Mentor Klub

Python tanfolyam

10. alkalom

Gyakorló- és házi feladatok - megoldás

- Készítsünk interaktív számológép programot.
- Induláskor kérjen be a felhasználótól egy egyszerű matematikai feladatot, mely 2 operandusból és 1 operátorból (+,-,*,/) áll.
  - A négy alapműveletet kell támogatni, a megfelelő matematikai szabályokkal.
    - Például a 0-val történő osztás nem engedélyezett. (Hiszen nincs értelme)
- Helyes feladat esetén a program számolja ki az eredményt.
- A programból az exit paranccsal lehet kilépni, melyet a felhasználó ír be.
- A program mindaddig kérjen be új feladatot, amíg a felhasználó nem lép ki.
- Hibás feladat esetén a program írja ki, hogy hibás feladat, és a hibás feladat okát is írja ki.
- Használjunk kivételkezelést és bontsuk függvényekre a programunkat.
- Az egyes függvények legyenek némák, azaz nem írhatnak ki semmit a kimenetre. Csak a főprogram írhat a kimenetre.

Ökölszabály a fejlesztés során

Mentor Klub

- Ha már valaki egyszer adott megoldást egy problémára
- És az a megoldás
  - Szabad hozzáférésű (Free Open Source Software – FOSS)
  - Több projektben is használatban van (azaz folyamatosa tesztelik)
  - Folyamatosan karbantartartott (jelennek meg belőle újabb verziók)
  - Létezik az általunk használt programozási nyelven
- És a projektünk vezetőjétől engedélyt kapunk a használatára

AKKOR VEGYÜK LE A “POLCRÓL”, ÉS INTEGRÁLJUK A PROJEKTÜNKBE

Adattípusok – Valami még hiányzik…

- Eddig megismerkedtünk az alapvető adattípusokkal, úgy mint:
  - numerikus (int, float), szöveges (str) logikai (bool), None
  - konténer típusokkal: lista (list), rendezett n-es (tuple), kulcs-érték párookal (dict)
- Azonban van még egy (kettő?) elemi típus, ami előbb utóbb szükséges lesz a szoftverfejlesztés során.
  - Ez a dátum típus
  - Pontosabban a datum és az idő leírására szolgáló típusrendszer.

Dátumok és Időpontok ábrázolása

Mentor Klub

- A dátumok ábrázolása és kezelése az IT egyik legösszefüggőbb kérdésköre.
- Pedig egy mindennapos dologról van szó… De nézzük csak….
  - Időzónák → Időzónák miatti datum elcsúszások (Magyarországon 1:00 az USA-ban még “tegnap”)
  - Szökőévek
  - Téli-nyári időszámítás (DST – Daylight saving time)
    - Magyarországon CET és CEST között ugrálunk, ami GMT+1 és GMT+2 (Greenich Mean Time)
  - Különböző naptárak használata (Gregorián, Julián)
  - Különböző formátumok
    - 12 órás vagy 24 órás
    - Év-hónap-nap / Nap-hónap-év, stb…

Standardekre van szükség

- Y2K (aki még emlékszik): A 70-80 években a programozók úgy gondolták, elég, ha 2 számjegyen tárolják az éveket. Tévedtek…
- Van egy adattípus, UNIX timestamp a neve. Ez az epoch-tól eltelt másodpercek számát jelenti.
  - Az epoch 1970 január 1. 00:00
  - Attól függően, hogy hány byte-on ábrázolják előbb-utóbb újból baj lehet
  - Csak hát ha a “Mennyi a pontos idő?” kérdésre az a válasz, hogy 1677085210, akkor nem sokan fogják érteni.
  - Szóval ez egy programozóknak szóló standard lett az évek során

De a felhasználóink nem csak programozók

- Van szabvány a datum és pontos idő kezelésére: ISO 8601
- A legelterjedtebb dátumformátum:
  - 2023-02-22T18:02:34.70+02:00
- Ez a szabvány választ ad rengeteg dátumokkal kapcsolatos kérdésre.
- A különböző programozási nyelvekben külön dátumkezelő könyvtárakat készítettek
  - Többféle dátumformátumot kezelnek
  - Konzekvensen kezelik a DST-t
  - Támogatják a dátumokkal kapcsolatos műveleteket

A datetime package

- Pythonban a datetime package használatával kezelhetünk dátumokat.
- Használata:

```
from datetime import datetime
#most
most=datetime.now() #Ennek a típusa egy datetime objektum
#év lekérdezése
print(most.year) #Nem függvény, hanem attribútum
#Természetesen a többi is lekérdezhető
```

Ha műveletet is szeretnénk végezni

```
#Akkor csak a datetime package-t importáljuk be
import datetime
most = datetime.datetime.now()
holnap = most + datetime.timedelta(days=1)
#Több más mértékegység is lehetséges

#Datetime objektumok összehasonlítása a <,>,==
relációs jelekkel történik.
```

Dátumok beolvasása szövegből

- Az `strptime()` függvény segítségével történik
  - Az első paramétere egy str típusú objektum, amiben a dátum van
  - A második paramétere egy ún. formatter string, amiben az első paraméterként kapott szöveg értelmezéséhez szükséges információ van.
- Cheat sheet: https://strftime.org/
- A leggyakrabban használtak:

Formatter	Jelentés
%Y	Évszám, pl 2023
%m	Hónap, vezető nullával, pl 02
%d	Nap, vezető nullával, pl 09
%H	Óra, 24 órás formátum, vezető nullával
%M	Perc, vezető nullával
%S	Másodperc, vezető nullával

Datetime konvertálása szöveggé

- Egy datetime objektumot az `strftime()` függvénnyel készíthetünk egy új szöveges objektumot, mely annak a formatternek fog megfelelni, amit paraméterként adunk át.
- Ugyanazok a formatterek használhatóak itt is, mint az strptime() esetén.
- Ezt a függvényt egy már létező datetime objektumon kell meghívni

```
from datetime import datetime
most=datetime.now()
print(most.strftime("%Y-%m.%d"))  #Direkt ilyen ☺
```

Gyakorló- és házi feladatok

Készítsünk programot, mely

- Bekér a felhasználótól egy dátumot *(A formátum rád van bízva ☺)*
- Bekér a felhasználótól egy fájlnevet
- A fájlban legyenek dátumok, soronként 1
  - A datum formátuma `2023.02.22` alakban legyen megadva.
- A program írja ki egy output_PONTOSIDŐ.txt nevű fájlba azokat a dátumokat 2023-02-22 alakban, melyek későbbiek a felhasználó által megadott dátumnál.
- A fájlnévben a PONTOSIDŐ 20230222185917 formátumú legyen

Gyakorló és házi feladatok (2)

- Fejlesszük tovább a programot úgy, hogy megfelelő kivételkezeléssel biztosítjuk a hiba nélküli futást.
  - Ha a felhasználó rossz formátumban adja meg a dátumot, kérjük be tőle mindaddig, amíge nem sikeres.
  - Ha a beneti fájl nem létezik, kérjünk be új fájlnevet.
  - Ha a fájlban lévő datum nem megfelelő formátumú, egyszerűen hagyjuk ki.