rešitev

January 28, 2024

1 Datotečni dnevnik

Nek računalnik hrani arhiv vseh sprememb datotek: vsakič, ko shranimo datoteko, zabeleži trenutni datum, ime datoteke in njeno dolžino. Podatki so lahko, recimo, takšni:

```
arhiv = [
    "12/31/2018 ime_dat.avi 21532",
    "10/16/2020 ime_dat.avi 314236",
    "10/16/2020 some_other.avi 314236",
    "1/1/1971 ime_dat.avi 312353",
    "5/16/2020 ime_dat.avi 21532",
    "10/18/2020 another_file.avi 351352",
    "10/18/2018 another_file.avi 314236",
]
```

Vsaka vrstica opisuje spremembo datoteke: - datum v obliki mesec/dan/leto, - ime datoteke; le-to nikoli ne vsebuje presledkov, - številka, ki predstavlja dolžino datoteke.

Med podatki so presledki. Vrstice niso urejene po datumu ali čemerkoli drugem.

Na disku, ki ga opisuje gornji seznam, so trenutno samo tri datoteke ("ime_dat.avi", "some_other.avi" in "another_file.avi"). Datoteka "ime_dat.avi" je bila shranjena štirikrat, in sicer 31. decembra 2018, 16. oktobra 2020, 1. januarja 1971, in 16. maja 2020. Na disku le zadnja različica, s 16. oktobra letos.

1.1 Obvezna naloga

Prve štiri funkcije, ki jih moraš napisati, prejmejo ime vrstice, kot je gornja. Te štiri funkcije se lahko seveda kličejo med sabo. Katera kliče katero, je stvar tvoje odločitve; nalogo je mogoče rešiti na različne načine.

- `datum(vrstica)` vrne datum kot terko (leto, mesec, dan). Klic `datum("5/15/1970 ime_dat.avi 42")` vrne `(1970, 5, 15)`.
- `ime(vrstica)` vrne ime datoteke. Klic `ime("5/15/1970 ime_dat.avi 42")` vrne niz `"ime_dat.avi"`.
- `dolzina(vrstica)` vrne dolžino datoteke. Klic `dolzina("5/15/1970 ime_dat.avi 42")` vrne 42.
- `podatki(vrstica)` vrne terko z gornjimi podatki. Klic `podatki("5/15/1970 ime_dat.avi 42")` vrne `((1970, 5, 15), "ime_dat.avi", 42)`.

Poleg teh štirih napiši še naslednji funkciji.

- `je_novejsa(s1, s2)` prejme dve vrstici in vrne `True`, če ima `s1` novejši (kasnejši) datum kot `s2`, in `False`, če ne. Klic `je_novejsa("11/16/2020 ime.txt 316", "11/15/2015 foo.txt 314")` vrne `True`.
- `najnovejsa(ime_datoteke, arhiv)` prejme ime datoteke in seznam, kot je na začetku naloge. Vrniti mora podatke o datoteki v času zadnje spremembe. Klic `najnovejsa("ime_dat.avi", arhiv)` (pri čemer je `arhiv` takšen, kot je definiran zgoraj) vrne `((2020, 10, 16), "ime_dat.avi", 314236)`, saj so to podatki o datoteki, kot je bila shranjena na zadnjega izmed štirih datumov, ko smo spreminjali to datoteko.

1.1.1 Rešitev

Prve štiri funkcije so vaje iz `split`-a. Vrstico s `split()` razsekamo na tri podnize, ki vebujejo datum, ime in dolžino, vse troje kot niz. Do teh treh delov pridemo torej z `datum, ime, dolzina = vrstica.split()`. Odtod naprej gre vsaka funkcija po svoje.

[1]:
```
def ime(vrstica):
    ime, datum, dolzina = vrstica.split()
    return ime

def datum(vrstica):
    ime, datum, dolzina = vrstica.split()
    mes, dan, let = datum.split("/")
    return int(let), int(mes), int(dan)

def dolzina(s):
    ime, datum, dolzina = vrstica.split()
    return int(dolzina)

def podatki(s):
    return datum(s), ime(s), dolzina(s)
```

Zadnja funkcija, podatki, le kliče gornje tri funkcije.

Seveda bi se dalo obrniti tudi drugače: lahko bi vse delo opravila zadnja funkcija, prve tri pa klicale njo. Pomembno je le, da si dela ne podvajamo in se te funkcije kličejo med seboj.

Preostali funkciji tudi ne bi smeli biti raketna znanost. `je_novejsa` mora le primerjati datume. Terki sta že oblikovani tako, da imamo najprej leto, nato mesec in dan. Ko Python primerja terke, najprej primerja prvi element; če sta enaka, primerja drugega in če sta enaka tudi tadva, še tretjega. Točno to, kar potrebujemo.

Funkcija `je_novejsa` pa je nekaj, kar prežvekujemo že ves čas: iskanje največjega elementa po določenem kriteriju - tokrat datumu.

[2]:
```
def je_novejsa(s1, s2):
    return datum(s1) > datum(s2)

def najnovejsa(ime, arhiv):
    naj_pod = None
    naj_dat = None
```

```
    for vrstica in arhiv:
        pod = dat, ime2, _2 = podatki(vrstica)
        if ime == ime2 and (naj_dat == None or dat > naj_dat):
            naj_pod = pod
            naj_dat = dat
    return naj_pod
```

Nekateri so funkcijo `je_novejsa` programirali zelo narobe, namreč tako:

[3]:
```
# Tole je narobe!

def je_novejsa(s1, s2):
    l1, m1, d1 = s1
    l2, m2, d2 = s2
    if l1 > l2:
        return True
    if m1 > m2:
        return True
    if d2 > d2:
        return True
    return False
```

Problem je v tem: če je `l1 > l2`, moramo res vrniti `False`. Če ni, pa ne moremo kar tako primerjati mesecev, temveč moramo prej preveriti, da sta leti enaki. Če je namreč `l1 < l2`, je potrebno vrniti `False`.

1.2 Dodatna naloga

- `datumi(ime_datoteke, arhiv)` vrne datume sprememb podane datoteke. Datumi morajo biti urejeni od kasnejših proti starejšim. Klic `datumi("ime_dat.avi", arhiv)` vrne `[(2020, 10, 16), (2020, 5, 16), (2018, 12, 31), (1971, 1, 1)]`.
- `brez(ime_datoteke, arhiv)` vrne seznam, ki je enak podanemu, le brez vrstica, ki se nanašajo na podano datoteko. Klic `brez("ime_dat.avi", arhiv)` vrne

```
["10/16/2020 some_other.avi 314236",
"10/18/2020 another_file.avi 351352",
"10/18/2018 another_file.avi 314236",
]
```

1.2.1 Rešitev

Gremo čez vse vrstice arhiva, pogledamo podatke in če je ime pravo, dodamo datum v seznam datumov. Na koncu pokličemo `sort` z argumentom `reverse=True`, da bodo datumi urejeni padajoče.

[4]:
```
def datumi(ime, arhiv):
    dats = []
    for vrstica in arhiv:
        dat, im, _ = podatki(vrstica)
```

```
        if im == ime:
            dats.append(dat)
    return sorted(dats, reverse=True)
```

V drugi funkciji pa pripravimo prazen seznam in vanj zlagamo vse vrstice, pri katerih je ime drugačno od podanega.

[5]:
```
def brez(im, arhiv):
    filtriran = []
    for vrstica in arhiv:
        if ime(vrstica) != im:
            filtriran.append(vrstica)
    return filtriran
```