resitev

January 28, 2024

V tej nalogi se bomo ukvarjali s podnapisi, kakršne dobimo v datotekah s končnico .srt. Gre za običajne besedilne datoteke v takšnem formatu::

```
1
00:00:02,618 --> 00:00:05,019
(Finch) We are being watched.

2
00:00:05,021 --> 00:00:07,288
The government has a secret system,

3
00:00:07,290 --> 00:00:11,626
a machine that spies on you
every hour of every day.

4
00:00:11,628 --> 00:00:14,128
I designed the machine
to detect acts of terror,

5
00:00:14,130 --> 00:00:15,963
but it sees everything,
```

- Podnapisi so oštevilčeni. Prva številka je vedno 1.
- Sledita časa, ko se podnapis pokaže in izgine, vmes je niz " –> ". Čas je vedno podan v obliki `hh:mm:ss,sss`, pri čemer imajo vsi časi vedno dve mesti (po potrebi z vodilno ničlo) in sekunde vedno tri decimalke, ločene z decimalno vejico (ne piko!). (Fun fact: format so si izmislili Francozi. Zato.)
- Sledi poljubno število vrstic besedila, ki se prikaže; navadno ena ali dve, vendar moramo predpostaviti, da jih je lahko tudi več.
- Sledi ena prazna vrstica.

Končni rezultat domače naloge bo funkcija, ki zamakne vse čase za podani interval, recimo sekundo in pol. Do nje pridemo po korakih. Napiši torej naslednje funkcije.

- `razberi_cas(s)` prejme čas kot niz oblike `hh:mm:ss,sss` in vrne čas v sekundah od začetka. Klic `13:12:33,125` vrne vrednost `13 * 3600 + 12 * 60 + 33.125` (kolikor že to je).
- `razberi_interval(s)` prejme niz oblike `hh:mm:ss,sss --> hh:mm:ss,sss` in vrne terko z

obema časoma v sekundah. Klic `razberi_interval("13:12:33,125 --> 13:12:37.574")` vrne `(13 * 3600 + 12 * 60 + 33.125, 13 * 3600 + 12 * 60 + 37.574)`.

- `zapisi_cas(t)` dela ravno obrano kot `razberi_cas`: prejme čas v sekundah in vrne ustrezni niz.

  Nasvet: tega na predavanjih nisem pokazal, vendar: preveri, kaj dobiš z `f"{3.14:06.2f}"` ali z `f"{42:05}`. To bo prišlo zelo prav.

- `zapisi_interval(od, do)` dela ravno obratno kot `razberi_interval`: prejme dva časa in vrne interval v obliki niza.

- `paketek(datoteka)` prejme odprto datoteko (ne imena datoteke, temveč objekt, ki že predstavlja datoteko!) in iz nje prebere naslednji podnapis (torej vse do naslednje prazne vrstice). Funkcija mora vrniti seznam vrstic.

  - Če je gornje besedilo shranjeno v datoteki s imenom podnapis.srt, in je bila datoteka odprta z `f = open("podnapis.srt")`, mora klic `paketek(f)` vrniti `["1", "00:00:02,618 --> 00:00:05,019", "(Finch) We are being watched."]`.
  - Če nato ponovno pokličemo `paketek(f)` funkcija vrne `["2", "00:00:05,021 --> 00:00:07,288", "The government has a secret system,"]`.
  - Če pokličemo še enkrat, vrne `["3", "00:00:07,290 --> 00:00:11,626", "a machine that spies on you", "every hour of every day."]` … in tako naprej.
  - …
  - Ko je datoteke konec, vrne `None`.

  Nasvet: uporabi zanko `for`. Da je konec paketka, boš izvedel(a) po tem, da je prebrala prazno vrstico. Da je konec datoteke boš izvedel(a) po tem, da zanka ni prebrala ničesar.

- `podnapis(datoteka)` je podoben funkciji `paketek(datoteka)`. Razlikuje se po tem, da ne vrne seznama vrstic, temveč terko, ki vsebuje indeks (kot `int`), čas začetka, čas konca (oboje kot `float`) in seznam z vrsticami besedila.

  - Če je `f` sveže odprta datoteka z gornjim besedilom, naj klic `podnapis(f)` vrne `(1, 2.618, 5.019, ["(Finch) We are being watched."])`.
  - …
  - Tretji klic funkcije vrne `(3, 7.290, 11.626, ["a machine that spies on you", "every hour of every day."]`.
  - …
  - Ko je datoteke konec, naj vrne `(0, 0, 0, [])`.

- `zapisi_podnapis(datoteka, indeks, od, do, besedilo)` dela ravno obratno kot funkcija `podnapis(datoteka)`: prejme datoteko, odprto za pisanje in točno takšne podatke, kot jih vrača prejšnja funkcija. V datoteko zapiše podatke v ustreznem formatu in na konec še prazno vrstico. Klic `zapisi_podnapis(f, 13, 3663.25, 3664.5, ["foo", "bar!"])` v datoteko doda vrstice

  ```
  01:01:03,250 --> 01:01:04,500
  foo
  bar!
  ```

  Nasvet: če testi pravijo, da izpis ni pravilen, poišči vrstico “Click to see difference” in klikni. Pokazalo se bo okno s primerjavo pravilnega in dejanskega besedila, v katerem bodo označene

razlike.

- `premakni(ime_datoteke, zamik)` prejme *ime datoteke* s podnapisi, na primer `podnapisi.srt`. Ustvariti mora novo datoteko, katere ime je enako imenu vhodne datoteke, le da je dodan `_popravljena`; za gornji primer torej `podnapisi_popravljena.srt`. V novo datoteko zapiše natančno enake podnapise, le da so vsi časi povečani za `zamik` (ta je lahko tudi negativen, a to ni pomembno). Če je gornje besedilo shranjeno v `podnapisi.srt`, bomo po klicu `premakni("podnapisi.srt", 1.5)` dobili datoteko `podnapisi_popravljena.srt`, ki bo videti enako kot vhodna datoteka, le vsi časi bodo povečani za sekundo in pol.

```
1
00:00:04,118 --> 00:00:06,519
(Finch) We are being watched.

2
00:00:06,521 --> 00:00:08,788
The government has a secret system,

3
00:00:08,790 --> 00:00:13,126
a machine that spies on you
every hour of every day.

4
00:00:13,128 --> 00:00:15,628
I designed the machine
to detect acts of terror,

5
00:00:15,630 --> 00:00:17,463
but it sees everything,
```

0.0.1 Rešitev

Čas razberemo iz niza tako, da niz razbijemo glede na dvopičje in shranimo v tri spremenljivke, `h`, `m` in `s`. Prvi dve pretvorimo v celo število (`int`) in pomnožimo s 3600 oz. 60, da dobimo sekunde. V `s` pa zamenjamo `,` s `.` in pretvorimo v `float`.

[1]:
```
def razberi_cas(cas):
    h, m, s = cas.split(":")
    return int(h) * 3600 + int(m) * 60 + float(s.replace(",", "."))
```

Par števil, interval, razbijemo glede na niz `" --> "` in za vsak del pokličemo `razberi_cas`.

[2]:
```
def razberi_interval(casi):
    od, do = casi.split(" --> ")
    return razberi_cas(od), razberi_cas(do)
```

Čas zapišemo tako, da iz podanega časa v sekundah naračunamo ure, minute in sekunde; ure in minute je potrebno spremeniti v cela števila, kar spet storimo z `int`. Nato vse skupaj sestavimo v

niz tako, kot smo se učili na predavanjih.

[3]:
```
def zapisi_cas(cas):
    h, m = int(cas / 3600), cas % 3600
    m, s = int(m / 60), m % 60
    return f"{h:02}:{m:02}:{s:06.3f}".replace(".", ",")
```

Interval zapišemo tako, da sestavimo niz iz nizov, ki ju vrne prejšnja funkcija.

[4]:
```
def zapisi_interval(od, do):
    return f"{zapisi_cas(od)} --> {zapisi_cas(do)}"
```

Zdaj pa končno nekaj bolj zabavnega.

Datoteko bomo brali z zanko `for`. Vsaki vrstici odluščimo beli prostor na začetku in na koncu; predvsem se moramo znebiti znaka za konec vrste. Če je vrstica prazna, prekinemo zanko, saj je paketek končan. Če ni prazna, pa jo dodamo v seznam vrstic … ki ga vrnemo po zanki.

Po zanki preverimo, ali smo sploh dobili kakšno vrstico. Če, potem ta seznam vrnemo. Sicer vrnemo `None`.

Zadnji dve vrstici bi lahko tudi izpustili. Funkcija, ki ne vrne ničesar, vrne `None`.

[5]:
```
def paketek(datoteka):
    vrstice = []
    for vrstica in datoteka:
        vrstica = vrstica.strip()
        if not vrstica:
            break
        vrstice.append(vrstica)
    if vrstice:
        return vrstice
    else:
        return None
```

Pogoj `if not vrstica` bi lahko dopolnili v `if not vrstica and vrstice`. Po takšni spremembi bi funkcija delovala tudi, če bi bilo med dvema paketkoma več praznih vrstic.

Da dobimo podnapis, pokličemo paketek. Če vrne `None`, vrnemo, kot zahteva naloga `0, 0 ,0, []`, sicer pa iz druge vrstice razberemo začetni čas ter vrnemo indeks (`int(paketek[0])`), oba časa in ostale vrstice.

[6]:
```
def podnapis(datoteka):
    paket = paketek(datoteka)
    if not paket:
        return 0, 0, 0, []
    od, do = razberi_interval(paket[1])
    return int(paket[0]), od, do, paket[2:]
```

Podnapis zapišemo tako, da v datoteko zapišemo niz z indeksom, niz s časoma in nato vse vrstice, potem pa še eno prazno vrstico.

[7]:
```
def zapisi_podnapis(datoteka, indeks, od, do, besedilo):
    datoteka.write(f"{indeks}\n")
    datoteka.write(f"{zapisi_interval(od, do)}\n")
    for vrstica in besedilo:
        datoteka.write(vrstica + "\n")
    datoteka.write("\n")
```

Zadnja funkcija je pa malo sitna: nekateri jeziki imajo poleg zanke `while` še zanko `do-while`, Python pa ne. V resnici jo potrebujemo redko … ampak tule bi jo. Brez nje moramo pisati `while True` in v njem `break`, ali pa moramo dvakrat klicati funkcijo `podnapis`. Tule sta obe različici; razlagati pa ni kaj.

[8]:
```
def premakni(ime_datoteke, zamik):
    inp = open(ime_datoteke)
    outp = open(ime_datoteke[:-4] + "_popravljena.srt", "w")

    while True:
        indeks, od, do, besedilo = podnapis(inp)
        if indeks == 0:
            break
        od += zamik
        do += zamik
        zapisi_podnapis(outp, indeks, od, do, besedilo)

def premakni(ime_datoteke, zamik):
    inp = open(ime_datoteke)
    outp = open(ime_datoteke[:-4] + "_popravljena.srt", "w")

    indeks, od, do, besedilo = podnapis(inp)
    while indeks != 0:
        od += zamik
        do += zamik
        zapisi_podnapis(outp, indeks, od, do, besedilo)
        indeks, od, do, besedilo = podnapis(inp)
```