Skriptni jezici – zadaci za 3. ciklus laboratorijskih vježbi

svibanj 2025.

1 Uvod

U okviru trećeg ciklusa laboratorijskih vježbi utvrđuju se i praktično primjenjuju osnove programskog jezika Python.

Studenti su dužni pripremiti se za laboratorijske vježbe samostalnim rješavanjem niza jednostavnih zadataka. Za zadatke koji obavljaju operacije nad datotekama (pretraživanje sadržaja, promjena imena i slično) potrebno je pripremiti datoteke prikladne za ispitivanje i demonstraciju rada programa.

Preduvjet za obavljanje laboratorijske vježbe je predaja (*upload*) datoteka s rješenjima zadataka kroz sustav Ferko (*https://ferko.fer.hr/ferko/*). Prilikom postavljanja datoteka treba poštovati upute o imenovanju datoteka, kao i o uvjetima pokretanja skripti (navođenje parametara i slično).
Napomena: Programi se moraju moći izvesti u Pythonu 3!

1.1 Protokol odrade laboratorijske vježbe

Ukratko, protokol odrade laboratorijske vježbe je:

1. student je dužan riješiti postavljene zadatke i datoteke s rješenjima predati kroz sustav Ferko;
2. student je dužan doći na laboratorijske vježbe u svom terminu;
3. na početku termina laboratorijskih vježbi studenti pišu kratku provjeru znanja (kviz-pitanja s ponuđenim odgovorima; točan odgovor nosi 0.5, a pogrešan -0.125 bodova);
4. tijekom laboratorijskog termina studenti dobivaju zadatak koji trebaju riješiti na licu mjesta (izlazni test). Zadatak se rješava na računalu, te se rješenje predaje kroz sustav Ferko (obavezno zaključati predaju).
   Napomene: Obavezno imenovati datoteku s rješenjem prema uputi, kao i predvidjeti pozivanje skripte iz naredbenog retka sa zadanim argumentima. Predano rješenje mora se moći izvesti u Pythonu 3!
5. tijekom laboratorijskog termina student treba pokazati svoja rješenja zadataka za pripremu asistentu, za što ga asistent ocjenjuje.

1.2 Resursi u laboratoriju

Vježbe se obavljaju u fakultetskim laboratorijima. Na računalima je instaliran `Python 3.12`, uključujući razvojnu okolinu IDLE.

Zadatak 1

Napisati skriptu koja će iz datoteke učitati dvije *rijetke* matrice, ispisati ih u punom obliku (kao 2D matricu brojeva), izračunati i ispisati njihov matrični umnožak u punom obliku, te konačno – pohraniti umnožak u drugu datoteku (u *rijetkom* obliku). Pojam *rijetka matrica* znači da je vrijednost većine elemenata 0, pa se ona podrazumijeva, a navode se samo vrijednosti elemenata matrice različitih od 0. Zapis matrica u datoteci (u rijetkom obliku) neka bude sljedeći:

- u prvom retku zapisa navode se broj redaka i broj stupaca cijele matrice;
- slijedi niz redaka – svaki redak definira jedan (različit od 0) element matrice, u obliku: redak stupac vrijednost, pri čemu je vrijednost realan broj, a indeksiranje redaka i stupaca kreće od 0;
- kraj zapisa pojedine matrice označava prazni redak;
- nakon praznog retka slijedi zapis druge matrice u istom obliku.

Primjer zapisa dvije rijetke matrice (u istoj datoteci):

```
2 3
0 0 2
0 2 -1
1 2 2

3 2
0 0 3
0 1 -1
1 0 2
2 0 -1
```

Matrice u programu pohraniti u obliku rječnika. Učitavanje pojedine matrice, ispis, pohranu rezultata, kao i množenje matrica, ostvariti potprogramima. Provjeravati ispravnost zapisa matrica u potprogramu za učitavanje, te usklađenost dimenzija u potprogramu za množenje.

Pri pozivu skripte navode se imena datoteke s matricama i datoteke u koju se zapisuje rezultat. Navodi se puni put (datoteke ne moraju biti u tekućem direktoriju). Primjer poziva skripte:

```
$ python zadatak1.py ../TestPrimjeri/matrice4.txt rezultat.txt

A:
  2.00   0.00  -1.00
  0.00   0.00   2.00

B:
  3.00  -1.00
  2.00   0.00
 -1.00   0.00

A*B:
  7.00  -2.00
 -2.00   0.00
```

```
Rezultat zapisan u datoteku rezultat.txt
2 2
0 0 7.00
0 1 -2.00
1 0 -2.00
```

Uputa za upload: Skriptu nazvati `zadatak1.py`.

Zadatak 2

U tekstualnoj datoteci pohranjeni su podaci koje je generirao program za poravnavanje slika (tj. to nije vaš zadatak). Program generira niz hipoteza, i za svaku hipotezu generira niz realnih brojeva koji predstavljaju udaljenosti točaka koje se uparuju. Svaki od generiranih nizova (realnih) brojeva, koji odgovara jednoj hipotezi, zapisan je u zasebnom retku. Brojevi su odvojeni prazninama. Duljina tih nizova je proizvoljna, ali svi nizovi u datoteci imaju jednak broj članova. Primjer takvog zapisa:

```
5.6 1.5 0.5 3.7 4.6 9.3 6.6 8.3 2.5 3.2 9.7 4.4 0.5 4.6 6.2 9.7 1.8 4.5 8.3 4.4
0.4 8.5 3.4 6.1 2.5 8.1 0.9 9.8 1.2 1.6 4.4 0.7 0.6 8.4 3.4 7.5 3.1 3.7 9.8 0.1
0.9 4.1 5.1 7.4 9.2 0.5 5.2 0.7 4.3 8.2 9.7 8.8 9.2 5.3 7.1 5.4 6.7 7.5 1.1 6.2
  ...
```

Za svaki takav niz brojeva izračunava se jedinstvena mjera – HD (*parcijalna usmjerena Hausdorffova udaljenost*), koja se dobije tako da se brojevi sortiraju rastućim redoslijedom, a zatim se uzme jedan od brojeva u nizu. Redni broj elementa niza koji se uzima kao mjera određen je parametrom Q (*kvantil*), koji se izražava kao postotak. Ako je primjerice u jednom nizu 100 brojeva, a parametar Q je 0.4, kao mjera HD uzima se 40. vrijednost u (sortiranom) nizu.

Vaš zadatak: Potrebno je napisati skriptu koja će na temelju ulazne datoteke koja se zadaje kao argument naredbenog retka generirati tablicu, u kojoj će pojedini redak odgovarati hipotezi, a u njemu će biti navedene vrijednosti mjere HD za različite vrijednosti parametra HD (korak neka bude `10%`, odnosno `0.1`)

Ispis treba biti kao u primjeru (prvi zapis u retku je redni broj hipoteze, tj. odgovarajućeg retka ulazne datoteke, a prvi redak je zaglavlje tablice):

```
$ python zadatak2.py ../TestPrimjeri/hipoteze2.txt

Hyp#Q10#Q20#Q30#Q40#Q50#Q60#Q70#Q80#Q90
001#0.5#1.8#3.2#4.4#4.5#4.6#6.2#8.3#9.3
002#0.4#0.7#1.2#2.5#3.4#3.7#6.1#8.1#8.5
003#0.7#1.1#4.3#5.2#5.4#6.7#7.4#8.2#9.2
  ...
```

Uputa za upload: Skriptu nazvati `zadatak2.py`.

Zadatak 3

Napisati skriptu koja će objediniti podatke iz niza datoteka i ispisati skupne podatke. Radi se o podacima o studenatima i evidenciji njihovih bodova na laboratorijskim vježbama. U jednoj datoteci – `studenti.txt` pohranjeni su podaci o studentima, pri čemu je u svakom retku podatak o jednom studentu: njegov matični broj, a zatim prezime i ime (polja su odvojena razmacima). Datoteke s podacima o bodovima s laboratorijskih vježbi pohranjene su u istom kazalu, a *ime svake datoteke* sadrži podatak o rednom broju vježbe i o grupi. Npr. `Lab_03_g08.txt` sadrži podatke o grupi 08 koja je radila 3. laboratorijsku vježbu. U datoteci je za svakog studenta koji je obavio vježbu zapisan matični broj i ostvareni broj bodova. Kazalo u kojem se nalaze podaci studentima i bodovima zadaje se kao argument pri pozivu skripte.

Konačni ispis treba biti kao u primjeru. Ako student nije odradio vježbu (nije evidentiran ni na jednom od popisa za tu vježbu), ispisuje se znak minus.

```
$ python zadatak3.py ../TestPrimjeri/zad3test

JMBAG         Prezime, Ime            L1      L2      L3
0036494111    Antić, Ante             1.0     2.0     3.0
0036489222    Ivanović, Ivana         2.5     2.5     1.0
0036492333    Perić, Petar            3.0     -       2.5
```

Skripta treba podatke spremati u odgovarajući rječnik, te treba provjeravati da ne dođe do prepisivanja podataka (npr. student se nalazi na dva popisa vezana za istu laboratorijsku vježbu – treba generirati upozorenje).

Uputa: za dohvat imena datoteka u kazalu može se koristiti standardni modul `os`. Funkcija `os.listdir(path)` vraća listu svih datoteka u kazalu `path`.

Uputa za upload: Skriptu nazvati `zadatak3.py`.

Zadatak 4

Napišite Python skriptu koja će kao argument naredbenog retka primiti znakovni niz koji sadrži adresu *web* stranice kojoj želimo pristupiti. Skripta treba otvoriti i učitati zadanu *web* stranicu, ispisati je na ekranu, a zatim obaviti niz pretraživanja u toj stranici.

- pronaći i izlistati sve linkove na druge stranice (ne kompliciratis parsiranjem HTML stranice, dovoljno je tražiti zapise oblika `href="url"`);
- napraviti listu svih *hostova* kojima se sa stranice može pristupiti (bez ponavljanja);
- za svaki *host* odrediti broj referenciranja u razmatranoj stranici;
- pronaći sve *e-mail* adrese u toj stranici;
- prebrojati linkove na slike (`<img src="url" ... >`).

Uputa: Koristiti standardne module `urllib.request` i `re`. Zadana stranica može se otvoriti pozivom funkcije `urllib.urlopen()`, pri čemu se URL zadaje u obliku znakovnog niza. Nakon

otvaranja, sadržaj stranice može se čitati metodom `read()`. Pritom treba voditi računa da se čitanjem ne dobiva znakovni niz već niz bajtova. Primjer:

```
import urllib.request

stranica = urllib.request.urlopen("http://www.python.org")
mybytes = stranica.read()
mystr = mybytes.decode("utf8")
print(mystr)
```

Nakon što dobijemo sadržaj web stranice u obliku znakovnog niza, pretraživanje regularnim izrazima može se postići pozivom funkcije `re.findall('regularni_izraz', string)`.

Uputa za upload: Skriptu nazvati `zadatak4.py`.