Laboratorijska vežba 2 - Python

Napomena: Svaki zadatak realizovati u okviru jedne funkcije.

1. Korišćenjem programskog jezika Python, napisati funkciju **poredak**, koja 2 liste brojeva objedinjuje u jednu listu koja se sastoji od tuple objekata koji imaju tri elementa. Prvi element rezultujuće kolekcije je element prve liste, drugi je element druge liste a treći ma vredost 'Jeste' ukoliko je element druge liste dva puta veći od elementa prve liste, odnosno 'Nije' ukoliko nije. Dužina liste je dimenzija duže od dve ulazne liste. N-ti tuple podatak rezultujuće kolekcije čine n-ti brojevi iz obe ulazne liste, gde na prvoj poziciji treba da se nađe manji, a na drugoj veći broj iz obe liste. Kraću ulaznu listu dopuniti sa kraja brojem 0, dok dužine obe liste ne budu iste. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

   ```
   Primer: poredak ([1, 7, 2, 4], [2, 5, 2]) = [(1, 2, 'Jeste'), (7, 5, 'Nije'),
           (2, 2, 'Nije'), (4, 0, 'Nije')]
   ```

2. Korišćenjem programskog jezika Python, napisati funkciju **spojidict**, koja 2 liste obejkata objedinjuje u listu sa elementima tipa dictionary. Dužina liste je dimenzija duže od dve ulazne liste. N-ti dictionary element rezultujuće kolekcije čine n-ti objekti iz obe ulazne liste, gde se na prvoj poziciji nalazi element prve liste uparen sa ključem `'prvi'`, a na drugoj poziciji element druge liste uparen sa ključem `'drugi'`. Kraću ulaznu listu dopuniti sa `'-'`, dok dužine obe liste ne budu iste. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

   ```
   Primer: spojidict([1, 7, 2, 4], [2, 5, 2]) = [{'prvi': 1, 'drugi': 2},
           {'prvi': 7, 'drugi': 5}, {'prvi': 2, 'drugi': 2}, {'prvi': 4, 'drugi': '-'}]
   ```

3. Korišćenjem programskog jezika Python, napisati funkciju **spoji**, koja 2 liste brojeva objedinjuje u jednu listu koja se sastoji od tuple objekata koji imaju tri elementa. Prvi element rezultujuće kolekcije je element prve liste, drugi je element druge liste a treći je zbir elemenata. Dužina liste je dimenzija duže od dve ulazne liste. N-ti tuple podatak rezultujuće kolekcije čine n-ti brojevi iz obe ulazne liste, gde na prvoj poziciji treba da se nađe manji, a na drugoj veći broj iz obe liste. Kraću ulaznu listu dopuniti sa kraja brojem 0, dok dužine obe liste ne budu iste. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

   ```
   Primer: spoji([1, 7, 2, 4], [2, 5, 2]) = [(1, 2, 3), (5, 7, 12), (2, 2, 4), (0, 4, 4)]
   ```

4. Korišćenjem programskog jezika Python, napisati funkciju **suma**, koja vraća sumu svih elemenata u listi brojeva i svim njenim podlistama. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi) i funkcije sum.

   ```
   Primer: suma([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) = 45
   ```

5. Korišćenjem programskog jezika Python, napisati funkciju **proizvod**, koja računa proizvod liste podlisti (A) i liste brojeva (B). Smatrati da je broj podlisti u listi A jednak dužini liste B. Funkcija vraća listu koja ima onoliko elemenata koliko je dužina ulaznih listi. N-ti element izlazne liste predstavlja sumu svih elemenata N-te podliste liste A koji u prethodno pomnoženi N-tim elementom u liste B. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi) i funkcije sum.

   ```
   Primer: proizvod([[1, 2, 3], [4, 5, 6], [7, 8, 9]], [1, 2, 3]) = [6, 30, 72]
   ```

6. Korišćenjem programskog jezika Python, napisati funkciju **objedini**, koja 2 liste brojeva objedinjuje u jednu listu koja se sastoji od parova brojeva (tuple). Dužina liste treba da je dimenzija duže od dve ulazne liste. N-ti tuple podatak rezultujuće kolekcije čine n-ti brojevi iz obe ulazne liste, gde na prvoj poziciji treba da se nađe manji, a na drugoj veći broj iz obe liste. Kraću ulaznu listu dopuniti sa kraja brojem 0, dok dužine obe liste ne budu iste. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

   `Primer: objedini([1, 7, 2, 4, 5], [2, 5, 2]) = [(1, 2), (5, 7), (2, 2), (0, 4), (0, 5)]`

7. Korišćenjem programskog jezika Python, napisati funkciju **objedini**, koja od liste tuple objekata kreira dictionary. Prvi element svakog tuple objekta postaje ključ rečnika, dok sve ostale vrednosti postaju vrednost (lista vrednosti). Ukoliko tuple podatak ima manje od 2 vrednosti, na mesto vrednosti postaviti None. Ukoliko ključ već postoji u rečniku, prepisati njegovu vrednost novom. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

   ```
   Primer: objedini([(1), (3, 4, 5), (7), (1, 4, 5), (6, 2, 1, 3)]) = { 1: [4, 5], 3: [4, 5],
           7: None, 6: [2, 1, 3] }
   ```

8. Korišćenjem programskog jezika Python, napisati funkciju **izracunaj**, koja u listi koja može da se sastoji od brojeva i podlisti, na n-tom mestu u rezultujućem nizu upisuje broj sa n-te pozicije iz ulaznog niza, a ukoliko se radi o podlisti, zamenjuje je proizvodom svih brojeva u podlisti. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

   `Primer: izracunaj([1, 5, [1, 5, 3], [4, 2], 2, [6, 3]]) = [1, 5, 15, 8, 2, 18]`

9. Korišćenjem programskog jezika Python, napisati funkciju **zamena**, koja u listi brojeva, brojeve manje od broja x, koji se prosleđuje kao argument, zamenjuje zbirom svih vrednosti ulazne liste, koje imaju indeks veći od indeksa broja koji se obrađuje. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

   `Primer: zamena([1, 7, 5, 4, 9, 1, 2, 7], 5) = [35, 7, 5, 19, 9, 9, 7, 7]`

10. Korišćenjem programskog jezika Python, napisati funkciju **stepen**, koja svaki par dva broja u ulaznoj listi (x, y), transformiše u x^y. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

    `Primer: stepen([1, 5, 2, 6, 1, 6, 3, 2, 9]) = [1, 25, 64, 6, 1, 216, 9, 512]`

11. Korišćenjem programskog jezika Python, napisati funkciju **proizvod**, koja vraća proizvod svih elemenata u listi brojeva i svim njenim podlistama. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi) i funkcije prod.

    `Primer: proizvod([[1, 3, 5], [2, 4, 6], [1, 2, 3]]) = 4320`

12. Korišćenjem programskog jezika Python, napisati funkciju **izracunaj**, koja u listi koja se sastoji od brojeva i podlisti, menja svaki broj njegovim kvadratom, dok listu zamenjuje zbirom kvadrata brojeva koji je čine. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

    `Primer: izracunaj([2, 4, [1, 2, 3], [4, 2], 2, [9, 5]]) = [4, 16, 14, 20, 4, 106]`

13. Korišćenjem programskog jezika Python, napisati funkciju **skupi**, koja kreira novu listu tako da svaka dva susedna elementa liste ciji su elementi iskljucivo podliste zamenjuje podlistom koja sadrži zbir elemeata na odgovarajućim pozicijama. Elemente koji nedostaju tretirati kao nule. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

    `Primer: skupi([[1, 3, 5], [2, 4, 6], [1, 2]]) = [[3, 7, 11], [3, 6, 6]]`

14. Korišćenjem programskog jezika Python, napisati funkciju **suma**, koja prvenstveno određuje proizvod elemenata u svakoj podlisti unutar prosleđene liste, a zatim sumu tako dobijenih elemenata. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi) i funkcije sum i prod.

    `Primer: suma([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) = 630`

15. Korišćenjem programskog jezika Python, napisati funkciju **promeni**, koja u listi brojeva, brojeve veće ili jednake broju x, koji se prosleđuje kao argument, umanjuje za x, dok brojeve manje od x uvećava za x. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

    `Primer: promeni([7, 1, 3, 5, 6, 2], 3) = [4, 4, 6, 2, 3, 5]`

16. Korišćenjem programskog jezika Python, napisati funkciju **broj**, koja na osnovu heksadekadnog broja koji predstavlja boju na ulazu u formatu #RGB, određuje integer reprezentaciju tog broja, koja se dobija sledećim obrascem: (R * 65536) + (G * 256) + B. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

    Napomena: Broj N je moguće prevesti iz baze B u bazu 10 korišćenjem funkcije int(N, B)

    `Primer: broj("#FA0EA0") = 16387744`

17. Korišćenjem programskog jezika Python, napisati funkciju **tekst**, koja karaktere u tekstu koji su van opsega malih, velikih slova i cifara (65-90 velika i 97-122 mala slova, 48-57 cifre), zamenjuje unicode vrednošću \ubbbb, gde bbbb predstavlja četvorocifrenu unicode reprezentaciju slova koje se menja. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

    Napomena: Za preuzimanje unicode reprezentacije slova moguće je koristiti funkciju ord, dok se za upisivanje određenog broja nula ispred broja koristi funkcija zfill(brojNula). Prevođenje broja iz dekadnog u heksadekadni se vrši korišćenjem funkcije hex.

    `Primer: tekst("Otpornost 10Ω.") = 'Otpornost\\u002010\\u03A9\\u002E'`

18. Korišćenjem programskog jezika Python, napisati funkciju **brojevi**, koja iz stringa izvlači listu svih brojeva koji se u njemu nalaze. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

    `Primer: brojevi("42+10=52;10*10=100") = [ 42, 10, 52, 10, 10, 100 ]`

19. Korišćenjem programskog jezika Python, napisati funkciju **brojanje**, koja broji koliko karaktera se ponavlja uzastopno više puta u prosleđenom stringu. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

    `Primer: izbaci("aatesttovi") = 2`

20. Korišćenjem programskog jezika Python, napisati funkciju **izracunaj**, koja računa vrednost elementa u rezultujućem nizu korišćenjem 3 uzastopna elementa u nizu koji je prosleđen, korišćenjem funkcije koja se takođe prosleđuje kao parametar. Zabranjeno je korišćenje petlji (osim u comprehension sintaksi).

    `Primer: izracunaj([2, 5, 1, 6, 7], lambda x, y, z: x + y * z) = [7, 11, 43]`