ÉRETTSÉGI – 2020.10.28.

Feldolgozandó fájl részlete:

```
2017.07.16
Games
7x01
60
1
2017.07.23
Games
7x02
60
1
…
```

1. Olvassa be és tárolja el a lista.txt fájl tartalmát!

```
# Beolvasás egy listába
i = 10
# Lista a sorozatoknak
sorozatok = list()
# Lista egy-egy résznek
resz = list()
# Számoljuk a sorokat
sorindex = 1
with open("lista.txt") as file:
    for line in file:
        line = line.strip('\n')
        # 4. és 5. sorban lévő értéket számmá konvertáljuk
        if sorindex == 4 or sorindex == 5:
            line = int(line)
        resz.append(line)
        # Ha elértük az ötödik sort:
        # Eltároljuk az összetartozó adatokat a sorozatok listába
        # Alapértékre állítjuk a rész listát és a sorindexet
        if sorindex == 5:
            sorozatok.append(resz)
            sorindex = 1
            resz = list()
        else:
            sorindex += 1
```

2. Írassa ki a képernyőre, hogy hány olyan epizód adatait tartalmazza a fájl, amelynek ismert az adásba kerülési dátuma!

A dátumokat mindig „éééé.hh.nn” formátumban rögzítették. Vannak olyan sorozatrészek, amelyeknek a lista rögzítésekor még nem tudták az adásba kerülésük idejét. Ezeknél a dátum helyett mindig az „NI” rövidítés szerepel.

```
# Megszámlálás, hány olyan elem van a 0. indexen, ami nem „NI”
szamlalo = 0
for i in sorozatok:
    if i[0] != "NI":
        szamlalo += 1
print("A listában {} db vetítési dátummal rendelkező epizód
van.".format(szamlalo))
```

3. Határozza meg, hogy a fájlban lévő epizódok hány százalékát látta már a listát rögzítő személy! A százalékértéket a minta szerint, két tizedesjeggyel jelenítse meg a képernyőn!

Az epizóddal kapcsolatos utolsó adat értéke „0” vagy „1”. Az 1-es számjegy jelöli, hogy az adott részt már megtekintette a lista készítője, a 0 pedig azt, hogy még nem látta.

```
# Megszámlálás, hány sora van a sorozatok listának
  elemek = len(sorozatok)

# Megszámlálás, hány olyan elem van az utolsó indexen, aminek az értéke 1
  megnezett = 0
  for i in sorozatok:
      if i[-1] == 1:
          megnezett += 1

# Százalékszámítás, kiíratás
  szazalek = megnezett / elemek * 100
  print("A listában lévő epizódok {:.2f}%-át látta".format(szazalek))
  # vagy
  print("A listában lévő epizódok {}%-át látta".format(round(szazalek,2)))
```

4. Számítsa ki, hogy összesen mennyi időt töltött a személy az epizódok megnézésével! Az eredményt a minta szerint nap, óra, perc formában adja meg!

Egy-egy epizód hossza a 3-as indexen van rögzítve, hogy meg lett e tekintve, az pedig az utolsó indexen.

```
# Összegszámítás, ha megnézte, akkor egy változóban összesítésre kerülnek a percek
  percek = 0
  for i in sorozatok:
      if i[-1] == 1:
          percek += i[3]

# Lebontás napra, órára, percre. Kiíratás
  nap = percek / 60 // 24
  # Maradék számítása
  percek = percek - (nap * 24 * 60)
  ora = percek // 60
  # Maradék számítása
  percek = percek - (ora * 60)
  print("Sorozatnézéssel {} napot {} órát és {} percet töltött".
        format(int(nap), int(ora), int(percek)))
```

5. Kérjen be a felhasználótól egy dátumot „éééé.hh.nn” formában! Határozza meg, hogy az adott dátumig megjelent epizódokból melyeket nem látta még! Az aznapi epizódokat is számolja bele! A feltételnek megfelelő epizódok esetén írja a képernyőre tabulátorral elválasztva az évad- és az epizódszámot, valamint a sorozat címét a minta szerint!

Át kell konvertálni a lista 0. indexén lévő értéket dátum formátumúvá.
Be kell kérni egy dátumértéket, szintén dátum formátumban.
Ki kell íratni a feltételnek megfelelő sorok 2. és 1. indexét.

```
# Lista első elemének konvertálása dátumértékké
  #Első sorba
  for i in range(len(sorozatok)):
      # NI értékek miatt
      try:
          sorozatok[i][0] = datetime.datetime.strptime(sorozatok[i][0],
                                                       '%Y.%m.%d').date()
      except:
          pass

# Lista első elemének konvertálása dátumértékké (Dasdi módszer)
  #Első sorba
  for i in range(len(sorozatok)):
      # NI értékek miatt
      if sorozatok[i][0] != "NI":
          sorozatok[i][0] = datetime.datetime.strptime(sorozatok[i][0],
                                                       '%Y.%m.%d').date()

# Dátumérték bekérése a felhasználótól
  datum = input("Adjon meg egy dátumot a következő formátumban (éééé.hh.nn)!
  Dátum = ")
  datum = datetime.datetime.strptime(datum,'%Y.%m.%d').date()

# Dátumértékek összehasonlítása és kiíratás
  for i in sorozatok:
      # NI értékek miatt
      try:
          if i[0] >= datum:
              print(i[2], i[1], sep="\t")
      except:
          pass
```

6. Készítse el az alábbi algoritmus alapján a hét napját meghatározó függvényt!

Csak át kell írni a leírtakat a nyelvnek megfelelően.

```
# A függvény
  def hetnapja(ev, ho, nap):
      napok = ['v','h','k','sze','cs','p','szo']
      honapok = [0, 3, 2, 5, 0, 3, 5, 1, 4, 6, 2, 4]
      if ho < 3: ev -= 1
      hetnapja = napok[ (ev + ev // 4 - ev // 100 +
                         ev // 400 + honapok[ho - 1] + nap) % 7 ]
      return hetnapja
```

7. Kérjen be a felhasználótól egy napot az előző feladatban látható rövidített alakban (h, k, sze, …)! Határozza meg, hogy a fájlban lévő sorozatok közül melyike(ke)t vetítik az adott napon! A sorozatok nevét jelenítse meg a képernyőn! Ha az adott napon egy sorozatot sem adtak adásba, akkor „Az adott napon nem kerül adásba sorozat.” üzenetet jelenítse meg!

Bekérés a felhasználótól.

Végig menve a sorozatok listán, a dátumértékeket elhelyezzük a hetnapja függvénybe és a kimenetét összehasonlítjuk a felhasználó által megadottal.
Ha egyeznek kiíratjuk a sorozat első indexét.
Szükség lesz egy változóra is, hogy hány kiíratás történt, ha 0, akkor kiíratjuk a megadott üzenetet.

```
# Napérték bekérése a felhasználótól
  nap = input("Adja meg a hét egy napját (például cs)! Nap =")x < y <= z AND:

# Lista bejárása, összehasonlítás
  szamlalo = 0
  for i in sorozatok:
      try:
          if nap == hetnapja(i[0].year, i[0].month, i[0].day):
              print(i[1])
      except:
          pass
      if szamlalo:
          print("Az adott napon nem kerül adásba sorozat")

# Ha csak egyszer szerepelhet egy sorozatcím
  szamlalo = 0
  napisorozat = list()
  for i in sorozatok:
      try:
          if nap == hetnapja(i[0].year, i[0].month, i[0].day):
              napisorozat.append(i[1])
      except:
          pass
      if szamlalo:
          print("Az adott napon nem kerül adásba sorozat")

  dictionary = dict.fromkeys(napisorozat)
  napisorozat = list(dictionary)
  for i in napisorozat:
      print(i)
```

8. Határozza meg sorozatonként az epizódok összesített vetítési idejét és az epizódok számát! A számításnál vegye figyelembe a vetítési dátummal nem rendelkező epizódokat is! A megoldás során felhasználhatja, hogy egy sorozat epizódjainak adatai egymást követik a forrásállományban. A listát írja ki a summa.txt fájlba! A fájl egy sorában a sorozat címe, az adott sorozatra vonatkozó összesített vetítési idő percben és az epizódok száma szerepeljen szóközzel elválasztva!

Szükség lesz egy újabb listára, amibe csak akkor helyezünk új értéket, ha egy újabb sorozat név megjelenik. Ha már van olyan eleme az új listánknak, ami sorra kerül, akkor csak módosítjuk a már eltárolt értékeket.
Az új listában el kell tárolni a sorozat nevét, a sorozat időtartamát és egy számot, ami megszámolja, hány része van.
A végén egy fájlba kiíratjuk az elkészült új listát.

ÉRETTSÉGI – 2020.10.28.

Új lista létrehozása, kiválogatás

```
  # Első elem, elhelyezése
  # Lista struktúrájának elkészítése
  egyedi = [[sorozatok[0][1], sorozatok[0][3], 1]]
  # További elemek
  for i in range(1, len(sorozatok)):
      talalat = 0
      for j in range(len(egyedi)):
          if egyedi[j][0] == sorozatok[i][1]:
              egyedi[j][1] += sorozatok[i][3]
              egyedi[j][2] += 1
              talalat = 1
      if not(talalat):
          egyedi.append([sorozatok[i][1], sorozatok[i][3], 1])
```

Lista tartalmának fájlba írása

```
  with open("summa.txt","w") as tabla:
      for i in egyedi:
          print(i[0], i[1], i[2], file=tabla)
```