PYTHON PROGRAMLAMA DİLİ

DOSYALAMA

Dr. Öğr. Üyesi M. Ozan AKI

DOSYALAMA

İşletim sistemlerinde içerisinde veri saklanan yapılara Dosya (**File**) adı verilir.

Bilgisayardaki kayıtlı dosyalarla ilgili, dosya **oluşturma** ve içine veri **yazma**, mevcut dosyadan veri dosya **okuma**, dosya içindeki verileri **değiştirme** ya da silme, dosyayı **kopyalama**, **taşıma** ya da dosyayı **silme** gibi işlemlere kısaca **dosyalama** işlemleri denir.

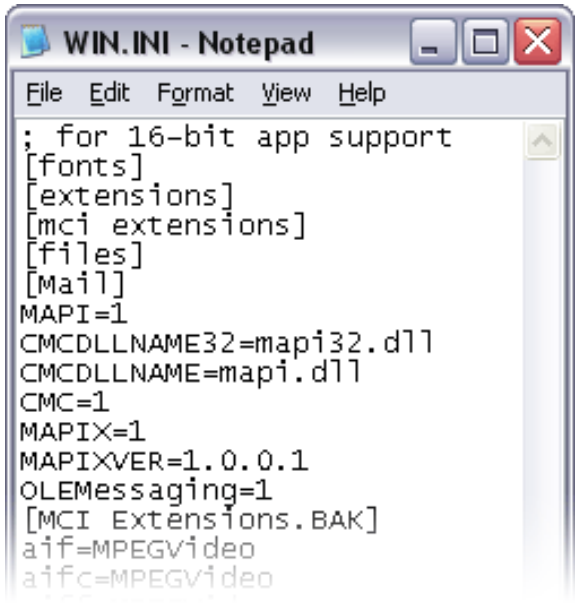
Dosyalama işlemleri, işletim sistemi aracılığı ile yapılır ve işletim sisteminin dosya yönetim yapısı kullanılır.

Dosyalar, klasörler (**Directory**, **Folder**) altında hiyerarşik bir yapıda saklanır.

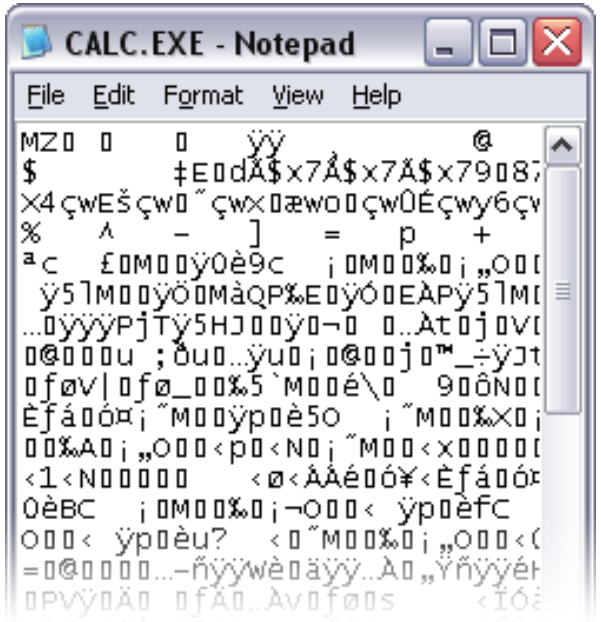
Bir dosyanın, bulunduğu klasörü adı ile birlikte ifade etmeye, dosya yolu (**Path**) olarak isimlendirilir.

DOSYALAMA

Metin (Text) dosyalar: İçinde metin (**text**) bilgi olan, herhangi editörle (notepad - not defteri) açıp görülebilen, değiştirilebilen, kaydedilebilen dosyalardır..



İkili (Binary) dosyalar: İçinde ikili (**binary**) kodlardan oluşan bilgiler bulunan dosyalardır. Bu dosyaların içeriği metin editörüyle açılsa bile anlamsız ve bir şey ifade etmeyen karakterler görünür.



DOSYALAMA - open

Dosya açmak için

degisken=**open(**“dosya_adı”, “mod”**)**

fonksiyonu kullanılır.

İlk parametre olarak dosya adı girilir. İkinci parametre ile, dosyaya be şekilde erişmek istediğimiz belirtilir.

Dönüş değeri bir değişkene aktarılır. Bu değişken artık dosya ile ilgili her türlü işlemi yapacağımız bir nesnedir.

Dosya açılamazsa **None** döner.

Dosya Erişim Modları:

“w”: (**Write**) Dosyaya veri yazmak için açar.
(dosya yoksa yeni oluşturur)

“a”: (**Append**) Dosyaya veri eklemek için açar
(dosya yoksa yeni oluşturur)

“x”: (**Create**) Yeni dosya oluşturur
(dosya zaten varsa hata verir)

“r”: (**Read**) Dosyadan veri okumak için açar
(dosya yoksa hata verir)

“…+”: Hem yazma hem okuma yapmayı sağlar

“…b”: (**Binary**) Dosyayı ikilik modda açar.

DOSYALAMA - open

Metin dosyalarının kaydedilirken hangi kodlama sistemine göre kaydedildiği, üzerinde işlem yaparken önemlidir.

Aksi halde uygun olmayan kodlama sisteminde okuma yapmak, anlamsız karakterlerin elde edilmesine neden olur.

Bunun için gerektiğinde, **open()** fonksiyonunun yanında;

encoding

Parametresi belirtilir.

Dosya Kodlama Sistemleri (Encoding):

“ANSI”: Standard kodlama sistemi. İçinde dillerin özel karakterleri (Türkçe’deki ç,ş,ö,ü,ğ,ü gibi) bulunmaz.

“utf-8”: İçerisinde tüm dünya dillerinin tüm özel karakterinin bulunduğu kodlama sistemdir.

Not: Kodlama sistemleri, işletim sisteminin diline göre değişebilir. Farklı bir ülkede yaşayan birinin kaydettiği dosya, kendi dilinin kodlama sisteminde kaydedilmiş olabilir.

DOSYALAMA - open

```
dosya=open("dosya.txt","r")
```

(metin dosyasını okumak için açar)

```
dosya=open("dosya.txt","r", encoding="utf-8")
```

(metin dosyasını utf-8 kodlaması ile okumak için açar)

```
dosya=open("dosya.txt","a+", encoding="ANSI")
```

(metin dosyasına ANSI kodlamasına göre veri eklemek için açar)

```
dosya=open("dosya.txt","r+")
```

(metin dosyasını hem okumak hem yazmak için açar. Dosya yoksa hata verir)

```
dosya=open("dosya.txt","w+")
```

(metin dosyasını hem okumak hem yazmak için açar. Dosya yoksa yeni açar)

```
dosya=open("dosya.txt","r+b")
```

(ikili dosyayı hem okumak hem yazmak için açar. Dosya yoksa hata verir)

```
dosya=open("dosya.txt","w+b")
```

(ikili dosyayı hem okumak hem yazmak için açar. Dosya yoksa yeni açar)

DOSYALAMA - write

Daha önce yazma modunda başarıyla açılmış dosyanın Dosya içine veri yazmak için;

dosya_değişkeni.**write(**“yazılacak veri”**)**

```
dosya=open(“dosya.txt”,”w”)
dosya.write(“yazılacak veri”)
```

```
dosya=open(“dosya.txt”,”a”)
dosya.write(“yazılacak veri”)
```

DOSYALAMA - writelines

Eğer bir dosyaya çok sayıda satırı bir seferde yazdırmak istersek;

dosya_değişkeni**.writelines(**liste**)**

Metodunu kullanabiliriz.

Bu fonksiyona, her bir elemanı satırlardan oluşan bir liste vermemiz gerekir.

Listenin her bir elelmanı bir satır olarak dosyaya yazılır.

```
liste=[“bir”,”iki”,”uc”]


dosya=open(“dosya.txt”,”w”)
dosya.writelines(liste)
```

DOSYALAMA - read

Varolan bir dosyadaki tüm veriyi okuyarak bir değişkene atamak için kullanılır.

veriler=dosya_değişkeni.**read()**

Eğer belirli sayıda karakter okunmak istenirse, read() fonksiyonuna karakter sayısı girilebilir.

Örneğin, dosyadan 10 karakter (byte) okumak için;

veriler=dosya_değişkeni.**read(10)**

```
dosya=open("dosya.txt","r")
veriler=dosya.read()
```

DOSYALAMA - readline

Varolan bir dosyadan bir seferinde bir satır okumak için kullanılır.

satir=dosya_değişkeni.**readline()**

Eğer belirli sayıda karaker okunmak istenirse, readline() fonksiyonuna karakter sayısı girilebilir.

Örneğin, satırdan 10 karakter okumak için;

satir=dosya_değişkeni.**readline(10)**

```
dosya=open("dosya.txt","r")
satir=dosya.readline()
```

DOSYALAMA - readlines

Varolan bir dosyadan bir her bir satısı ayrı ayrı bir liste elemanı olarak okumak için kullanılır.

satir=dosya_değişkeni.**readlines()**

Eğer belirli sayıda satır okunmak istenirse, readlines() fonksiyonuna satır sayısı girilebilir.

Örneğin, dosyadan 10 satır okumak için;

satirlar=dosya_değişkeni.**readlines(10)**

```
dosya=open("dosya.txt","r")
satirlar=dosya.readlines()
```

DOSYALAMA - close

Açılmış olan her dosyanın, işi bittiğinde kapatılması gerekir.

Açık olan bir dosyayı kapatmak için;

dosya_değişkeni.**close()**

Metodu kullanılır.

```
dosya=open("dosya.txt","r")
veriler=dosya.read()
dosya.close()
```

DOSYALAMA – for-in

for döngüsü, **in** ifadesi ile birlikte bir dosya değişkeni üzerinde kullanılırsa, dosyanın her bir satırı döngü değişkenine atanarak dosya sonuna kadar her bir satır okunur.

```
dosya=open("dosya.txt","r")
for satir in dosya:
  print(satir, end='')
dosya.close()
```

DOSYALAMA – with-as

Bir dosyanın açılıp, çeşitli işlemler yapıldıktan sonra kapatılması gerekliliği bazen dosyanın kapatılmasının unutulması nedeniyle sorunlara yol açabilmektedir.

Bu gibi durumların önüne geçmek için;

with

İfadesi kullanılır. Tüm dosya işlemleri **with** kapsamında yapılır ve **with** bloğu sonlandığında dosya otomatik olarak kapatılır.

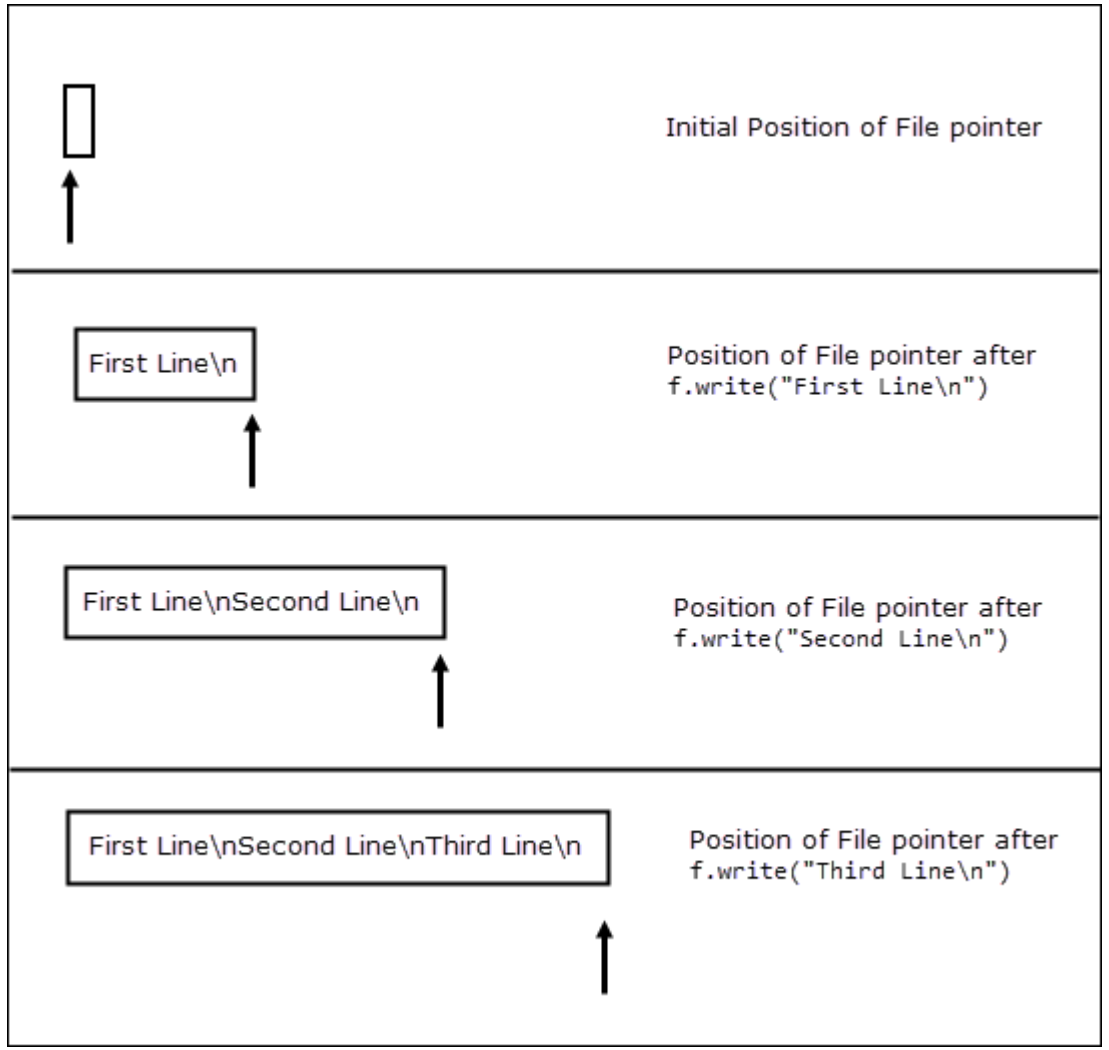
```
with open("dosya.txt","r") as dosya:
    for satir in dosya:
        print(satir, end='')
```

DOSYALAMA – cursor (imleç)

Dosyalarda işlem yaparken, tıpkı terminal ekranında yanıp sönen imleç (**cursor**) gibi, o anda hangi konumda işlem yaptığımızı gösteren bir imleç (**cursor**) bulunur.

Bu imleç, okuma ya da yazma işlemi yaptıkça otomatik olarak ilerler. (tıpkı terminal ekranındaki imleç gibi)

Bazen dosyadaki bu imlecin konumunu yeniden ayarlamamız ya da yerini öğrenmemiz gerekebilir.



DOSYALAMA – tell

Dosyada, imlecin mevcut konumunu öğrenmek için;

dosya_değişkeni.**tell()**

Metodu kullanılır.

Dönen bilgi **byte** cinsinden konum bilgisidir.

```
with open("dosya.txt","r") as dosya:
    once=dosya.tell()
    print(once)
    dosya.read(10)
    sonra= dosya.tell()
    print(sonra)
```

DOSYALAMA – seek

Dosyada, imlecin konumunu değiştirmek için;

dosya_değişkeni.**seek(x)**

Metodu kullanılır. Dosya imleci, x olarak verilen konuma gider.

Konum bilgisi **byte** cinsinden belirtilir.

```
with open("dosya.txt","r") as dosya:
    veri1=dosya.read(10)
    dosya.seek(20)
    veri2=dosya.read(10)
```

DOSYALAMA – metotlar

Dosya nesnesinde tanımlı bazı metotlar;

file=open(…)

file.name: Dosyanın adı. (**open()** ile verdiğimiz ilk parametredir)

file.mode: Dosyanın açma modu

file.encoding: Dosyanın kodlama sistemi

file.closed: Dosya kapatılmış mı? (kapatıldıysa True, kapatılmadıysa False)

file.readable(): Dosya okunabilir mi? (okunabilir: True, okunamaz: False)

file.writeable(): Dosya yazılabilir mi? (yazılabilir: True, yazılamaz: False)

DOSYALAMA – os

Dosyalarla uğraşırken çoğu zaman dosya silme ve klasör (**directory**, **folder**) yolları (**path**) ile uğraşmamız gerekecektir.

Dosyalar ile ilgili işlemler için **os** modülü ve klasör yolları ile ilgili işlemler için **os** modülün altındaki **path** nesnesi kullanılır.

```
import os
```

```
from os import path
```

DOSYALAMA – os

os.getcwd(): O anda geçerli olan klasörü döndürür.

os.chdir(d): geçerli klasörü verilen d klasörü olarak değiştirir.

os.listdir(d): Verilen d klasöründeki dosya ve alt klasör listesini döndürür

os.remove(f): Verilen f dosyasını siler.

os.mkdir(d): Verilen d isminde yeni bir klasör oluşturur.

os.rmdir(d): Verilen d klasörünü siler. (klasör boş olmalıdır)

os.removedirs(d): Verilen d klasörü ve tüm alt klasörleri siler.

os.rename(f1,f2): Verilen f1 isimli dosyasını adını f2 olarak değiştirir.

DOSYALAMA – os.path

os.path.exists(f): Verilen f dosyasının mevcut olup olmadığını döndürür.

os.path.join(d1,d2,d3,…): Ayrı ayrı verilen klasör yolu parçalarını birleştirir.

os.path.dirname(f): Verilen f dosyasının sadece klasör yolunu döndürür.

os.path.abspath(f): Verilen f dosyasının mutla yolunu döndürür.

os.path.split(f): Verilen f yolunun tüm parçalarını ayrı ayrı döndürür.

os.path.splitdrive(f): Verilen f yolunun sürücü ve yolunu ayrı ayrı döndürür.

os.path.splitext(f): Verilen f dosyasının klasör yolunu ve dosya uzantısını ayırır

os.path.isfile(f): Verilen f adının bir dosya olup olmadığın döndürür.

os.path.isdir(f): Verilen f adını bir klasöre olup olmadığını döndürür.

DOSYALAMA – shutil

Dosya kopyalamak, klasör silmek gibi daha üst düzey işlemler için shutil modülü kullanılır.

shutil.copy(f1,f2): Verilen f1 dosyasın f2 ile belirtilen yere kopyalar.

shutil.move(f1,f2): Verilen f1 dosya f2 konumuna taşır.

shutil.copytree(d1,d2): Verilen d1 klasörünü, tüm alt klasör ve dosyaları ile birlikte verilen d2 hedefine kopyalar

shutil.rmtree(d): Verilen s klasörünü, tüm alt klasör ve dosyalarıyla birlikte siler.