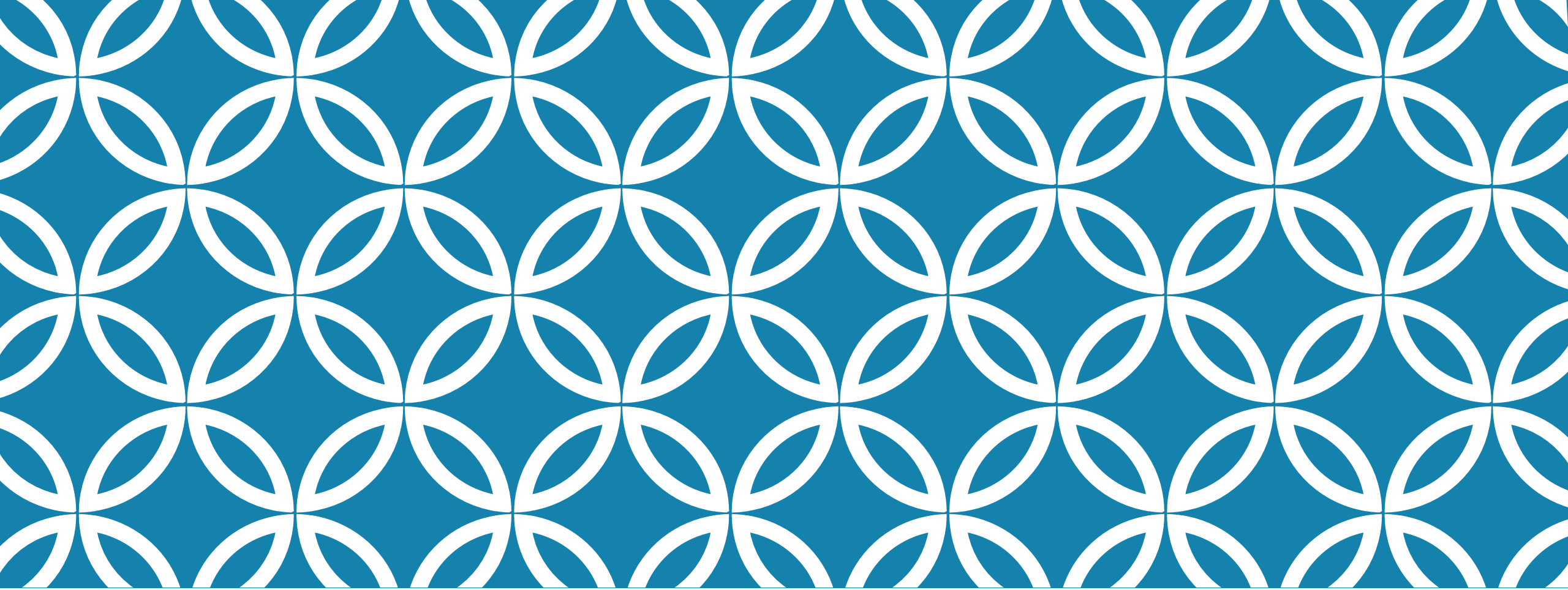

PROGRAMLAMA TEMELLERİ

Öğr. Gör. Erhan AKAGÜNDÜZ

TARİH VE METİN İŞLEMLERİ

- Python dilinde zaman bilgisi tutmak için kendine ait bir veri tipi bulunmaz.
- Python zaman ve tarih bilgilerini bir değişkene atanmış veri değil de oluşturulmuş bir nesne olarak görüp işler.
- Bu nedenle zaman nesnesi ile çalışmak için programlama dili ile birlikte gelen **datetime** modülünü kullanmak gerekir.
- **datetime** modülü; zaman, saat ve tarihlerle ilgili işlemler için çeşitli fonksiyonlar ve özellikler sağlayan sınıfları içerir.

TARİH VE METİN İŞLEMLERİ

- **datetime.date:** Tarihle ilgili nitelikleri ve fonksiyonları barındıran sınıftır. **year (yıl), month (ay)** ve **day (gün)** özelliklerini içerir.
- **datetime.time:** Zamanla ilgili nitelikleri ve fonksiyonları barındıran sınıftır. **hour (saat), minute (dakika), second (saniye) , microsecond (mikrosaniye)** ve **tzinfo (saat dilimi)** özelliklerini içerir.
- **datetime.datetime:** date ve time sınıflarının birleşiminden ve ilave birkaç fonksiyondan oluşur. Örneklerde sıklıkla kullanılacak sınıf budur.
- **datetime.timedelta:** İki date, time veya datetime nesnesi arasındaki zaman farkını mikrosaniye cinsinden veren sınıftır.
- **datetime.tzinfo:** date ve time sınıflarının saat dilimi özelliklerini tutmak için oluşturulmuş abstract (temel) sınıftır.

TARİH VE METİN İŞLEMLERİ

- **now( ) :** datetime modülü içindeki datetime sınıfına ait bu fonksiyon içinde bulunulan andaki tarih ve saat bilgilerini verir.

```
from datetime import datetime #datetime sınıfı içe aktarılıyor.
an = datetime.now() #tarih nesnesi oluşturulup now() fonksiyonu ile zaman bilgisi atanıyor.
print("tarih ve saat = ",an) #an nesnesi ekrana yazdırılıyor.
```

```
tarih ve saat =  2024-09-03 18:28:56.055615
```

TARİH VE METİN İŞLEMLERİ

- **today( ) : now()** fonksiyonu ile aynı işleve sahiptir. Bulunulan günün tarih ve saat bilgilerini verir. Eğer datetime sınıfı ile değil de date sınıfı ile beraber kullanılırsa sadece yıl, ay ve gün bilgisini verecektir. now() fonksiyonu saat bilgisi içerdiğinden date sınıfı ile beraber kullanılmaz.

```
from datetime import datetime #datetime sınıfı içe aktarılıyor.
bugun = datetime.today()
print("bugun = ",bugun)
print(bugun.weekday()) #haftanın kaçıncı günü - Pazartesi 0 ile Pazar 6 arası
print(bugun.year) # yıl bilgisi
print(bugun.month) # ay bilgisi
print(bugun.day) # gün bilgisi
print(bugun.hour) # saat bilgisi
print(bugun.minute) # dakika bilgisi
print(bugun.second) # saniye bilgisi
```

TARİH BİLGİSİNİN BİÇİMLENDİRİLMESİ

- **strftime() :** strftime() fonksiyonu date, datetime veya time nesnesini kullanarak bize tarih ve zamanı bildiren biçimlendirilmiş *string* bir değer verir. Böylece size tarih ve zaman bilgilerini ihtiyaçlarınız doğrultusunda biçimlendirme imkanı sunar.

```
import datetime
an = datetime.datetime.now()
print(an.strftime('%Y')) # Yıl
print(an.strftime('%X')) # Saat
print(an.strftime('%d')) # Gün - ayın kaçıncı günü
print(an.strftime('%A')) # Gün - İsim olarak
print(an.strftime('%B')) # Ay - İsim olarak
```

```
2024
18:40:01
03
Tuesday
September
```

TARİH BİLGİSİNİN BİÇİMLENDİRİLMESİ

```
import datetime
import locale
locale.setlocale(locale.LC_ALL, 'tr_TR.UTF-8')
an = datetime.datetime.now()
print(an.strftime('%Y')) # Yıl
print(an.strftime('%X')) # Saat
print(an.strftime('%d')) # Gün - ayın kaçıncı günü
print(an.strftime('%A')) # Gün - İsim olarak
print(an.strftime('%B')) # Ay - İsim olarak
print(an.strftime('%d %B %Y')) # gün ay ismi yıl şeklinde
print(an.strftime('%d.%m.%Y tarihinde buluşalım.')) # aralarına nokta konarak
```

```
2024
18:42:57
03
Salı
Eylül
03 Eylül 2024
03.09.2024 tarihinde buluşalım.
```

STRİNG (METİN) İŞLEMLERİ

- Python’da stringler karakterleri temsil eden baytları içeren listelerdir.
- Python doğrudan karakterleri temsil eden bir veri tipi içermez.
- Bu nedenle tek bir karakteri temsil eden veri, tek elemanlı bir string dizisidir.
- String verileri kolayca düzenlemek ve değiştirmek için dil içinde gelen gömülü fonksiyonlar vardır.
- Python’da string verilerle nasıl çalışıldığı örnekler eşliğinde incelenebilir.

STRİNG VERİLERİ BİRLEŞTİRME

- Python’da stringler karakterleri temsil eden baytları içeren listelerdir.
- Python doğrudan karakterleri temsil eden bir veri tipi içermez.
- Bu nedenle tek bir karakteri temsil eden veri, tek elemanlı bir string dizisidir.
- String verileri kolayca düzenlemek ve değiştirmek için dil içinde gelen gömülü fonksiyonlar vardır.
- Python’da string verilerle nasıl çalışıldığı örnekler eşliğinde incelenebilir.

```
ad = "Cüneyt"
soyad = "Arkın"
ara=" "
ad_soyad = ad + ara + soyad
print(ad_soyad)
```

```
Cüneyt Arkın
```

STRİNG VERİ İÇİNDEKİ BİR KARAKTERE ERİŞME

- Stringler, karakterlerden oluşmuş listeler olduğundan köşeli parantez ([ ]) ve index sayısı ile istenilen karaktere rahatça erişebilir.

```
sehir = "Gaziantep"
karakter = sehir[3] #dördüncü harfi alıyoruz
print(karakter) #i
print(sehir[0]) #G
print(sehir[5]) #n
```

```
i
G
n
```

STRİNG VERİNİN UZUNLUĞU

- Daha önce listelerde kullanılan len() fonksiyonu ile string verinin uzunluğunu öğrenebilirsiniz.

```
sehir = "Ankara"
print("sehir değişken boyutu: ", len(sehir)) #sehir değişkeninin boyutu
boyut = len(sehir)
son_karakter=sehir[boyut-1] # değişkendeki son karaktere erişilir.
print("Son karakter: ", son_karakter)
for harf in sehir: #stringdeki bütün harflere for döngüsü ile erişilir.
    print(harf)
```

```
sehir değişken boyutu:  6
Son karakter:  a
A
n
k
a
r
a
```

STRİNG VERİYİ PARÇALAMA (SLİCE) VE BÖLME (SPLİT)

- **slice()** fonksiyonu ile köşeli parantezleri kullanarak metinden parçalar alınabilir.
- Bunun için köşeli parantez içine çekilmek istenilen karakter aralığının indisini girmek yeterlidir.
- **Degisken[baslangicIndex : bitisIndex]** şeklinde kullanılır.
- Burada, **baslangicIndex** dâhil, **bitisIndex** dâhil değildir.

STRİNG VERİYİ PARÇALAMA (SLİCE) VE BÖLME (SPLİT)

- **slice()** fonksiyonu

```
veri = "Bilişim Teknolojileri"
print(veri)
print(veri[2:8]) # 2.index dahil, 8.index dahil değildir.
print(veri[2:]) # 2.indexten itibaren stringi bütünüyle alır.
print(veri[5:15]) #5.index dâhil, 15.index dâhil değildir.
print(veri[:12]) # 0. indexten 12. index'e kadar string'i parçalayacaktır.
```

```
Bilişim Teknolojileri
lişim
lişim Teknolojileri
im Teknolo
Bilişim Tekn
```

STRİNG VERİYİ PARÇALAMA (SLİCE) VE BÖLME (SPLİT)

- **split()** fonksiyonu ile, metin verisi belirlenen karakterler baz alınarak bölünebilir. Bölünen metin parçaları dizi hâlinde verilecektir.

```
veri = "Bilişim Teknolojileri"
metin="ilkbahar,yaz,sonbahar,kış"
veri_bolum = veri.split(" ") #veri değişkenini boşluktan itibaren bölüyoruz.
metin_bolum = metin.split(",") #metin değişkenini virgüllerden bölüyoruz.
metin_bolum_ikieleman = metin.split(",",1) #1 parametresi vererek metni sadece 2 parçaya bölüyoruz
print(veri_bolum)
print(metin_bolum)
print(metin_bolum_ikieleman)
```

```
['Bilişim', 'Teknolojileri']
['ilkbahar', 'yaz', 'sonbahar', 'kış']
['ilkbahar', 'yaz,sonbahar,kış']
```

STRİNG VERİ İÇİNDE KARAKTER DEĞİŞTİRME, KARAKTER EKLEME VE ÇIKARMA

- **replace()** fonksiyonu ile bir string içerisindeki herhangi bir karakter değiştirilebilir.

```
kelime = "Bilişim"
cumle = "Merhaba Dünya"
print(kelime)
print(kelime.replace("i","o")) #kelimedeki i'leri o karakteri ile değiştirelim.
print(cumle)
print(cumle.replace("Merhaba","Selam")) #cümledeki "Merhaba"'yı "Selam" ile değiştirelim.
```

```
Bilişim
Boloşom
Merhaba Dünya
Selam Dünya
```

STRİNG VERİ İÇİNDE KARAKTER DEĞİŞTİRME, KARAKTER EKLEME VE ÇIKARMA

- **strip()** fonksiyonu ile bir string içinden karakter çıkarılabilir.

```
cumle = " bilişim teknolojilerine giriş "# baştaki ve sondaki boşluklar silinir.
print(cumle.strip()) # <bosluk>,b,i,l karakterleri silinir.
print(cumle.strip(" bil"))
cumle2 = 'python çok kullanışlı'
print(cumle2.strip("pyt"))
```

```
bilişim teknolojilerine giriş
şim teknolojilerine giriş
hon çok kullanışlı
```

STRİNG VERİ İÇİNDE KARAKTER DEĞİŞTİRME, KARAKTER EKLEME VE ÇIKARMA

- **join()** fonksiyonu ile bir string verinin içerdiği her bir karakterden sonra yeni bir karakter eklenebilir.
- Formatı= "eklemek istenilen string ya da karakter değer".join(değişkenin kendisi = elimizde olan string veri)

```
kelime = "Gaziantep"
print(".".join(kelime))
```

```
G.a.z.i.a.n.t.e.p
```

STRİNG VERİ İLE BÜYÜK VE KÜÇÜK HARF DEĞİŞİMİ YAPMA

- **upper() :** Karakter dizisindeki bütün harfleri büyütür.
- **lower() :** Karakter dizisindeki bütün harfleri küçültür.
- **capitalize() :** Karakter dizisinin ilk harfini büyütür.
- **title() :** Karakter dizisindeki her kelimenin ilk harfini büyütür.
- **swapcase() :** Karakter dizisindeki büyük harfleri küçük, küçük harfleri büyük hâle getirir.

STRİNG VERİ İLE BÜYÜK VE KÜÇÜK HARF DEĞİŞİMİ YAPMA

```
kelime = 'Bilişim teknolojileri'
print(kelime.upper()) # metni büyük harfe çevirir.
print(kelime.lower()) # metni küçük harfe çevirir.
print(kelime.capitalize()) # dizinin harfini büyütür.
print(kelime.title()) # kelimelerin ilk harflerini büyütür.
print(kelime.swapcase()) # büyük küçük değişimi yapar.
```

```
BILIŞIM TEKNOLOJILERI
bilişim teknolojileri
Bilişim teknolojileri
Bilişim Teknolojileri
bILIŞIM TEKNOLOJILERI
```