Skriptni jezici – zadaci za 3. ciklus laboratorijskih vježbi

svibanj 2020.

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 pristupiti ispitivanju laboratorijske vježbe u za to predviđenom 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. nakon kratke provjere znanja student preuzima programski zadatak koji treba riješiti u zadanom vremenu (45 minuta). Zadatak se rješava na računalu, a rješenje se 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!

Priprema se ocjenjuje s najviše 3 boda, kratka provjera znanja s 3 boda te programski zadatak s 4 boda.

1.2 Resursi u laboratoriju

Ove godine se ispitivanje provodi on-line korištenjem sustava *moodle* i *Ferko*.

Zadatak 1

Napisati skriptu koja će iz (jedne) datoteke učitati dvije *rijetke* matrice, ispisati njihov umnožak u potpunom obliku, i konačno – pohraniti ga u drugu datoteku u *rijetkom* obliku. Zapis matrica neka bude sljedeći:

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

Primjer zapisa dvije rijetke matrice:

```
4 5
1 1 12.567
2 1 -3.23
3 4 1

5 6
1 2 2.67
2 3 3
4 4 -3.14
4 5 7.89
```

Matrice u programu pohraniti u obliku rječnika. Učitavanje pojedine matrice ostvariti kao potprogram. Provjeravati ispravnost zapisa matrica, te usklađenost dimenzija za operaciju množenja i ispisivati odgovarajuće poruke o pogrešci.

Uputa za upload: Skriptu nazvati `zadatak1.py`. Skripta kao argumente naredbenog retka prima imena ulazne i izlazne datoteke (ime može sadržavati i cijeli put do datoteke, tj. datoteka se ne mora nalaziti u istom direktoriju kao skripta), a na standardni izlaz ispisuje rezultat množenja matrica zapisanih u ulaznoj datoteci. Za implementaciju množenja matrica nije dozvoljeno korištenje posebnih biblioteka.

Zadatak 2

U datoteci, koja se zadaje kao argument naredbenog retka, 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. Primjer takvog zapisa:

```
1.23 2.57 7.56 12.21 3.22 2.12 1.07 ....
2.56 6.38 4.11 3.28 13.56 7.32 12.77 ....
```

Za svaki takav niz brojeva izračunava se jedinstvena mjera – HD (tzv. *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 (čije ime 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 u sljedećem obliku (prvi zapis u retku je redni broj hipoteze, tj. odgovarajućeg retka ulazne datoteke, a prvi redak je zaglavlje tablice):

```
Hyp#Q10#Q20#Q30# ... #Q90
001#1.25#1.75#2.21# ... #21.34
```

Uputa za upload: Skriptu nazvati `zadatak2.py`. Ime ulazne datoteke (uključujući put) navodi se kao argument naredbenog retka. Skripta treba provjeriti broj argumenata i čitljivost zadane datoteke te u slučaju pogreške ispisati odgovarajuću poruku. Rezultat se ispisuje na standardni izlaz.

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. 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 (kao dvoznamenkasti broj): npr. `Lab03_g08.txt`. U datoteci je za svakog studenta koji je obavio vježbu zapisan matični broj i ostvareni broj bodova (broj s pomičnom točkom). Konačni ispis na standardni izlaz treba biti u obliku:

```
JMBAG    Prezime, Ime   L1    L2    L3  ...
0036...  Antic, Ante    4.0   2.5   6.0 ...
```

Skripta treba podatke spremati u odgovarajući rječnik, pri čemu treba provjeravati da ne dođe do prepisivanja podataka (npr. ako se student nalazi na dva popisa vezana za istu laboratorijsku vježbu, treba ispisati upozorenje i prekinuti učitavanje).

Uputa: za dohvat imena datoteka u kazalu može se koristiti standardni modul `os`. Funkcija `os.listdir(".")` vraća listu svih datoteka u tekućem kazalu. Za detalje proučiti dokumentaciju.

Uputa za upload: Skriptu nazvati `zadatak3.py`. Ime direktorija (uključujući put) u kojem se nalaze ulazne datoteke navodi se kao argument naredbenog retka.

Zadatak 4

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

- pronaći i ispisati sve linkove na druge stranice (ne komplicirati s 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.request.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`. Kao argument naredbenog retka navodi se URL web stranice kojoj treba pristupiti.