# Multithreading and Multiprocessing

# Multithreading => bu malumotlarni bir vaqtning uzida uqish degani yani bir nechra malumotlarni hajmi qanaqa bulishidan qatiy nazar bir vaqtni uzida uqib javob qaytaradi bizga

## Pythonda Threadingga kirish

**Multithreading** protsessorning bir vaqtning o'zida bir nechta iplarni bajarish qobiliyati sifatida aniqlanadi. Oddiy, bir yadroli protsessorda u iplar o'rtasida tez-tez almashinish yordamida erishiladi. Bu **kontekstni almashtirish** deb ataladi . Kontekstni almashtirishda har qanday uzilish (I/U tufayli yoki qo'lda o'rnatilgan) sodir bo'lganda, ipning holati saqlanadi va boshqa ipning holati yuklanadi. Kontekstni almashtirish shunchalik tez-tez sodir bo'ladiki, barcha iplar parallel ravishda ishlayotganga o'xshaydi (bu **multitasking** deb ataladi ).

|  |
| --- |
|  |

# Biz bundan foydalanish uchun threading kutub xonasidan foydalanamiz

# Keeling soda cod sifatida korsatib utaman

import threading  
import time  
import requests  
import json  
  
  
  
def print\_numbers():  
 for i in range(1, 10):  
 print(i)  
 time.sleep(2)  
  
  
def print\_characters():  
 for i in 'ABCDEFG':  
 print(i)  
 time.sleep(1)  
  
#  
# # print\_numbers()  
thread1 = threading.Thread(target=print\_numbers)  
thread2 = threading.Thread(target=print\_characters)  
thread1.start()  
#  
thread2.start()  
thread1.join()

ushbu codimizda soddagina ketma ket malumotlarni uqish jarayonini kurishimiz mumkun

Multiprocessing=>bu bizga ishlamay turgan prosessorni ishlatib beradi desak ham buladi yani biz biror filega malumot qushmoqchi bulsak usha malumotni agarda bir nechrta filega maulot qushmoqchi bulsak bir vaqtni uzida u filega qanaqa hajmdagi malumot qushishimizdan qatiy nazar bir vaqtda qushadi yani biz bunda ham vaqtdan yuqamiz lekin bizga koproq quvvat va xotira talab qiladi prosessorimizga yuklama tushadi

Basilar buni Asincrom bilan adashtiradi u bilan buni farqi Asyncromda malumotlar eng kopi yani ishlashi eng kech ishlab bulgunicha kutadi va keyin bizga natija qaytaradi yani bu 1-funcsiyamiz 2 sekundda ishlab bulsa 2-funcsiya 4 sekunnda ishlasa dasturni 4 sekundda natija chiqaradi bizga a lekn multiproceesingda esa bir vaqtda birdaniga natija qaytaradi bzga

Keeling buni qisqacha kodda korsatib utamiz.

import multiprocessing  
import time  
from datetime import datetime  
  
def write\_to\_file(file\_path, data):  
 time.sleep(3)  
 with open(file\_path, 'w') as file:  
 file.write(data)  
 print(f"Data written to {file\_path}")  
def main():  
 file\_path1 = 'file1.txt'  
 file\_path2 = 'file2.txt'  
 process1 = multiprocessing.Process(target=write\_to\_file, args=(file\_path1, file\_path1))  
 process2 = multiprocessing.Process(target=write\_to\_file, args=(file\_path2, file\_path2))  
 process1.start()  
 process2.start()  
  
 # Wait for both processes to finish  
 process1.join()  
 process2.join()  
  
 print("Main process continues execution.")  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 print(datetime.now())  
 main()  
 print(datetime.now())

korishingiz mumkun ikkita filega maulumot qushayabmiz bir vaqtning uzida yani 3 sekundda qushadi

lekn biz qushayotgan malumotlar hajmi ikkitasida ham har xil usha uchun ham multiprocessing bizga kerak vaqtdan yutish uchun

Agarda biz dasturimizda for aperatoridan foydalansang multiprocessingning bizga nafi tegmaydi sabab for bizga ketma ket chiqarib ber degani hisoblanadi u chiziqli ishlab beradi bizga yani birinchi filega malumot qushadi ikkichisiga keyin qushadi qarangki vaqtdan yutqazib buldik usha uchun multiprocessingda for aperatoridan foydalanmaymiz

endi Multihreading va Multiprocessing ning asosiy farqi Multhreading xotiradan qoshimcha foydalanib yechadi Multiprocessing esa prosessorni ishlamay turgan qismidan foydalanib yechadi