(PTIA1201) Elemi programozás

Dr. Facskó Gábor, PhD
tudományos főmunkatárs
facskog@gamma.ttk.pte.hu

Pécsi Tudományegyetem, Természettudományi Kar, Matematikai és Informatikai Intézet, 7624 Pécs, Ifjúság utja 6.
Wigner Fizikai Kutatóközpont, Űrfizikai és Űrtechnikai Osztály, 1121 Budapest, Konkoly-Thege Miklós út 29-33.
`https://facsko.ttk.pte.hu`

2024. november 29.

Programozási tételek, egyszerű feladatok

- Lineáris egyenletrendszerek megoldása Gauss-eliminációval
- ~~Vektoriális szorzat bevezetése a NegyesVektor osztályba~~
- Számológép fejlesztése

Grafikus felület (GUI)

- Tkinter csomag
- Nagyon egyszerű alkalmazás
- Lebutított számológép készítése
  - Gombok 0-9, összeadás, szorzás, kilépés, törlés
  - Átméretezhető gombok és számológép.

Párhuzamosítás I

- Hogyan tudunk függvényeket párhuzamosan futtatni?
  - A webes alkalmazásoknál célszerű, hogy egyszerre, párhuzamosan sokan tudjuk használni
  - A processzor gyakran vár valamilyen erőforrásra, pl. egy webes letöltésnél
  - A többmagos processzorok korában ki szeretnénk használni a tényleges párhuzamosítás lehetőségét
- A párhuzamosítás a tényleges párhuzamos végrehajtást jelenti, ami a többmagos processzorok kihasználásával valósul meg
- Áttekintés
  - Egy processzormagunk van, és egy programszálat használhatunk
  - Nem valódi párhuzamosítás, hanem egy ideig az egyik fut, utána a másik, majd ismét az egyik, így folytatva, amíg be nem fejeződnek
    Pl. weboldalak letöltéseű

Párhuzamosítás II

- Aszinkron futtatás
  - Az aszinkron függvények képesek egy hosszabb, processzort nem igénylő művelet előtt átadni a futási lehetőséget, majd visszatérni, ha ismét lehetőséget kapnak és a művelet véget ért

Párhuzamosítás III

- Generátorok
  - A generátorok olyan függvények, amelyek az eredményt nem a yield kulcsszó segítségével adják vissza
  - A return befejezi a függvény futását, a yield később folytatja.
  - Egy programon belül több yield is lehet, ugyanabban a szekvenciában, vagy akár cikluson belül.
  - A generátorok átmenetet képeznek a függvények és az osztályok között.
  - Kinézetre szinte egy az egyben olyanok mint a függvények, viszont a futtatáshoz példányosítani kell mint az osztályokat, és a next() globális függvény hívásával vagy cikluson belül tudjuk elérni az eredményeket
  - Ha a generátor véget ért, azaz már nincs további yield, akkor a következő next() hívás eredménye a StopIteration kivétel. Ezt a ciklus automatikusan lekezeli.
  - Példaprogram

Párhuzamosítás IV

- Coroutine
  - A coroutine egy függvény, ami elé a fejlécben odaírjuk azt, hogy async
  - A coroutine meghívásakor a függvény elé kell írni az await kulcsszót
  - Az await kulcsszót viszont csak olyan függvénybe tudjuk írni, ami maga is async
  - A 22-es csapdáját az asyncio.run() függvény segítségével oldhatjuk fel
- Az aszinkron függvények hívásának akkor van értelme, ha egyszerre többet hívunk
- Ehhez taszkokat hozhatunk létre az asyncio.create_task() függvény segítségével
- Példaprogram

  !!! blokkoló time.sleep(1) vs. nem blokkoló await asyncio.sleep(1)
- Az async függvénynek lehet visszatérési értéke, amit értékül kap az await hívás
- Az aszinkron programozásban gyakori, hogy több függvényt párhuzamosan futtatni, majd összevárni mindegyik eredményét
  - Egy listába tesszük a taszkokat, majd egy másik ciklusban mindegyikre meghívjuk az await-et

Párhuzamosítás V

- A gather() függvény: paraméterül tetszőleges számú aszinkron függvényt kell átadni, és az gondoskodik az összevárásról.
  Példaprogramok
- Aszinkron fájlkezelés - példaprogram
- Aszinkron netes letöltés - példaprogram

Vége

Köszönöm a figyelmüket!