Напредни Пајтон 5

Паралелизам, прије него конкурентност

Увод

- Нити (multithreading)
- Процеси (multiprocessing)
- Асинк (asyncIO)

Процеси, нити и асинк

Процеси и нити

Multiprocessing

- нови процес на OS
- много се спорије покреће
- Нема дијељења меморије (али има IPC и queue)
- Нема мутекса, моја меморија је моја меморија
- Један GIL (Global Interpreter Lock) по процесу

Multithreading

- Све нити у једном процесу
- Брзо стартује
- Дијели меморију међу нитима
- мутекси, семафори и остало контролишу дијељене ресурсе
- Један GIL за све

Шта је Нит?

- Посебан ток извршавања – иако се не извршавају истовремено, тако изгледа
- Само личе на процесе
- Кад неки дио кода чека на нешто је добар кандидат. НПР Техничар чека на Ишју
- За разлику од Ц кода, овдје све нити остају у једном контексту.
- Демонске Нити – Умиру са главним програмом
  - https://github.com/python/cpython/blob/df5cdc11123a35065bbf1636251447dobfe789a5/Lib/threading.py#L1263
  - `daemon=True`

Нити покретање

- Помоћу функције коју уграђујемо у Thread
- Или у ThreadPoolExecutor

- Помоћу класе Thread коју насљеђујемо и преписујемо метод run()

И главни програм је НИТ – на шта се често заборави
Али даље све је исто:
start()
join()

Race contition или Кондиција расе ☺

- Мутекси
  - Када треба да заштитимо вриједности у меморији
- Семафори
  - Када треба да успоставимо одређени редослијед
- Низови - queue
  - И једно и друго

Задатак

- Направити систем у којем Клијент креира ишјуе, а преузимају их Техничке особе.
  - Преузимају по својој могућности : Јуниор и Вођа не могу најтеже
- Послије сваког преузимања покреће се низ израчунавања морала и исписује се просјечан морал канцеларије.
- Послије креирања сваког ишјуа Клијент и Вођа имају интеракцију која Вођи окида додатно израчунавање морала

AsyncIO

- Све се дешава у једној нити и на једном процесору, тачније кору!
- Па како онда може бити брже? И зашто се зове асинхроно и конкурентно?
- Најбољи примјер је симултани меч шаховског велемајстора са групом обичних смртника.
  - 1 велемајстор, вуче потез у 5 секунди
  - 12 обичних смртника, вуче потез у 55 секунди
  - Велемајстор их матира за цирка 40 потеза, укупно 80
- Управо то што је једна страна спорија, даје могућност да се друга бави другим стварима и тако уштеди на времену

async/await и остали концепти

- Од Пајтона 3.5 уведене су нове кључне ријечи async и await
- async – дефинише функција као корутину – специјализовани генератор (функција која помоћу yield може да на кратко преда контролу неком другом)
- await – дефинише мјесто у функцији гдје ће функција предати контролу.

```
async def unutrasnja_funkcija():
    # simuliramo neko cekanje
    await asyncio.sleep(1)
    return 1  # ovo je mjesto


async def vanjska_fja():
    # ovdje ide neki kod koji je sinhron

    # ovdje mozes da ides i radis nesto korisno
    # jer je ova unutrasnja funkcija spora
    # vrati se kada unutrasnja funkcija zavrsi
    retval = await unutrasnja_funkcija()

    # ovdje opet moze biti nekog koda
    return retval # ovo je mjesto
```