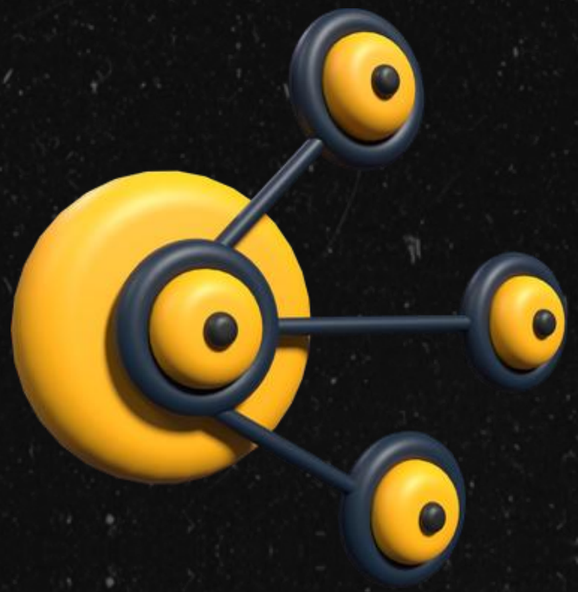
Kodu bilgisayarlarınız da yazınız.

```
takim = input("Hangi takımı tutuyorsun :")

print("En büyük : " + takim)
```

{03} Kullanıcıdan Veri Alma

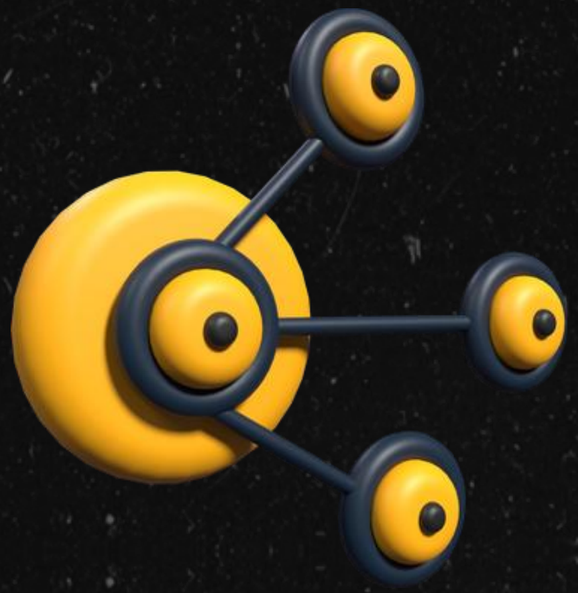
Furkandurmus.com adresini açınız ve alt kımısdaki Python_lab aracını açarak size verilen şifre ile giriş yapınız.

furkandurmus.com/Python_lab



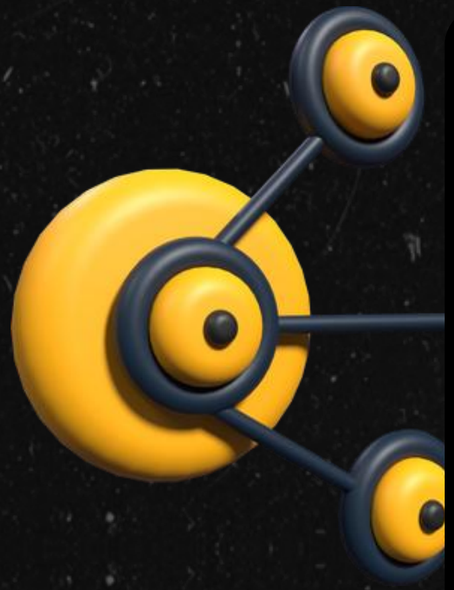
{02} Kullanıcıdan Veri Alma

Kullanıcıdan isim ve soy isim alınız. İsim ve soy isim arasına . (nokta) koyarak @samsun.edu.tr uzantılı mail adresini ekrana yazdırınız.

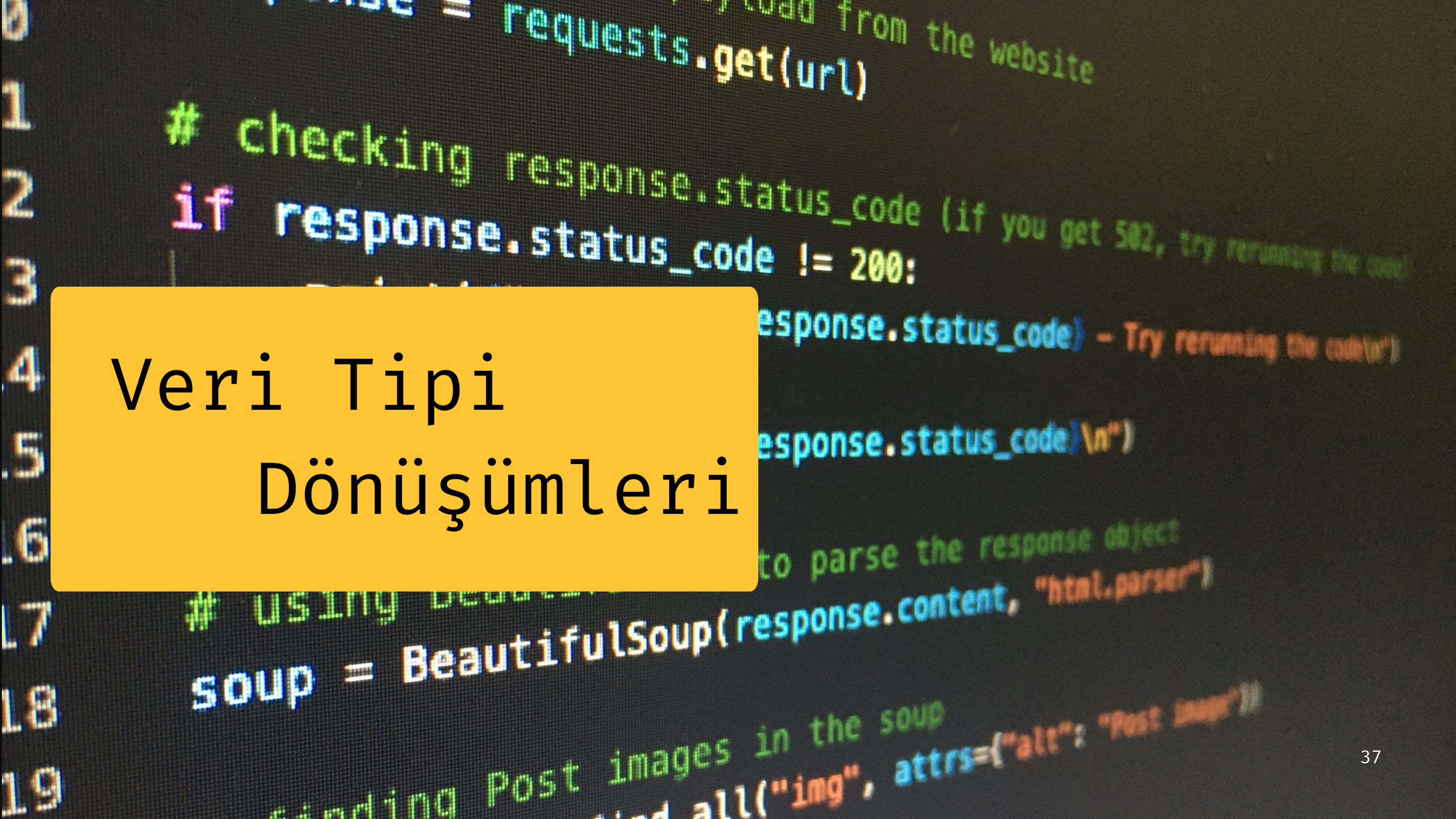


{03} Kullanıcıdan Veri Alma

Kullanıcıdan isim ve soy isim alınız. İsim ve soy isim arasına . (nokta) koyarak @samsun.edu.tr uzantılı mail adresini ekrana yazdırınız.



```
ad = input("Adınız : ")
soyad = input("Soyadınız : ")

eposta = ad + "." + soyad + "@samsun.edu.com"
print("Oluşturulan e-posta adresi:", eposta)
```



Veri Tipi Dönüşümleri

{04} Veri Tipi Dönüşümleri

Aşağıdaki kodu çalıştırınız.

```
sayi = input("bir sayi giriniz")
print(sayi * sayi)
```


Öğr. Gör. Furkan DURMUŞ

{04} Veri Tipi Dönüşümleri

```
sayi = input("bir sayi giriniz")
print(sayi * sayi)
```

Ne hatası alındı?

{04} Veri Tipi Dönüşümleri

Aşağıdaki kodu çalıştırınız.

```
sayi = input("bir sayi giriniz")

print(type(sayi))
```

{04} Veri Tipi Dönüşümleri

input()

fonksiyonu kullanıcının girdiği her şeyi string (yani metin) olarak alır.

{04} Veri Tipi Dönüşümleri

input()

fonksiyonu kullanıcının girdiği her şeyi string (yani metin) olarak alır.

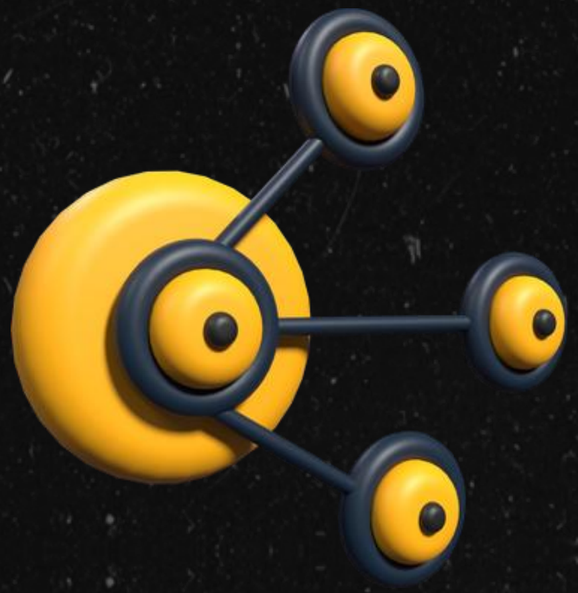
Eğer bu veriyi sayılarla işlem yapmak için kullanmak istersek, veri tipi dönüşümü yapmamız gerekir.

Bunun için inputtan önce çevirmek istediğimiz tip yazılır.

Öğr. Gör. Furkan DURMUŞ

{04} Veri Tipi Dönüşümleri

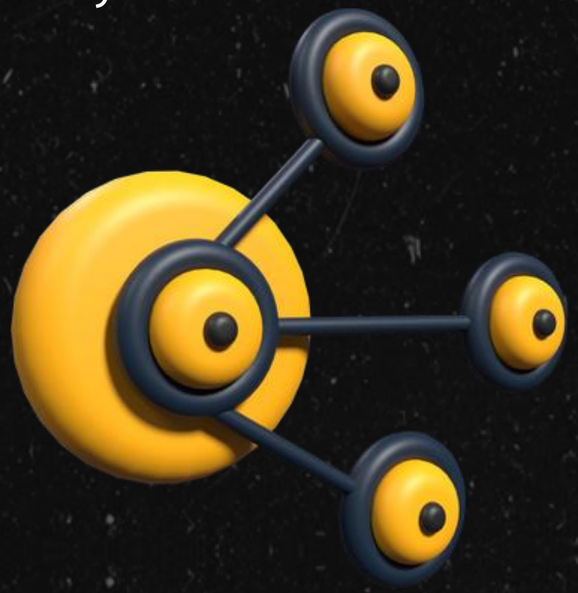
Kullanıcıdan alınan sayı ile matematiksel işlem yapma



```
sayi = int(input("sayi giriniz :"))
print(type(sayi))
print(sayi * sayi)
```

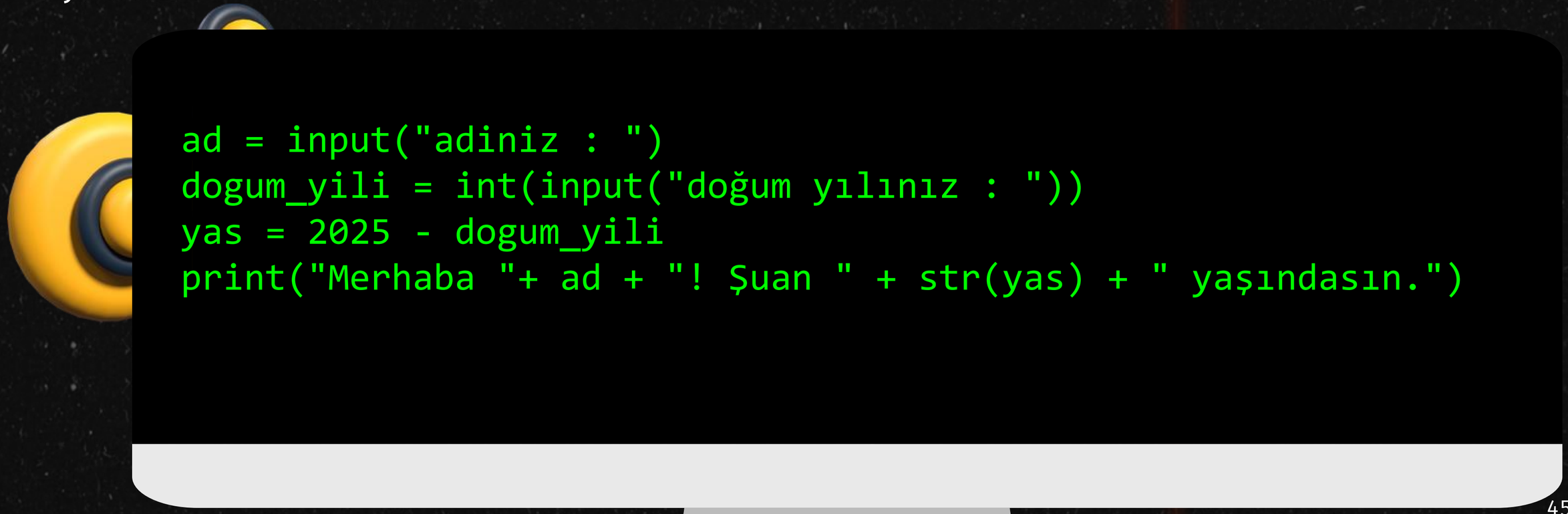
{04} Veri Tipi Dönüşümleri

Kullanıcıdan adını, doğum yılını alıp ekrana ‘Merhaba ..... ! Şuan .... yaşındasın.’ şeklinde çıktı alan kodu ekrana yazdırınız.



{04} Veri Tipi Dönüşümleri

Kullanıcıdan adını, doğum yılını alıp ekrana ‘Merhaba ..... ! Şuan .... yaşındasın.’ şeklinde çıktı alan kodu ekrana yazdırınız.

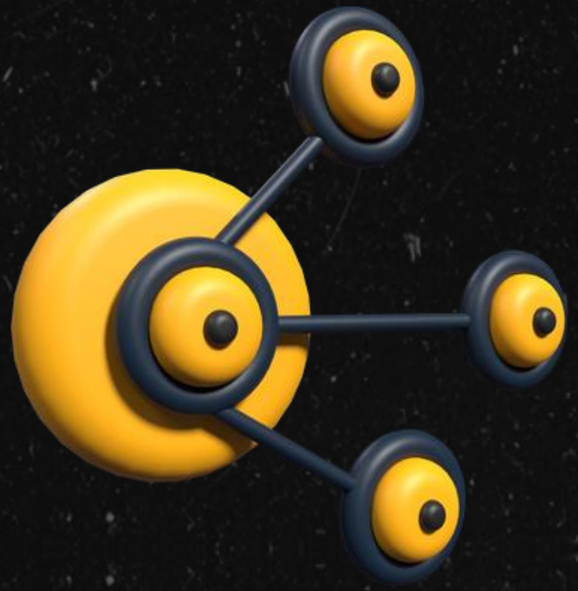


```
ad = input("adiniz : ")
dogum_yili = int(input("doğum yılınız : "))
yas = 2025 - dogum_yili
print("Merhaba "+ ad + "! Şuan " + str(yas) + " yaşındasın.")
```


Öğr. Gör. Furkan DURMUŞ

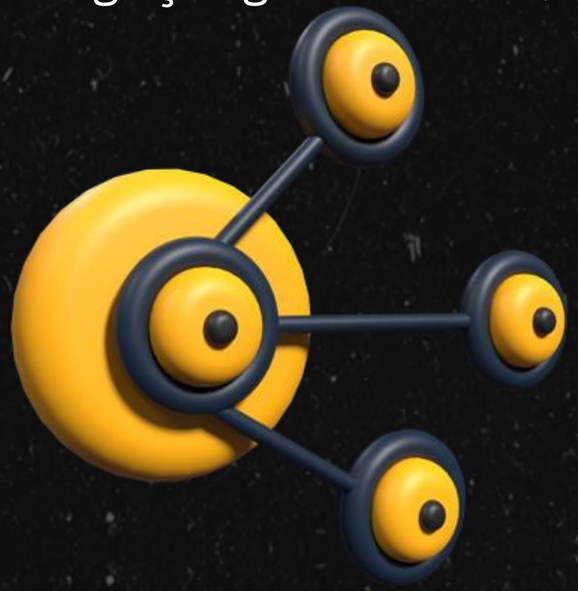
{04} Veri Tipi Dönüşümleri

Kullanıcıdan 2 sayı alınız ve çarpımlarının sonucunu ekrana yazdırınız. Ardından 2 sayı daha alıp aldığınız yeni sayıların farkını ekrana yazdırınız. Kod sadece 1 kez çalıştırılacak, kodu tek seferde yazınız.



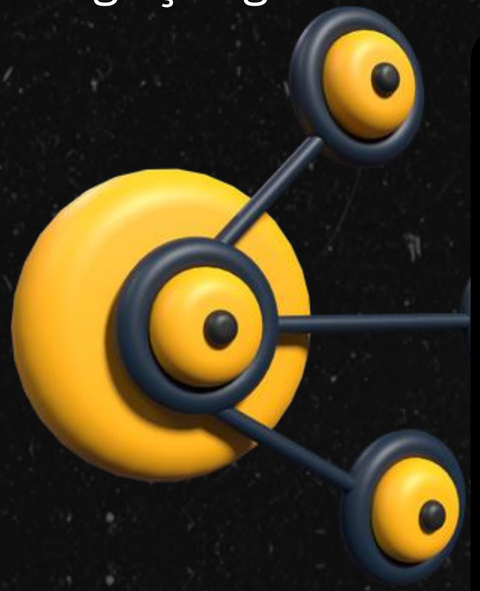
{04} Veri Tipi Dönüşümleri

Kullanıcıdan Temel Programlama dersinde aldığı 4 notu isteyiniz. Bu notların ortalamasını, en yüksek notu, ve ortalama 50 nin üzerinde ise dersten geçtiğini True / False olarak ekrana yazdırınız.



{04} Veri Tipi Dönüşümleri

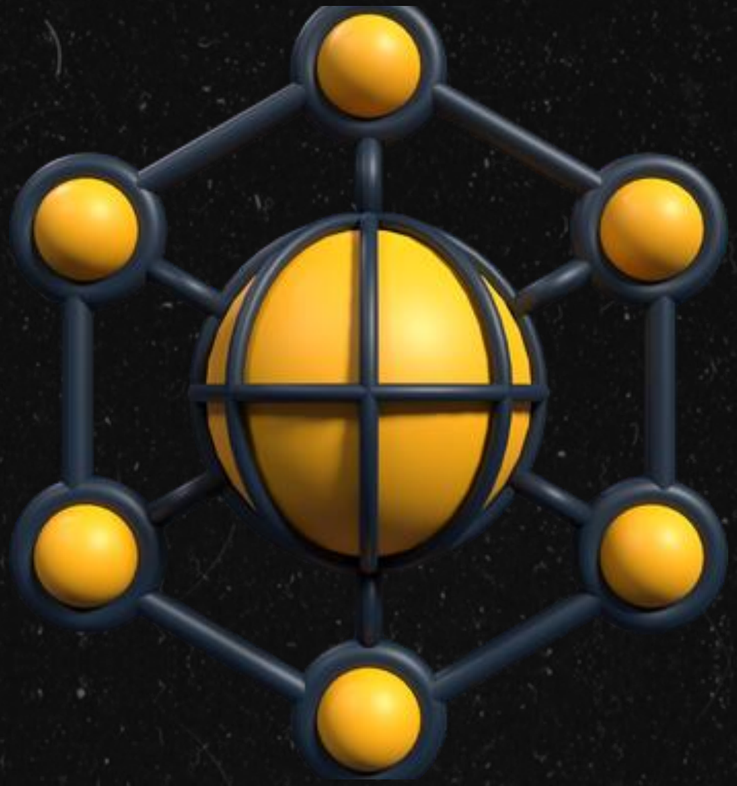
Kullanıcıdan Temel Programlama dersinde aldığı 4 notu isteyiniz. Bu notların ortalamasını, en yüksek notu, ve ortalama 50 nin üzerinde ise dersten geçtiğini True / False olarak ekrana yazdırınız.



```
not1 = int(input("1. notu girin: "))
not2 = int(input("2. notu girin: "))
not3 = int(input("3. notu girin: "))
not4 = int(input("4. notu girin: "))
ortalama = (not1 + not2 + not3 + not4)  / 4
en_yuksek = max(not1,not2,not3,not4)
gecti_mi = ortalama > 50
print(f"Notların ortalaması: {ortalama}")
print(f"En yüksek not: {en_yuksek}")
print(f"Dersten geçti mi? {gecti_mi}")
```

Listeler

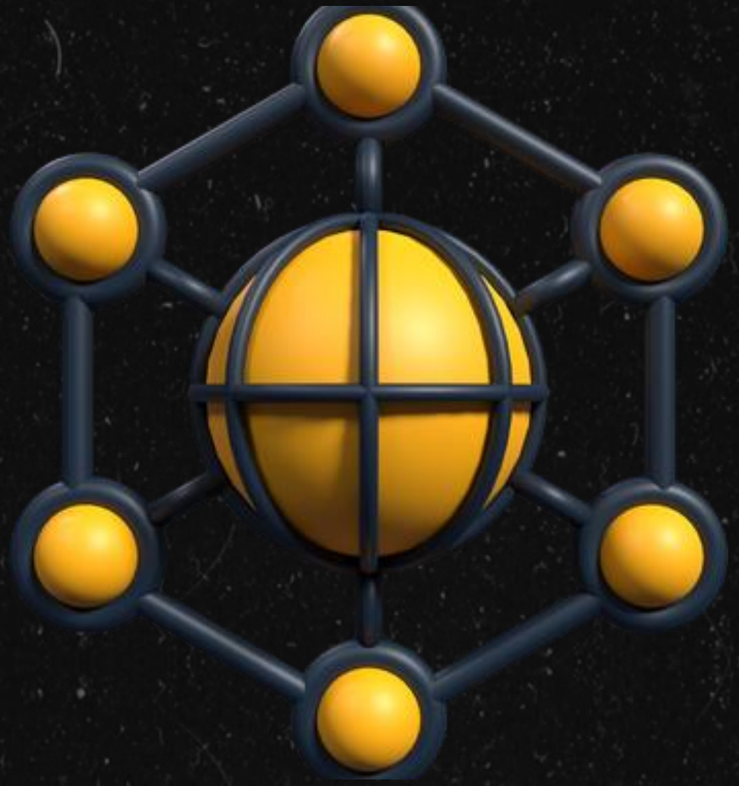
{05} Listeler



Farklı verilerin bir dizi hâlinde tutulduğu koleksiyonlara liste adı verilir.

Daha önce int, float, string gibi veri türlerini öğrenmiştiniz. Bu veri tiplerini kullanarak tek bir veriyi tutabilirsiniz.

{05} Listeler



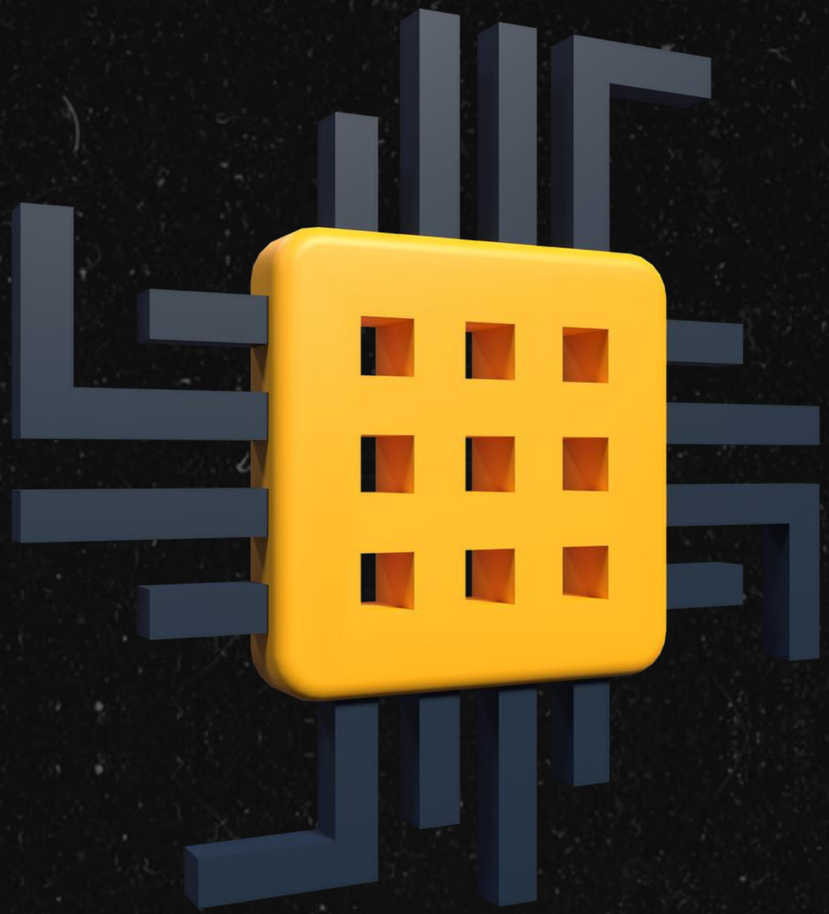
Birden fazla veriyi sıralı ve değiştirile bilen bir yapıda tutmak için listeler kullanılır.

Listeler ile farklı veri tiplerini tutabilirsiniz.

Python programlama dilinde listeler köşeli parantez ile tanımlanmaktadır.

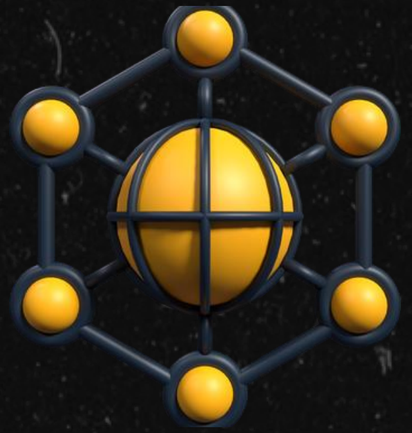
Öğr. Gör. Furkan DURMUŞ

{05} Listeler



```
ilk_liste=[“Ankara”, 312, 0.6]

print(ilk_liste)
```

{05} Listeler



Liste veri tipi için tanımlanan ilk liste incelendiğinde sırasıyla string, integer ve float tiplerinin bir arada kullanıldığı görülmektedir.

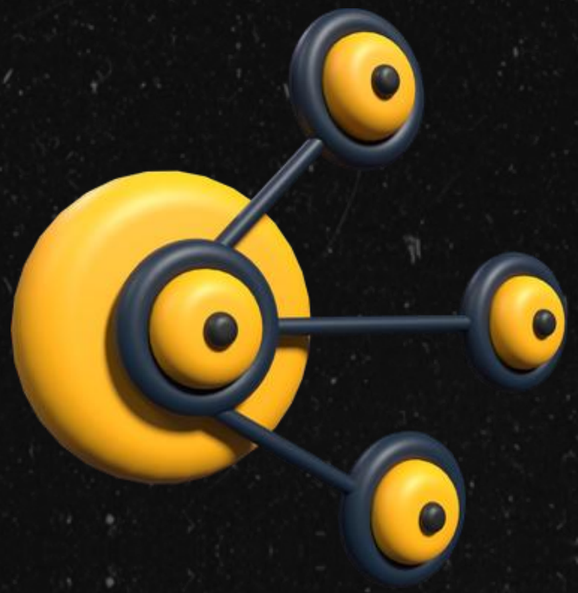
Üç elemanı olan bu listeyi yazdırmak için daha önce kullanılan print() fonksiyonunu kullanmanız gerekmektedir.

Bu örneğin liste veri tipinde olduğunu doğrulamak için type fonksiyonu kullanılabilir. type(ilk_liste) kod satırı çalıştırıldığında ekran çıktısı <class 'list'> olacaktır.

Öğr. Gör. Furkan DURMUŞ

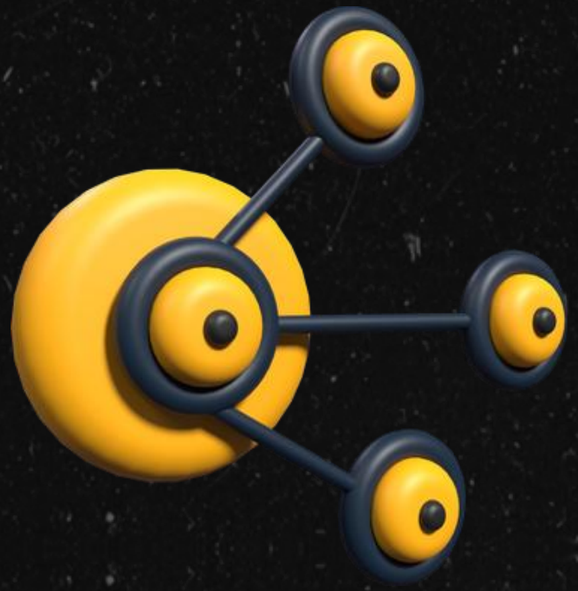
{05}Listeler

Elemanları haftanın günleri olan bir liste oluşturunuz ve ekrana yazdırınız.



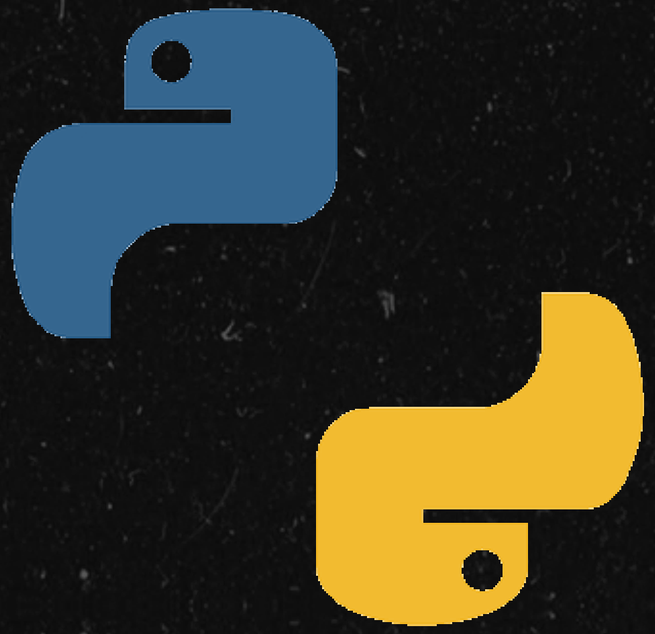
{05}Listeler

Elemanları haftanın günleri olan bir liste oluşturunuz ve ekrana yazdırınız.



```
gunler = ["Pazartesi", "Salı",
"Çarşamba", "Perşembe", "Cuma",
"Cumartesi", "Pazar"]

print(gunler)
```

{05} Listeler – Index Kullanımı



Liste içindeki elemanlara erişmek için ilgili elemanın indeksi kullanılır. Bazı kaynaklarda indis olarak da karşınıza çıkabilir.

İlk elemanın indisi her zaman 0 (sıfır) olarak kabul edilir.

Öğr. Gör. Furkan DURMUŞ

{05} Listeler – Index Kullanımı

```
sehirler=[“Ankara”, “Bursa”, “Çanakkale”, “Denizli”, “Eskişehir”]
```

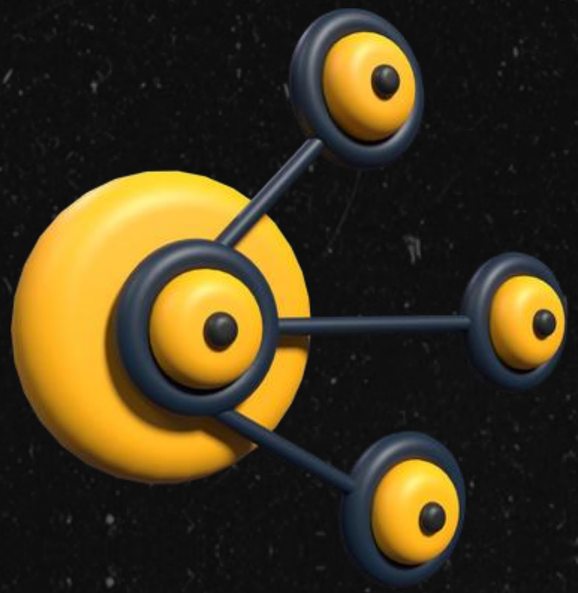
{05} Listeler – Index Kullanımı

```
sehirler=[“Ankara”, “Bursa”, “Çanakkale”, “Denizli”, “Eskişehir”]
```

İndeksi	0	1	2	3	4
Değeri	Ankara	Bursa	Çanakkale	Denizli	Eskişehir

{05} Listeler – Index Kullanımı

İndeksi 2 olan elemanı ekrana yazdırınız.



```
sehirler=[“Ankara”, “Bursa”,
“Çanakkale”, “Denizli”,
“Eskişehir”]


print(sehirler[2])
```

{05} Listeler – Index Kullanımı

```
ders=['K','O','D','L','A','M','A']
```

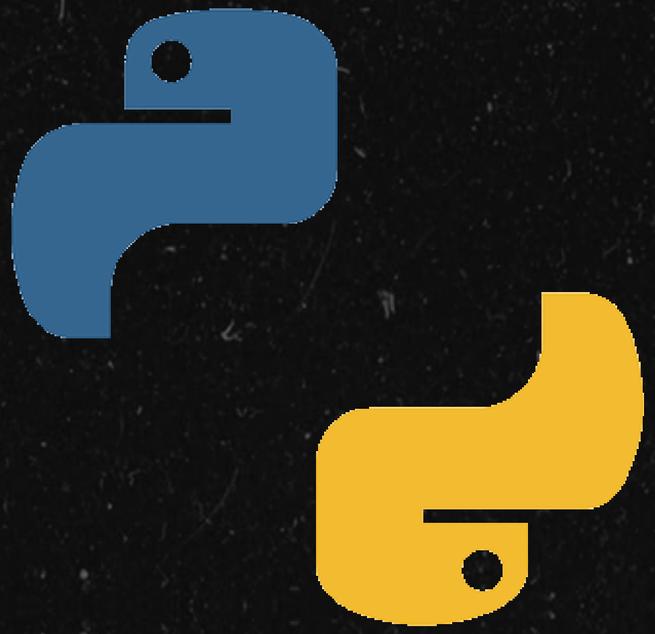
İşlem	Sonuç
print(ders[0])	
print(ders[2])	
print(ders[5])	

{05} Listeler – Index Kullanımı

```
ders=['K','O','D','L','A','M','A']
```

İşlem	Sonuç
print(ders[0])	**K**
print(ders[2])	**D**
print(ders[5])	**M**

{05} Listeler – Index Kullanımı



İndeksler negatif olarak da yazılabilir.

Örneğin -1 indeksi sondaki elemanı gösterirken -2 indeksi sondan bir öncekini gösterir.

```
sehirler=[“Ankara”, “Bursa”, “Çanakkale”, “Denizli”, “Eskişehir”]

print(sehirler[-2])
```

{05} Listeler – Index Kullanımı

```
ders=['K','O','D','L','A','M','A']
```

İşlem	Sonuç
print(ders[-1])	
print(ders[-4])	
print(ders[-3])	

{05} Listeler – Index Kullanımı

```
ders=['K','O','D','L','A','M','A']
```

İşlem	Sonuç
print(ders[-1])	**A**
print(ders[-4])	**L**
print(ders[-3])	**A**

{05} Listeler – Index Kullanımı

```
          0   1   2   3   4   5   6
ders=['K','O','D','L','A','M','A']
         -7  -6  -5  -4  -3  -2  -1
```

{05} Listeler – Index Kullanımı

Listelerde indekslerle birlikte iki nokta (:) operatörü kullanılarak istenilen elemanlara ulaşılabilir.

Bu işlem için `liste[başlangıç indeksi:bitiş indeksi]` yapısı kullanılır.

Başlangıç indeksi dahil olurken bitiş indeksi dahil olmaz.

{05} Listeler – Index Kullanımı

```
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
print(asal_sayilar[1:4])
```

{05} Listeler – Index Kullanımı

```
                    0  1  2  3   4   5   6   7   8
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
print(asal_sayilar[1:4])
```

ÇIKTI //
[3,5,7]

{05} Listeler – Index Kullanımı

```
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
print(asal_sayilar[5:])
```

{05} Listeler – Index Kullanımı

```
                    0  1  2  3   4   5   6   7   8
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
print(asal_sayilar[5:])
```

```
ÇIKTI //
[13,17,19,23]
```

{05} Listeler – Index Kullanımı

```
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
print(asal_sayilar[:5])
```

{05} Listeler – Index Kullanımı

```
                    0  1  2  3   4   5   6   7   8
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
print(asal_sayilar[:5])
```

ÇIKTI //
[2,3,5,7,11]

{05} Listeler – Index Kullanımı

```
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
print(asal_sayilar[0:6:2])
```

{05} Listeler – Index Kullanımı

```
                    0  1  2  3   4   5   6   7   8
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
print(asal_sayilar[0:6:2])
```

ÇIKTI //
[0,5,11]

{05} Listeler – Index Kullanımı

```
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
print(asal_sayilar[::3])
```

{05} Listeler – Index Kullanımı

```
                    0  1  2  3   4   5   6   7   8
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
print(asal_sayilar[::3])
```

ÇIKTI //
[2,7,17]

{05} Listeler – Index Kullanımı

```
liste[ başlangıç indeksi : bitiş indeksi : artış ]
```

{05} Listeler – Index Kullanımı

Aşağıdaki listeyi tersten yazdıran kodu yazınız.

```
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
```

{05} Listeler – Index Kullanımı

Aşağıdaki listeyi tersten yazdıran kodu yazınız.

```
asal_sayilar=[2, 3, 5, 7, 11, 13, 17, 19, 23]
              -9 -8 -7 -6  -5  -4 -3  -2   -1
print(asal_sayilar[::-1])
```

```
ÇIKTI //
[23, 19, 17, 13, 11, 7, 5, 3, 2]
```

{05} Listeler – Index Kullanımı

```
tek_sayilar=[3, 5, 7, 9, 11, 13, 15, 17, 19]
```

İşlem	Sonuç
print(tek_sayilar[0:6])	
print(tek_sayilar[2:5])	
print(tek_sayilar[3:8])	
print(tek_sayilar[:5])	
print(tek_sayilar[3:])	
print(tek_sayilar[0:8:2])	
print(tek_sayilar[::3])	

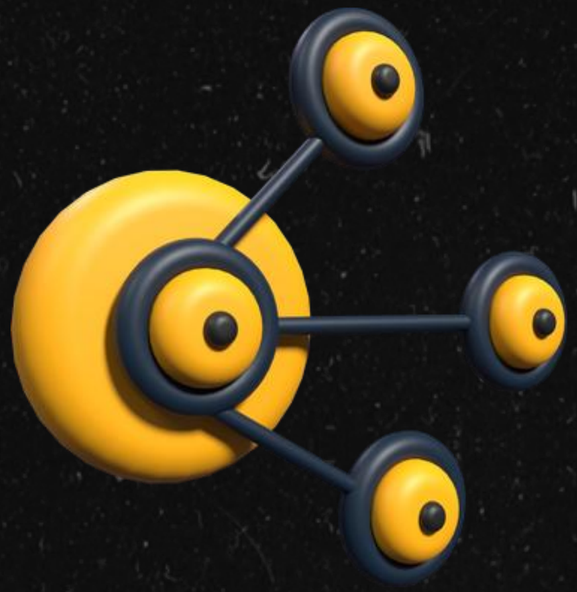
{05} Listeler – Index Kullanımı

```
tek_sayilar=[3, 5, 7, 9, 11, 13, 15, 17, 19]
```

İşlem	Sonuç
print(tek_sayilar[0:6])	**[3, 5, 7, 9, 11, 13]**
print(tek_sayilar[2:5])	**[7, 9, 11]**
print(tek_sayilar[3:8])	**[9, 11, 13, 15, 17]**
print(tek_sayilar[:5])	**[3, 5, 7, 9, 11]**
print(tek_sayilar[3:])	**[9, 11, 13, 15, 17, 19]**
print(tek_sayilar[0:8:2])	**[3, 7, 11, 15]**
print(tek_sayilar[::3])	**[3, 9, 15]**

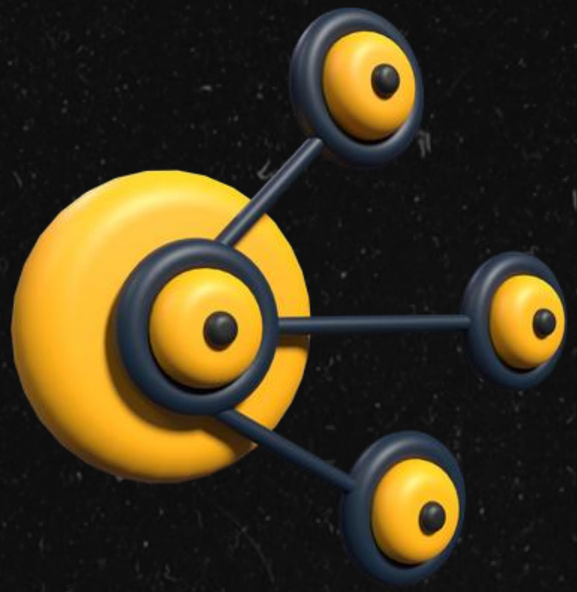
{05} Listeler

Bir sınıfta öğrenci notları 34, 25, 75, 79, 100, 83, 65, 18, 91 olarak kayıt edilmiştir. Öğrenci notlarından oluşan bir liste tanımlayınız ve en yüksek not alan öğrenci ve en düşük not alan öğrencinin not farkını ekrana yazdırınız.



{05} Listeler

Bir sınıfta öğrenci notları 34, 25, 75, 79, 100, 83, 65, 18, 91 olarak kayıt edilmiştir. Öğrenci notlarından oluşan bir liste tanımlayınız ve en yüksek not alan öğrenci ve en düşük not alan öğrencinin not farkını ekrana yazdırınız.



```
notlar = [34, 25, 75, 79, 100, 83, 65, 18, 91]

print(max(notlar) - min(notlar))
```


Öğr. Gör. Furkan DURMUŞ

{05} Listeler – Eleman Değiştirme

Liste veri tipindeki bir elemanın indeksi kullanılarak yeni değer atanabilir.

{05} Listeler – Eleman Değiştirme

```
ornek=['Y','A','N','I','T'] #YANIT kelimesinin harflerinden bir liste oluşturuldu.
print(ornek)                #Liste ekrana yazdırıldı.
ornek[0]='K'                #Listenin ilk elemanı (indeksi sıfır) K olarak değiştirildi.
print(ornek)                #Listenin yeni değeri KANIT kelimesinin harflerine dönüştü.
```

Liste Operatör
ve Fonksiyonları

{06} Listeler – len Operatörü

Listelerin eleman sayısına ulaşmak için İngilizce uzunluk anlamına gelen length kelimesinin kısaltması olan `len()` fonksiyonu kullanılır.

Öğr. Gör. Furkan DURMUŞ

{06} Listeler – len Operatörü

```
sayi=[20, 40, 60, 80]
print(len(sayi))
```

{06} Listeler – len Operatörü

```
sayi=[20, 40, 60, 80]
print(len(sayi))
```

ÇIKTI //
4

{06} Listeler – in Operatörü

Bir elemanın listede olup olmadığını kontrol eder.

Eleman listede var ise True yok ise False çıktısını üretir.

{06} Listeler – in Operatörü

```
renkler=[“mavi”, “yeşil”, “kırmızı”, “mor”]

print(“mavi” in renkler)
```

{06} Listeler – in Operatörü

```
renkler=[“mavi”, “yeşil”, “kırmızı”, “mor”]

print(“mavi” in renkler)
```

ÇIKTI //
TRUE

Öğr. Gör. Furkan DURMUŞ

{06} Listeler – Append Fonksiyonu

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.append(“bellek”)
print(donanim)
```

{06} Listeler – Append Fonksiyonu

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.append(“bellek”)
print(donanim)
```

```
ÇIKTI //
[‘yazıcı’, ‘klavye’, ‘işlemci’, ‘bellek’, ‘sabit disk’, ‘bellek’]
```

{06} Listeler – Append Fonksiyonu

Append: Listenin sonuna eleman eklemek için kullanılır.

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
 donanim.append(“bellek”)
 print(donanim)
```

ÇIKTI //
[‘yazıcı’, ‘klavye’, ‘işlemci’, ‘bellek’, ‘sabit disk’, ‘bellek’]

{06} Listeler – Extend Fonksiyonu

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
yazilim=[“işletim sistemi”, “web tarayıcı”]
donanim.extend(yazilim)
```

{06} Listeler – Extend Fonksiyonu

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
yazilim=[“işletim sistemi”, “web tarayıcı”]
donanim.extend(yazilim)
```

```
ÇIKTI //
[‘yazıcı’, ‘klavye’, ‘işlemci’, ‘bellek’, ‘sabit disk’, ‘işletim sistemi’, ‘web tarayıcı’]
```

{06} Listeler – Extend Fonksiyonu

Extend: Listeleri birleştirmek için kullanılır.

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
yazilim=[“işletim sistemi”, “web tarayıcı”]
donanim.extend(yazilim)
```

ÇIKTI //
[‘yazıcı’, ‘klavye’, ‘işlemci’, ‘bellek’, ‘sabit disk’, ‘işletim sistemi’, ‘web tarayıcı’]

{06} Listeler – Insert Fonksiyonu

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.insert(2, “tarayıcı”)
print(donanim)
```

{06} Listeler – Insert Fonksiyonu

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.insert(2, “tarayıcı”)
print(donanim)

ÇIKTI//
[‘yazıcı’, ‘klavye’, ‘tarayıcı’, ‘işlemci’, ‘bellek’, ‘sabit disk’]
```

{06} Listeler – Insert Fonksiyonu

Insert: Listenin belirtilen konumuna (indeksine) eleman eklemek için kullanılır.

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.insert(2, “tarayıcı”)
print(donanim)
```

```
ÇIKTI//
[‘yazıcı’, ‘klavye’, ‘tarayıcı’, ‘işlemci’, ‘bellek’, ‘sabit disk’]
```

{06} Listeler – Remove Fonksiyonu

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.remove(“klavye”)
print(donanim)
```

{06} Listeler – Remove Fonksiyonu

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.remove(“klavye”)
print(donanim)

ÇIKTI //
[‘yazıcı’, ‘işlemci’, ‘bellek’, ‘sabit disk’]
```

{06} Listeler – Remove Fonksiyonu

Remove: Listenin içindeki değeri verilen elemanı siler.

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.remove(“klavye”)
print(donanim)
```

ÇIKTI //
[‘yazıcı’, ‘işlemci’, ‘bellek’, ‘sabit disk’]

{06} Listeler – Pop Fonksiyonu

Pop: Listede belirtilen konumdaki (indeks) elemanı siler.

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.pop(3)
print(donanim)
```

Ekran Çıktısı: [‘yazıcı’, ‘klavye’, ‘işlemci’, ‘sabit disk’]

{06} Listeler – Clear Fonksiyonu

Clear: Listenin tüm elemanlarını siler ve boş bir liste ortaya çıkarır.

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.clear()
print(donanim)
```

Ekran Çıktısı: [ ]

{06} Listeler – Index Fonksiyonu

Index: Bir elemanın listedeki konumunu bulur.

Listedeki “sabit disk” elemanının indeksini bulunuz.

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
print(donanim.index(“sabit disk”))
```

Ekran Çıktısı: 4

{06} Listeler – Count Fonksiyonu

Count: Listede belirtilen elemandan kaç adet olduğunu bulur.

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”, “klavye”]
say=donanim.count(“klavye”)
print(say)
```

Ekran Çıktısı: 2

{06} Listeler – Sort Fonksiyonu

Sort: Listenin içindeki elemanları sıralar. Burada liste elemanlarının string, int vb. veri tiplerine uygun olarak sıralanacağı unutulmamalıdır.

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.sort()
print(donanim)
```

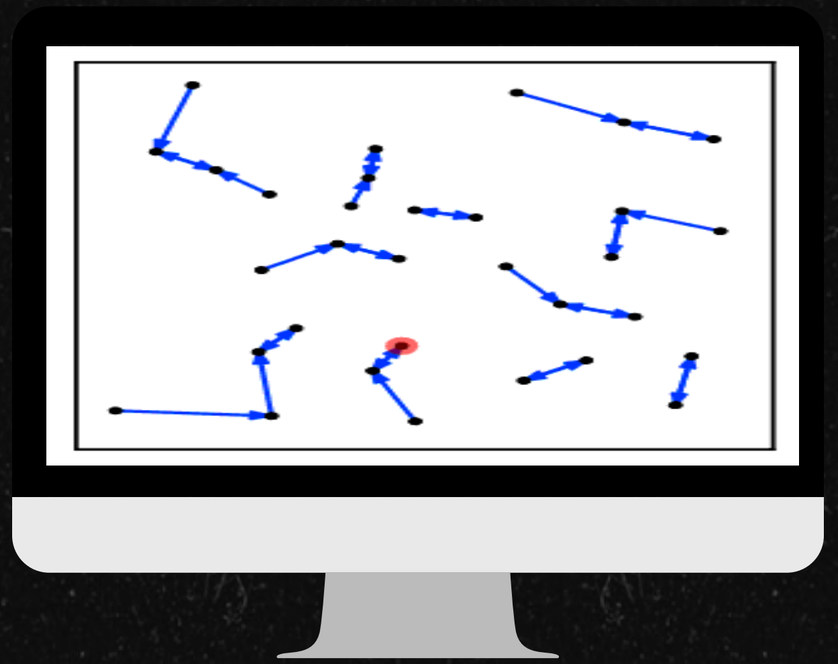
Çıktı: [‘bellek’, ‘işlemci’, ‘klavye’, ‘sabit disk’, ‘yazıcı’]

{06} Listeler – Reverse Fonksiyonu

Reverse: Listeyi sondan başa doğru yani ters yazar.

```
donanim=[“yazıcı”, “klavye”, “işlemci”, “bellek”, “sabit disk”]
donanim.reverse()
print(donanim)
```

Ekran Çıktısı: [‘sabit disk’, ‘bellek’, ‘işlemci’, ‘klavye’, ‘yazıcı’]



SON