**1. Faylni ochish. Fayllar bilan ishlovchi metodlar**

**Faylni ochish**

Python’da fayl bilan ishlash uchun birinchi navbatda uni **ochish** kerak bo‘ladi. Buning uchun open() funksiyasi ishlatiladi.

**Sintaksis:**

fayl = open("nomi.txt", "rejim")

* "nomi.txt" – fayl nomi (yoki yo‘li)
* "rejim" – ochish rejimi

**Asosiy rejimlar:**

* "r" – o‘qish (read). Fayl mavjud bo‘lishi shart.
* "w" – yozish (write). Agar fayl bo‘lmasa, yangi ochadi; bo‘lsa, eski ma’lumotni o‘chiradi.
* "a" – qo‘shish (append). Ma’lumot oxiriga yoziladi.
* "b" – ikkilik (binary) rejim (masalan, rasm, video, audio fayllar).
* "x" – yangi fayl yaratish (agar fayl mavjud bo‘lsa, xatolik beradi).

**Misol:**

fayl = open("test.txt", "w") # yozish rejimida ochish

fayl.write("Salom, Python!\n")

fayl.close()

**Fayllar bilan ishlovchi metodlar**

**O‘qish metodlari**

* read() – butun faylni satr sifatida o‘qiydi.
* read(n) – n ta belgini o‘qiydi.
* readline() – bitta qatordan o‘qiydi.
* readlines() – barcha qatorlarni ro‘yxat (list) ko‘rinishida qaytaradi.

**Misol:**

fayl = open("test.txt", "r")

print(fayl.read()) # butun faylni o‘qish

fayl.close()

fayl = open("test.txt", "r")

print(fayl.readline()) # 1-qator

print(fayl.readline()) # 2-qator

fayl.close()

fayl = open("test.txt", "r")

print(fayl.readlines()) # ['Salom, Python!\n', 'Ikkinchi qator\n']

fayl.close()

**Yozish metodlari**

* write(matn) – faylga matn yozadi.
* writelines(ro‘yxat) – ro‘yxatdagi elementlarni faylga yozadi.

**Misol:**

fayl = open("new.txt", "w")

fayl.write("Birinchi qator\n")

fayl.write("Ikkinchi qator\n")

fayl.writelines(["Uchinchi qator\n", "To‘rtinchi qator\n"])

fayl.close()

**Qo‘shimcha metodlar**

* close() – faylni yopadi.
* seek(n) – faylda kursorni n-pozitsiyaga o‘tkazadi.
* tell() – kursorning joriy joylashuvini qaytaradi.

**Misol:**

fayl = open("test.txt", "r")

print(fayl.read(5)) # dastlabki 5 ta belgi

print("Kursor joylashuvi:", fayl.tell())

fayl.seek(0) # boshiga qaytarish

print(fayl.readline())

fayl.close()

📌 Xulosa: Python’da faylni ochish uchun open() funksiyasi ishlatiladi. Fayldan ma’lumot o‘qish uchun read(), readline(), readlines() metodlari, yozish uchun write(), writelines() metodlari mavjud. Faylni yopish uchun close() yoki avtomatik yopish uchun with open(...) as fayl: usuli qo‘llaniladi.

**with open() usuli bilan fayllar bilan ishlash**

**Nima uchun with open() kerak?**

Oddiy holatda faylni ochib, ishlatib bo‘lgach uni close() metodi bilan yopish kerak. Agar dasturda xatolik chiqsa, fayl yopilmasligi mumkin. Bu esa resurslar ortiqcha band bo‘lishiga olib keladi.

Shu muammoni hal qilish uchun Python’da **kontekst menejeri** (with open()) mavjud. U faylni avtomatik ochadi va yopadi.

**Sintaksis**

with open("fayl\_nomi.txt", "rejim") as fayl:

# fayl bilan ishlash

...

# fayl avtomatik yopiladi

* fayl\_nomi.txt – fayl nomi
* rejim – ochish rejimi ("r", "w", "a", "b")
* fayl – ochilgan fayl obyektiga murojaat qilish nomi

**Misollar**

**1. Fayldan o‘qish**

with open("test.txt", "r") as fayl:

matn = fayl.read()

print(matn)

# fayl avtomatik yopildi

**2. Faylga yozish**

with open("output.txt", "w") as fayl:

fayl.write("Bu faylga yozildi.\n")

fayl.write("Ikkinchi qator.")

# fayl yopiladi, ma’lumot saqlanadi

**3. Qatorma-qator o‘qish**

with open("test.txt", "r") as fayl:

for qator in fayl:

print(qator.strip())

**4. Bir nechta qatordan ro‘yxat olish**

with open("test.txt", "r") as fayl:

qatorlar = fayl.readlines()

print("Fayldagi qatorlar soni:", len(qatorlar))

**Afzalliklari:**

1. Fayl avtomatik yopiladi (xatolik bo‘lsa ham).
2. Kod ixcham va o‘qilishi oson bo‘ladi.
3. Resurslarni tejaydi.

✅ **Xulosa:**  
with open() — fayllar bilan ishlashda xavfsiz va to‘g‘ri usul hisoblanadi. Faylni yopishni unutishdan saqlaydi va kodni soddalashtiradi.

### ****4. Kataloglar (papkalar) bilan ishlash****

## ****Katalog tushunchasi****

Katalog (yoki papka) – bu fayllar va boshqa papkalarni o‘zida saqlovchi tuzilma. Python’da kataloglar bilan ishlash uchun asosan **os** va **shutil** modullari ishlatiladi.

## ****Asosiy amallar****

### ****1. Joriy katalogni aniqlash****

import os

print("Joriy katalog:", os.getcwd())

➡ Natija: C:\Users\User\Documents

### ****2. Katalog ichidagi fayl va papkalarni ko‘rish****

import os

print(os.listdir(".")) # joriy papkadagi hamma fayl va papkalar

print(os.listdir("C:/Windows")) # boshqa papkadan

### ****3. Yangi katalog yaratish****

import os

os.mkdir("yangi\_papka") # bitta katalog yaratish

os.makedirs("asosiy/sub1") # ichma-ich papkalar yaratish

### ****4. Katalogni o‘chirish****

import os

os.rmdir("yangi\_papka") # bo‘sh papkani o‘chirish

os.removedirs("asosiy/sub1") # ichma-ich bo‘sh papkalarni o‘chirish

⚠️ Agar papka ichida fayllar bo‘lsa, shutil.rmtree() ishlatiladi:

import shutil

shutil.rmtree("asosiy") # papka va ichidagi hamma fayllarni o‘chiradi

### ****5. Fayl va katalog mavjudligini tekshirish****

import os

print(os.path.exists("test.txt")) # True yoki False

print(os.path.isdir("asosiy")) # katalog ekanligini tekshiradi

print(os.path.isfile("test.txt")) # fayl ekanligini tekshiradi

### ****6. Fayl va katalog nomlarini birlashtirish****

Yo‘llarni qo‘lda yozish o‘rniga os.path.join() ishlatiladi. Bu barcha tizimlarda (Windows/Linux/Mac) to‘g‘ri ishlaydi.

import os

path = os.path.join("asosiy", "sub1", "file.txt")

print(path) # asosiy/sub1/file.txt

### ****7. Fayl va katalogni nomini o‘zgartirish****

import os

os.rename("eski.txt", "yangi.txt") # faylni qayta nomlash

os.rename("eski\_papka", "yangi\_papka") # papkani qayta nomlash

### ****8. Faylni boshqa papkaga ko‘chirish yoki nusxalash****

shutil moduli orqali:

import shutil

shutil.move("test.txt", "asosiy/test.txt") # faylni ko‘chirish

shutil.copy("asosiy/test.txt", "nusxa.txt") # fayl nusxa olish

## ****Amaliy misollar****

### Misol 1: Joriy papkada faqat .txt fayllarni chiqarish

import os

for fayl in os.listdir("."):

if fayl.endswith(".txt"):

print(fayl)

### Misol 2: Papka ichidagi barcha fayllarni rekursiv ko‘rish

import os

for root, dirs, files in os.walk("."):

print("Papka:", root)

for fayl in files:

print(" -", fayl)

### Misol 3: Faylning to‘liq yo‘lini olish

import os

print(os.path.abspath("test.txt"))

✅ **Xulosa:**  
Python’da kataloglar bilan ishlash uchun **os** va **shutil** modullari qo‘llaniladi. Ular orqali katalog yaratish, o‘chirish, fayl va papkalarni ko‘chirish yoki nusxalash, katalog tarkibini ko‘rish mumkin.

Siz xohlaysizmi, endi men **“Fayllar va kataloglar bilan ishlash” mavzusi bo‘yicha amaliy mashqlar (topshiriqlar va yechimlari)** tayyorlab beray?

### ****Fayllar va kataloglar bilan ishlash – Amaliy mashqlar****

Quyida mavzuni mustahkamlash uchun **topshiriqlar va ularning yechimlari** berilgan.

## ****1-mashq: Matn yozish va o‘qish****

**Topshiriq:** Foydalanuvchidan ism kiritilsa, uni user.txt fayliga yozing. So‘ng faylni ochib, yozilgan ismni ekranga chiqaring.

**Yechim:**

# Ismni foydalanuvchidan olish

ism = input("Ismingizni kiriting: ")

# Faylga yozish

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

f.write(ism)

# Fayldan o‘qish

with open("user.txt", "r") as f:

print("Fayldan o‘qildi:", f.read())

## ****2-mashq: Qatorlar sonini hisoblash****

**Topshiriq:** data.txt faylida nechta qator borligini aniqlang.

**Yechim:**

with open("data.txt", "r") as f:

qatorlar = f.readlines()

print("Qatorlar soni:", len(qatorlar))

## ****3-mashq: Faqat**** .txt ****fayllarni chiqarish****

**Topshiriq:** Joriy katalogda mavjud bo‘lgan barcha .txt fayllarni ekranga chiqaring.

**Yechim:**

import os

for fayl in os.listdir("."):

if fayl.endswith(".txt"):

print(fayl)

## ****4-mashq: Faylga sonlar yozish va yig‘indisini hisoblash****

**Topshiriq:** 1 dan 10 gacha bo‘lgan sonlarni numbers.txt fayliga yozing. Keyin fayldan o‘qib, yig‘indisini hisoblang.

**Yechim:**

# Faylga yozish

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

for i in range(1, 11):

f.write(str(i) + "\n")

# Fayldan o‘qish va yig‘indisini hisoblash

with open("numbers.txt", "r") as f:

sonlar = f.readlines()

sonlar = [int(x.strip()) for x in sonlar]

print("Yig‘indi:", sum(sonlar))

## ****5-mashq: Faylni nusxalash****

**Topshiriq:** test.txt faylidan nusxa olib, yangi faylni copy.txt deb saqlang.

**Yechim:**

import shutil

shutil.copy("test.txt", "copy.txt")

print("Fayl nusxalandi.")

## ****6-mashq: Papka yaratish va faylni ko‘chirish****

**Topshiriq:** new\_folder nomli papka yarating va test.txt faylini ichiga ko‘chiring.

**Yechim:**

import os, shutil

# Papka yaratish

if not os.path.exists("new\_folder"):

os.mkdir("new\_folder")

# Faylni ko‘chirish

shutil.move("test.txt", "new\_folder/test.txt")

print("Fayl ko‘chirildi.")

## ****7-mashq: Eng uzun qatorni topish****

**Topshiriq:** poem.txt faylidan eng uzun qatorni toping.

**Yechim:**

with open("poem.txt", "r") as f:

qatorlar = f.readlines()

eng\_uzun = max(qatorlar, key=len)

print("Eng uzun qator:", eng\_uzun.strip())

✅ **Xulosa:**  
Bu mashqlar orqali siz faylni ochish, o‘qish, yozish, kataloglar bilan ishlash, nusxalash va ko‘chirish amallarini mustahkamlaysiz.

Sizga ushbu mashqlarni **mustaqil mashq uchun savollar ro‘yxati** tarzida ham tuzib beraymi (yechimsiz)?

**2. os modulining imkoniyatlari**

Python’da **os moduli** operatsion tizim bilan ishlash imkonini beradi. Bu modul yordamida fayl va kataloglar ustida amallar bajarish, tizim ma’lumotlarini olish, dastur ish jarayonini boshqarish mumkin.

**Asosiy imkoniyatlar**

**1. Joriy ishchi katalog bilan ishlash**

import os

print(os.getcwd()) # joriy katalogni olish

os.chdir("C:/Users") # katalogni o‘zgartirish

print(os.getcwd())

**2. Katalog yaratish va o‘chirish**

os.mkdir("yangi\_papka") # bitta papka yaratish

os.makedirs("a/b/c") # ichma-ich papkalar yaratish

os.rmdir("yangi\_papka") # bo‘sh papkani o‘chirish

os.removedirs("a/b/c") # ichma-ich bo‘sh papkalarni o‘chirish

**3. Katalog tarkibini ko‘rish**

print(os.listdir(".")) # joriy katalogdagi fayl va papkalar

print(os.listdir("C:/Windows"))

**4. Fayl va katalog haqida ma’lumot olish**

print(os.path.exists("test.txt")) # mavjudligini tekshirish

print(os.path.isfile("test.txt")) # fayl ekanini aniqlash

print(os.path.isdir("yangi\_papka")) # katalog ekanini aniqlash

**5. Fayl va katalog nomlarini birlashtirish**

path = os.path.join("asosiy", "sub1", "file.txt")

print(path) # asosiy/sub1/file.txt

**6. Fayl va katalogni qayta nomlash yoki ko‘chirish**

os.rename("eski.txt", "yangi.txt") # faylni nomini o‘zgartirish

os.rename("eski\_papka", "yangi\_papka") # papkani nomini o‘zgartirish

**7. Tizim buyruqlarini bajarish**

os.system("dir") # Windowsda katalog tarkibini ko‘rsatadi

os.system("ls") # Linux/MacOS da katalog tarkibini ko‘rsatadi

**8. Muhit o‘zgaruvchilari bilan ishlash**

print(os.environ) # barcha muhit o‘zgaruvchilari

print(os.environ["PATH"]) # PATH qiymatini olish

**9. Fayl va papka yo‘llari bilan ishlash (os.path yordamida)**

import os

print(os.path.abspath("test.txt")) # to‘liq yo‘lni olish

print(os.path.basename("C:/data/test.txt")) # fayl nomi

print(os.path.dirname("C:/data/test.txt")) # katalog nomi

print(os.path.split("C:/data/test.txt")) # ('C:/data', 'test.txt')

print(os.path.splitext("test.txt")) # ('test', '.txt')

**10. Rekursiv yurish (os.walk)**

Katalog ichidagi hamma fayl va papkalarni ko‘rish:

for root, dirs, files in os.walk("."):

print("Papka:", root)

for fayl in files:

print(" -", fayl)

**Xulosa**

os moduli Python’da operatsion tizim bilan ishlash uchun kuchli vosita hisoblanadi.  
U orqali:

* katalog yaratish va o‘chirish,
* fayllar bilan ishlash,
* tizim buyruqlarini bajarish,
* muhit o‘zgaruvchilarini olish,
* yo‘l nomlarini boshqarish mumkin.

Siz xohlaysizmi, men endi **shutil modulining imkoniyatlari** haqida ham batafsil qilib yozib beray?

**3. shutil modulining imkoniyatlari**

**Umumiy tushuncha**

Python’dagi **shutil** moduli fayl va kataloglar bilan **yuqori darajadagi amallar** bajarishga mo‘ljallangan. Bu modul yordamida fayl va kataloglarni **nusxalash, ko‘chirish, o‘chirish, arxivlash** kabi ishlarni qilish mumkin.

**Asosiy imkoniyatlar**

**1. Faylni nusxalash**

import shutil

shutil.copy("test.txt", "nusxa.txt") # faqat faylni nusxalash

shutil.copy2("test.txt", "nusxa2.txt") # faylni meta-ma’lumotlari bilan nusxalash

➡ copy2() funksiyasi faylning yaratilgan vaqti, ruxsatlari kabi atributlarini ham nusxa qiladi.

**2. Faylni ko‘chirish**

shutil.move("test.txt", "yangi\_papka/test.txt")

➡ Faylni boshqa joyga o‘tkazadi yoki qayta nomlaydi.

**3. Fayl va katalogni o‘chirish**

shutil.rmtree("yangi\_papka") # papka va ichidagi barcha fayllarni o‘chiradi

➡ os.rmdir() faqat bo‘sh papkani o‘chiradi, shutil.rmtree() esa to‘liq papkani ichidagilari bilan o‘chiradi.

**4. Arxivlash va arxivni ochish**

# Papkani arxivlash

shutil.make\_archive("backup", "zip", "asosiy\_papka")

# Arxivni ochish

shutil.unpack\_archive("backup.zip", "chiqarilgan\_papka")

**5. Disk bo‘sh joyini tekshirish**

total, used, free = shutil.disk\_usage("C:/")

print("Umumiy hajm:", total // (2\*\*30), "GB")

print("Band joy:", used // (2\*\*30), "GB")

print("Bo‘sh joy:", free // (2\*\*30), "GB")

**6. Fayl atributlarini olish**

stat = shutil.stat("test.txt")

print(stat.st\_size, "bayt") # fayl hajmi

**Amaliy misollar**

**Misol 1: Avtomatik zaxira nusxa yaratish**

import shutil

shutil.copy("ma’lumot.txt", "ma’lumot\_backup.txt")

print("Zaxira yaratildi.")

**Misol 2: Papkani to‘liq ko‘chirish**

import shutil

shutil.move("projekt", "arxiv/projekt")

**Misol 3: Papkani zip arxiv qilish**

import shutil

shutil.make\_archive("projekt\_backup", "zip", "projekt")

print("Arxiv yaratildi.")

**Xulosa**

shutil moduli **os modulini to‘ldiruvchi** hisoblanadi va fayllar bilan ishlashda qulaylik yaratadi.  
U orqali:

* fayl va kataloglarni nusxalash,
* ko‘chirish,
* o‘chirish,
* arxivlash va arxivni ochish,
* disk hajmini tekshirish mumkin.

Siz xohlaysizmi, endi men sizga **os va shutil modullarini solishtirib**, ularning farqlari va qaysi holatda qaysi biridan foydalanish kerakligini jadvalda tushuntirib beray?

**os va shutil modullarini solishtirish**

Quyidagi jadvalda ikkala modulning imkoniyatlari, farqlari va qaysi vaziyatda foydalanilishi kerakligi ko‘rsatilgan:

**1. Asosiy farqlar**

| **Xususiyat / Amallar** | **os moduli** | **shutil moduli** |
| --- | --- | --- |
| Fayl yoki katalog yaratish | ✅ os.mkdir(), os.makedirs() | ❌ (yo‘q) |
| Fayl yoki katalogni o‘chirish | ✅ os.remove(), os.rmdir() | ✅ shutil.rmtree() (ichidagilar bilan) |
| Faylni o‘qish/yozish | ❌ (buning uchun open() ishlatiladi) | ❌ |
| Fayl mavjudligini tekshirish | ✅ os.path.exists() | ❌ |
| Katalog tarkibini ko‘rish | ✅ os.listdir() | ❌ |
| Fayl yo‘llari bilan ishlash | ✅ os.path funksiyalari (join, split) | ❌ |
| Faylni nusxalash | ❌ | ✅ shutil.copy(), shutil.copy2() |
| Faylni ko‘chirish | ❌ | ✅ shutil.move() |
| Arxiv yaratish/ochish | ❌ | ✅ shutil.make\_archive(), shutil.unpack\_archive() |
| Disk xotirasi haqida ma’lumot | ❌ | ✅ shutil.disk\_usage() |

**2. Qachon os, qachon shutil ishlatish kerak?**

* Agar sizga **katalog yaratish, o‘chirish, fayl mavjudligini tekshirish, yo‘l bilan ishlash** kerak bo‘lsa → **os** modulidan foydalaniladi.
* Agar sizga **fayl yoki katalogni nusxalash, ko‘chirish, arxivlash, to‘liq katalogni ichidagilari bilan o‘chirish** kerak bo‘lsa → **shutil** modulidan foydalaniladi.

**3. Misol – birgalikda ishlatish**

import os, shutil

# Fayl mavjudligini tekshirish

if os.path.exists("test.txt"):

# Faylning nusxasini olish

shutil.copy("test.txt", "nusxa.txt")

print("Nusxa yaratildi.")

else:

print("test.txt mavjud emas.")

✅ **Xulosa:**

* **os** → **asosiy fayl va kataloglarni boshqarish** uchun ishlatiladi.
* **shutil** → **nusxalash, ko‘chirish, arxivlash kabi yuqori darajadagi amallar** uchun ishlatiladi.

Siz xohlaysizmi, endi men ushbu mavzuni **umumiy xulosa va test savollari** bilan yakunlab beray?

Zo‘r ✅ Endi siz uchun **savol–javob uslubida** os va shutil bo‘yicha misollar tuzib beraman.

## ❓ Savol 1.

Joriy katalogdagi barcha fayl va papkalarni ekranga chiqaring.

### ✅ Yechim:

import os

for item in os.listdir():

print(item)

## ❓ Savol 2.

projects nomli yangi katalog yarating va uni keyin o‘chiring.

### ✅ Yechim:

import os

# Katalog yaratish

os.mkdir("projects")

print("projects katalogi yaratildi")

# Katalogni o‘chirish

os.rmdir("projects")

print("projects katalogi o‘chirildi")

## ❓ Savol 3.

notes.txt faylini backup\_notes.txt nomi bilan nusxalang.

### ✅ Yechim:

import shutil

shutil.copy("notes.txt", "backup\_notes.txt")

print("notes.txt fayli nusxalandi → backup\_notes.txt")

## ❓ Savol 4.

backup\_notes.txt faylini archive nomli katalogga ko‘chiring. Agar archive katalogi mavjud bo‘lmasa, uni yarating.

### ✅ Yechim:

import os, shutil

if not os.path.exists("archive"):

os.mkdir("archive")

shutil.move("backup\_notes.txt", "archive/backup\_notes.txt")

print("backup\_notes.txt fayli archive katalogiga ko‘chirildi")

## ❓ Savol 5.

source\_folder katalogini uning ichidagi barcha fayllari bilan birga source\_folder\_copy nomi bilan nusxalang.

### ✅ Yechim:

import shutil

shutil.copytree("source\_folder", "source\_folder\_copy")

print("source\_folder katalogi → source\_folder\_copy ga nusxalandi")

## ❓ Savol 6.

old\_data katalogini ichidagi barcha fayllari bilan birga butunlay o‘chiring.

### ✅ Yechim:

import shutil

shutil.rmtree("old\_data")

print("old\_data katalogi butunlay o‘chirildi")

❓ Siz xohlaysizmi, men shu savollarni **10–15 ta qilib test–masala shaklida** tuzib beray?

### 📌 3-mavzu: ****Fayl va katalog yo‘lini o‘zgartirish (Python**** os ****va**** shutil****)****

Python’da fayl yoki katalogni boshqa joyga ko‘chirish yoki yo‘lini o‘zgartirish uchun asosan **os** va **shutil** modullari ishlatiladi.

## 🔹 1. Faylni boshqa papkaga ko‘chirish (shutil.move)

import shutil

# Faylni yangi joyga ko‘chirish

shutil.move("notes.txt", "archive/notes.txt")

print("notes.txt → archive papkasiga ko‘chirildi")

📌 Agar archive papkasi mavjud bo‘lmasa, avval uni os.mkdir() bilan yaratish kerak.

## 🔹 2. Fayl nomini o‘zgartirish (os.rename)

import os

# Fayl nomini o‘zgartirish

os.rename("old\_name.txt", "new\_name.txt")

print("Fayl nomi o‘zgartirildi: old\_name.txt → new\_name.txt")

## 🔹 3. Katalog nomini o‘zgartirish

import os

# Katalog nomini o‘zgartirish

os.rename("data", "data\_backup")

print("Katalog nomi o‘zgartirildi: data → data\_backup")

## 🔹 4. Faylni bir papkadan boshqasiga ko‘chirish (yo‘lni o‘zgartirish)

import os

# Faylni boshqa katalogga ko‘chirish

os.rename("archive/notes.txt", "backup/notes.txt")

print("notes.txt fayli archive → backup papkasiga o‘tkazildi")

## 🔹 5. Faylning ****to‘liq yo‘lini olish**** (os.path)

import os

file\_path = os.path.abspath("notes.txt")

print("Faylning to‘liq yo‘li:", file\_path)

## 🔹 6. Faylning faqat papka qismini olish

import os

file\_path = "/home/user/documents/notes.txt"

directory = os.path.dirname(file\_path)

print("Papka yo‘li:", directory)

## 🔹 7. Faylni katalog va nom qismlariga ajratish

import os

path = "/home/user/docs/report.pdf"

dir\_name, file\_name = os.path.split(path)

print("Katalog:", dir\_name)

print("Fayl nomi:", file\_name)

✅ Xullas, **os.rename()** – nom va yo‘lni o‘zgartiradi,  
**shutil.move()** – fayl yoki katalogni yangi joyga ko‘chiradi.

Siz xohlaysizmi, men bu mavzu bo‘yicha ham **savol–javob ko‘rinishida masalalar** tuzib beray?

### 📝 3-mavzu: ****Fayl va katalog yo‘lini o‘zgartirish****

Quyida siz uchun **savol–javob (masala va yechim)** ko‘rinishida misollar tayyorladim:

## ❓ Savol 1:

data.txt faylini **backup** papkasiga ko‘chiring.

✅ **Yechim:**

import shutil

# Faylni backup papkasiga ko‘chirish

shutil.move("data.txt", "backup/data.txt")

print("Fayl muvaffaqiyatli ko‘chirildi!")

## ❓ Savol 2:

old\_report.docx faylining nomini **new\_report.docx** ga o‘zgartiring.

✅ **Yechim:**

import os

# Fayl nomini o‘zgartirish

os.rename("old\_report.docx", "new\_report.docx")

print("Fayl nomi o‘zgartirildi!")

## ❓ Savol 3:

images papkasining nomini **photos** ga almashtiring.

✅ **Yechim:**

import os

# Katalog nomini o‘zgartirish

os.rename("images", "photos")

print("Papka nomi o‘zgartirildi!")

## ❓ Savol 4:

notes.txt fayli **archive** papkasida joylashgan. Uni **documents** papkasiga o‘tkazing.

✅ **Yechim:**

import os

# Faylni boshqa papkaga ko‘chirish

os.rename("archive/notes.txt", "documents/notes.txt")

print("Fayl archive → documents papkasiga o‘tkazildi!")

## ❓ Savol 5:

project/main.py faylining to‘liq yo‘lini ekranga chiqaring.

✅ **Yechim:**

import os

path = os.path.abspath("project/main.py")

print("To‘liq yo‘l:", path)

## ❓ Savol 6:

Quyidagi yo‘ldan papka va fayl nomini ajratib ko‘rsating:  
/home/user/docs/report.pdf

✅ **Yechim:**

import os

path = "/home/user/docs/report.pdf"

dir\_name, file\_name = os.path.split(path)

print("Papka:", dir\_name)

print("Fayl:", file\_name)

**📝 4-mavzu: Katalog va fayl bilan ishlovchi funksiya va metodlar**

Bu mavzuda **asosiy funksiyalar va metodlar**ni misollar bilan tushuntirib, har biriga **savol–javob (masala–yechim)** tayyorlab berdim.

**📂 Fayl va katalog bilan ishlashda ishlatiladigan metodlar**

**OS modulida:**

* os.mkdir(path) – bitta papka yaratadi
* os.makedirs(path) – ichma-ich papkalar yaratadi
* os.listdir(path) – berilgan papkadagi fayl va papkalarni ko‘rsatadi
* os.remove(path) – faylni o‘chiradi
* os.rmdir(path) – bitta bo‘sh papkani o‘chiradi
* os.removedirs(path) – ichma-ich bo‘sh papkalarni o‘chiradi
* os.path.exists(path) – fayl/papka mavjudligini tekshiradi
* os.path.isdir(path) – bu yo‘l papkaga tegishlimi?
* os.path.isfile(path) – bu yo‘l faylgami?

**SHUTIL modulida:**

* shutil.copy(src, dst) – faylni nusxalash
* shutil.copytree(src, dst) – butun papkani nusxalash
* shutil.rmtree(path) – papkani ichidagi fayllari bilan o‘chiradi

**❓ Savol 1:**

test nomli yangi papka yarating.

✅ **Yechim:**

import os

os.mkdir("test")

print("Papka yaratildi!")

**❓ Savol 2:**

project/app/logs nomli ichma-ich papkalarni yarating.

✅ **Yechim:**

import os

os.makedirs("project/app/logs")

print("Ichma-ich papkalar yaratildi!")

**❓ Savol 3:**

documents papkasidagi barcha fayl va papkalarni ekranga chiqaring.

✅ **Yechim:**

import os

files = os.listdir("documents")

print("documents tarkibi:", files)

**❓ Savol 4:**

old.txt faylini o‘chirib tashlang.

✅ **Yechim:**

import os

os.remove("old.txt")

print("Fayl o‘chirildi!")

**❓ Savol 5:**

empty\_folder nomli bo‘sh papkani o‘chiring.

✅ **Yechim:**

import os

os.rmdir("empty\_folder")

print("Bo‘sh papka o‘chirildi!")

**❓ Savol 6:**

my\_folder papkasining mavjudligini tekshiring.

✅ **Yechim:**

import os

if os.path.exists("my\_folder"):

print("Papka mavjud")

else:

print("Papka mavjud emas")

**❓ Savol 7:**

image.png yo‘li faylmi yoki papkaga tegishlimi, aniqlang.

✅ **Yechim:**

import os

if os.path.isfile("image.png"):

print("Bu fayl")

elif os.path.isdir("image.png"):

print("Bu papka")

else:

print("Bunday yo‘l yo‘q")

**❓ Savol 8:**

report.docx faylini backup papkasiga nusxa oling.

✅ **Yechim:**

import shutil

shutil.copy("report.docx", "backup/report.docx")

print("Fayl nusxalandi!")

**❓ Savol 9:**

data papkasini ichidagi fayllari bilan birga data\_backup papkasiga nusxa ko‘chiring.

✅ **Yechim:**

import shutil

shutil.copytree("data", "data\_backup")

print("Papka nusxalandi!")

**❓ Savol 10:**

temp\_folder papkasini ichidagi barcha fayllari bilan birga o‘chirib tashlang.

✅ **Yechim:**

import shutil

shutil.rmtree("temp\_folder")

print("Papka ichidagi fayllari bilan o‘chirildi!")

Xohlaysizmi, men sizga keyingi **5-mavzu: Fayl va katalog huquqlari bilan ishlash** bo‘yicha ham shunday savol–yechimlar tayyorlab beray?

### 📝 Fayl va katalog huquqlari bilan ishlash (Python’da)

Fayllar va kataloglarda huquqlar – bu **o‘qish (read)**, **yozish (write)** va **bajarish (execute)** imkoniyatlarini belgilashdir. Python’da buni asosan **os** va **stat** modullari orqali amalga oshirish mumkin.

## 1. Fayl yoki katalog huquqlarini ko‘rish

❓ **Savol:** example.txt faylining huquqlarini ekranga chiqaring.

✅ **Yechim:**

import os

import stat

# Fayl huquqlarini olish

permissions = os.stat("example.txt").st\_mode

print("O‘qish huquqi:", bool(permissions & stat.S\_IRUSR))

print("Yozish huquqi:", bool(permissions & stat.S\_IWUSR))

print("Bajarish huquqi:", bool(permissions & stat.S\_IXUSR))

## 2. Fayl huquqlarini o‘zgartirish

❓ **Savol:** data.txt fayliga faqat **o‘qish** huquqini qoldiring.

✅ **Yechim:**

import os

import stat

# Faqat o‘qish huquqi (600 → faqat egasiga o‘qish)

os.chmod("data.txt", stat.S\_IRUSR)

print("Huquq faqat o‘qish uchun o‘rnatildi!")

## 3. Faylni ****faqat egasiga**** o‘qish va yozish huquqi berish

❓ **Savol:** secret.txt faylini faqat egasi o‘qiy olishi va yozishi mumkin bo‘lsin.

✅ **Yechim:**

import os

import stat

# O‘qish va yozish (600)

os.chmod("secret.txt", stat.S\_IRUSR | stat.S\_IWUSR)

print("Faylga faqat egasi o‘qish va yozish huquqiga ega.")

## 4. Faylni ****hamma foydalanuvchilarga**** o‘qish imkoniyatini berish

❓ **Savol:** public.txt faylini barcha foydalanuvchilar o‘qiy olishi kerak.

✅ **Yechim:**

import os

import stat

# Hammaga o‘qish huquqi (444)

os.chmod("public.txt", stat.S\_IRUSR | stat.S\_IRGRP | stat.S\_IROTH)

print("Fayl hammaga o‘qishga ruxsat qilindi.")

## 5. Fayl egasi va guruhini o‘zgartirish

❓ **Savol:** log.txt faylini user:staff egasiga o‘tkazing.

✅ **Yechim (Linux/Unix):**

import os

# user id (1000), group id (1000) deb tasavvur qilamiz

os.chown("log.txt", 1000, 1000)

print("Fayl egasi va guruhi o‘zgartirildi.")

⚠️ os.chown faqat Linux/Unix’da ishlaydi. Windows’da qo‘llab-quvvatlanmaydi.

## 6. Fayl bajariladigan (executive) bo‘lishi

❓ **Savol:** script.py faylini bajariladigan qilib qo‘ying.

✅ **Yechim:**

import os

import stat

# Egasi uchun bajarish huquqi

os.chmod("script.py", stat.S\_IRUSR | stat.S\_IWUSR | stat.S\_IXUSR)

print("script.py bajariladigan qilindi!")

Xohlaysizmi, men bu yerda **Linux terminaldagi chmod buyruqlari** bilan Python os.chmod ni solishtirib ham jadval qilib beray?

**🔑 Fayl va katalog huquqlari: Linux chmod va Python os.chmod taqqoslash jadvali**

Linux’da **chmod** orqali, Python’da esa **os.chmod** + stat kutubxonasi orqali fayl/katalog huquqlari o‘rnatiladi.

**📊 Jadval**

| **Vazifa** | **Linux chmod buyrug‘i** | **Python os.chmod kodi** |
| --- | --- | --- |
| Faqat egasi o‘qiy olsin (400) | chmod 400 file.txt | os.chmod("file.txt", stat.S\_IRUSR) |
| Egasi o‘qish va yozish huquqi (600) | chmod 600 file.txt | `os.chmod("file.txt", stat.S\_IRUSR |
| Egasi o‘qish, yozish va bajarish huquqi (700) | chmod 700 file.txt | `os.chmod("file.txt", stat.S\_IRUSR |
| Hammaga o‘qish huquqi (444) | chmod 444 file.txt | `os.chmod("file.txt", stat.S\_IRUSR |
| Hammaga o‘qish va bajarish huquqi (555) | chmod 555 file.txt | `os.chmod("file.txt", stat.S\_IRUSR |
| Hammaga o‘qish, yozish, bajarish huquqi (777) | chmod 777 file.txt | `os.chmod("file.txt", stat.S\_IRWXU |

**📌 Asosiy stat konstantalari (Python)**

* stat.S\_IRUSR → Egasi uchun o‘qish (r)
* stat.S\_IWUSR → Egasi uchun yozish (w)
* stat.S\_IXUSR → Egasi uchun bajarish (x)
* stat.S\_IRGRP → Guruh uchun o‘qish
* stat.S\_IROTH → Boshqalar uchun o‘qish
* stat.S\_IRWXU → Egasi uchun rwx
* stat.S\_IRWXG → Guruh uchun rwx
* stat.S\_IRWXO → Boshqalar uchun rwx

⚡ **Xulosa:**

* Linux’da sonli (chmod 644) yoki simvolli (chmod u+r) format ishlatiladi.
* Python’da esa **os.chmod** funksiyasi bilan stat konstantalari bit-wise (|) orqali qo‘shiladi.