**Dosya İşlemleri :**

Bazı durumlarda daha önce bilgisayara girilen ya da bilgisayardan elde edilen verileri saklamak ve onlar üzerinde işlem yapmak gerekir. Herhangi bir kurum stok takibi, müşteri takibi, muhasebe işlemleri gibi çok sayıda kişi ve malzeme kayıtlarının bilgisayara girilmesi ve bu kayıtlar üzerinde işlem yapılması gerekebilir.

Program ve veriler daha sonra kullanılabilmek için kalıcı olarak bilgisayarın disk/hardisk, flash bellek gibi depolama birimlerinde saklamak gerekir. Bu depolama birimlerinde saklanan veri ve program gruplarına dosya(file) adı verilir.

Klasik dosyalama yöntemleri;

**a)** Metin (Text) dosyaları

**b)** İkili Dosyalar (Binary Files)

**Metin (Text) Dosyalar ;** genellikle metin editörü (notepad gibi) ile oluşturulan programlardır. Bir veya daha fazla metin karakterleri içeren dosyalardır. Metin dosyalarında her bir satır; **satır sonu belirteci (\n veya\r gibi) ile sonlanır.** Metin dosyalarının uzantısı genelde "\*.txt, \*.dat, \*.doc, \*.csv, \*.html, \*.xml" şeklinde olmakla birlikte program dosyalarında "\*.py gibi" içerik olarak birer metin dosyalarıdır. Derlenmiş bir Python dosyası ise binary formatta bir dosyadır.

**İkili (Binary) Dosyalar ;** sadece metin (text) bilgisi içermeyen, resim ses veya video bilgileri de içerebilen, dosyaya erişimin byte (karakter) bazında olduğu dosyalardır. Genelde metin dosyası olmayan bütün dosyalar binary olarak kabul edilir. (.exe, .gif, .jpg, \*.zip, .hex gibi).

Genelde ikili dosyalama sayı saklamada ve birden fazla dosyayı birleştirme veya parçalama uygulamalarında kullanılır.

İkili dosyaların içeriği ; byte dizileri halinde bir hex editör programı yardımıyla insanların okuyabileceği formata dönüştürülür.

**Temel Dosya İşlemleri :**

Dosyalarla ilgili yapılan işlemlerde aşağıdaki işlemler takip edilir ;

**1.** Dosyanın açılması (open komutu ile)

**2.** Dosya ile ilgili işlem yapılması (dosyadaki bir veriyi okuma, dosyaya yazma, dosyadaki bir veri düzeltme/güncelleme veya silme gibi)

**3.** Dosyanın kapatılması (close komutu ile)

Python dilinde dosya açma/kapama için iki farklı yapı kullanılır.

**Klasik Yapı ;**

f=open("deneme.txt","w")

f.write("mesaj yazıldı\n")

f.close()

**with...as Yapısı ;**

with open("deneme.txt","w") as f:

f.write("mesaj yazıldı\n")

Bu yapıda dosya otomatikman kapatılır.

Bir dosyayı kullanmadan önce açmamız gerekir. Dosyayı açarken de genelde açma amacınızı (modumuzu) belirtme zorunluluğu vardır.

**"r"--"rb"=** Dosyayı **okuma** (read) amaçlı açar. Okunacak dosya mevcut değilse hata mesajı verir. **Dosya açma modu belirtilmezse varsayılan olarak bu modda dosya açılır.**

**"w"--"rw" =** Dosyayı **yazma** (write) amaçlı açar. Eğer açılan dosya isminde başka bir dosya varsa o dosyanın içeriği silinir, yoksa ilk defa oluşturulur.

**"a"--"ab" =** Dosyayı **ekleme** (append) amaçlı açar . Bir dosya hem **okuma** hem de **yazma** amaçlı kullanılacak ise bu modda açılabilir. Var olan bir dosya bu modda açıldı ise yeni bilgiler dosya **sonuna** eklenir.

**"x"--"xb"** = **"w" ile aynı işleve**(write) sahiptir. Farkı ; Eğer açılan dosya isminde başka bir dosya varsa o doyanının **içeriğini silmez,** onun yerine **hata mesajı verir.**

**"r+"--"rb+"** = Dosayı hem **okuma** hem de **yazma** amaçlı açar. Ancak dosyanın önceden oluşturulmuş olması gerekir.

**"w+"--"wb+"** = Dosayı hem **okuma** hem de **yazma** amaçlı açar. Ancak dosya daha önceden oluşturulmuş ise dosya içerği silinir, yoksa yeniden oluşturulur.

**"a+"--"ab+"** = Dosya hem **okuma** hem de **yazma** amaçlı açılır. Dosya daha önceden oluşturulmadı ise yeniden oluşturulur.

**"x+"--"xb+"** = Dosya hem **okuma** hem de **yazma** amaçlı açılır. Ancak dosyanın isminde başka bir dosya varsa o dosyanın içeriği silinmez, onun yerine hata mesajı verir.

Dosya ile ilgili işlemlerde dikkat edilmesi gerekenler ;

**a.** Açılan dosya işlem bittikten sonra mutlaka kapatılmalı.

**b.** Dosya hangi amaçla açıldı ise o amaca uygun işlemelr gereçekleştirilir.

**c.** Disk ya da diskette daha önce kullanılmış bir isimle dosya açılmaya çalışılırsa hata oluşur.

**d.** Eğer açık dosya tekrardan açılmaya çalışılırsa veya kapalı dosay tekrardan kapatılmaya çalışılırsa hata mesajı alınır.

**e.** Eğer var olmayan bir dosayı okumak için açmaya veya kapatmaya çalışırsak hata mesajı alırız.

**Klasik dosya işlemlerinde kullanılan Python komutları ;**

**open() =** Dosyayı açmak için kullanılır.

**write()** = Dosyaya istenen veriyi yazar. Eğer satır satır yazdırmak istenirse metin içerisinde **"\n"** karakteri kullanılır.

**writelines()** = Dosyaya verilen listeyi (**birden fazla satırlık veriyi**) yazar.

**read(n)** = Dosyadan **n karakterlik** veri okur.

**read()** = Tüm dosyayı okur sonucu **string** olarak döndürür.

**readline()** = Dosyadan satır satır veri okur.

**readlines()** = Tüm dosyayı okur, sonucu **liste** olarak döndürür.

**seek()** = Dosya işaretcisinin belirttiği bir kayıta gider.

**tell()** = Dosya işaretcisinin pozisyonunu (dosya başına göre) verir.

**close()** = Dosyayı kapatır.

**mode** = Dosyanın açılış amacını (modunu) döndürür.

**closed** =Dosyanın açık/kapalı olma durumunu döndürür.

**name** = Dosyanın adını döndürür.

**CSV Uzantılı Dosyaları Okuma ve Yazma :**

Csv uzantılı dosyalar; veri tabanı kullanıcıları için verilerin virgüller ile ayrıdlığı, belli bir düzende yazılıp, kaydedildiği özel dosylardır. import csv ile import edilir.

Exel veya LibreOffice Calc benzeri hesap tablosunu CSV formatında kaydedip, tablo içerğini Python ile okuyabilir üzerinde işlem yapabilir ve CSV formatında tekrar kaydedebiliriz.

Python CSV modülü; bir csv dosyasındadn veri okurken reader(), dosyaya veri/kayıt yazarken ise writer() sınıflarına ait fonksiyonları kullanır. Ayrıca DictReader ve DictWriter sınıflarını kullanarak sözlük yapsınında veri okuyabilir yazabilir.

csv.writerows(kayitlar) = Tüm kayıtları csv dosyasına yazar.

csv.writerow(kayit) = geçerli kayıdı csv dosayasına yazar.

CSV dosyalrında veri analizi pandas ile yapılır.

**pickle Modülü ile İkili/binary Dosyalara Veri Yazma/Okuma :**

pickle modülü;bir binary dosya veya byte dizisini okuma veya oluşturmak için kullanılır ve kendine özgü metotlara sahiptir. Örneğin bir binary dosyaya veri yazmak için dump(), veri okumak için load() metotlarına sahiptir.

os Modülü ile Dosya-Dizin(Klasör) İşlemleri :

import os ile import edilir.

os.getcwd() = Geçerli dizinin (current working dictionary) yolu ile verir.

os.chdir() = Geçerli dizini değiştirir.

os.listdir() = Geçerli dizinin altında yer alan dosyaların listesini verir.

os.remove() = Belirtilen dosayayı siler.

os.mkdir() = Klasör oluşturur.

os.rename() = Dosya veya klasörün ismini değiştirir. os.rename(Dosya,YeniDosya)

os.path.islife() = Dosya ise true değerini döndürür.

os.path.isdir() = Klasör ise true değerini döndürür.

os.path.exists() = Dosya ya da Klasör mevcut ise true değerini döndürür.

os.path.split() = Dosya adı ile yolunu ayırır.

os.path.splitext() = Dosyanın uzantısnını ayırır.

**Bir Web Sayfasından Veri Okumak:**

Python'da dosya okuma rahatlığında bir web adresinden (URL) açıp, o Web adresinden veri çekilebilir ya da içerği okunabilir. Bunun için urlopen() fonksiyonu kullanılır.

Örneğin;

import urllib.request

dosya=urllip.request.urlopen("http://www.bbc.com")