1. Napišite funkciju pod nazivom pravougaonik koja prihvata dva cela broja m i n kao argumente i štampa m × n boks koji se sastoji od zvezdica. Dole je prikazan izlaz funkcije pravougaonik(2,4)

**\*\*\*\***

**\*\*\*\***

1. Napišite funkciju pod nazivom dodaj\_uzvicnik koja prihvata listu stringova i dodaje znak uzvika (!) na kraj svakog od stringova iz liste. Program treba da modifikuje originalnu listu i ne vraća ništa.
2. Napišite istu funkciju ali da ona ne modifikuje originalnu listu već da vraća novu listu.
3. Napišite funkciju suma\_cifara koja prihvata jedan ceo broj num i vraća sumu njegovih cifara.
4. Digitalni koren nekog broja n se dobija na sledeći način: Saberemo cifre broja n da dobijemo novi broj. Saberimo cifre tog broja da dobijemo novi broj. Nastavljamo ovaj proces sve dok broj dobijen sabiranjem cifara ne bude jednocifren. Taj broj je digitalni koren. Na primer, ako je n=45893, posle sabiranja njegovih cifara imamo 4+5+8+9+3 = 29. Saberemo cifre broja 29 pa dobijemo 2+9=11. Sada saberemo cifre broja 11i dobijemo 1+1=2. Pošto je 2 jednocifren, on je digitalni koren broja 45893. Napišite funkciju koja vraća digitalni koren celog broja n.
5. Napišite funkciju prva\_razlika koja prihvata dva stringa i vraća prvu poziciju na kojoj se stringovi razlikuju. Ako su stringovi identični funkcija vraća -1.
6. Napišite funkciju binom koja prihvata dva cela broja n i k i vraća binomijalni koeficijent n C k. Definicija: n C k = n!/(k!(n-k)!).
7. Napišite funkciju koja prihvata ceo broj n i vraća slučajan ceo broj sa tačno n cifara. Na primer ako je n jednako 3,tada bi 125 i 593 bili validne vrednosti za vraćanje iz funkcije, ali bi 093 bilo neispravno jer je to u stvari broj 93, koji je dvocifren.
8. Napišite funkciju broj\_faktora koja prihvata ceo broj i vraća koliko faktora (delioca) ima taj broj.
9. Napišite funkciju faktori koja prihvata ceo broj i vraća listu njegovih faktora.
10. Napišite funkciju najbliži koja prihvata listu brojeva L i broj n i vraća najveći element iz L koji nije veći od n. Na primer ako je L=[1,6,3,9,11] a n=8, tada funkcija treba da vrati 6, zato što je 6 najbliža vrednost broju 8 u L, a koja nije veća od 8.
11. Napišite funkciju poklapanje koja prihvata dva stringa i vraća koliko ima poklapanja između dva stringa. Poklapanje je slučaj kada stringovi imaju isto slovo na istoj poziciji. Na primer, ‘python’ i ‘path’ se poklapaju na prvom, trećem i četvrtom slovu, pa funkcija treba da vrati 3.
12. Setimo se da ako je s neki string tada s.find(‘a’) pronalazi poziciju (indeks) prvog slova a u s. Problem je što ona ne pronalazi sve pozicije slova a. Napišite program findall koji za dati string i za dato slovo, vraća listu koja sadrži sve pozicije na kojima se nalazi to slovo u stringu. Funkcija treba da vrati praznu listu ako se slovo ne nalazi u stringu.
13. Napišite funkciju promeni\_velicinu koja za dati string vraća string u kojem su sva velika slova zamenjena malim i obratno.
14. Napišite funkciju is\_sorted koja za datu listu vraća True ako je lista sortirana, a False u suprotnom slučaju.
15. Napišite funkciju root koja za dati broj x i ceo broj n vraća x 1/n. Pri definiciji funkcije postavite podrazumevajuću (default) vrednost za n da bude 2.
16. Napišite funkciju one\_away koja prihvata dva stringa i vraća True ako su stringovi iste dužine i razlikuju se samo u jednom slovu, kao na primer bike/hike ili water/wafer.
17. Napišite funkciju primes koja za dati broj n vraća listu prvih n prostih brojeva. Neka podrazumevajuća vrednost za n bude 100.

* Modifikujte gornju funkciju tako da ima i jedan opcioni argument pod nazivom start koji omogućava da lista startuje sa vrednošću većom od 2.Funkacija treba da vrati listu prvih n prostih brojeva koji su veći ili jednaki vrednosti varijable start. Podrazumevajuća vrednost za start treba da bude 2.

1. Napišite funkciju koja za dati ceo broj, vraća naziv toga broja na engleskom (srpskom) jeziku. Na primer, za broj 123456 kao argument funkcija treba da vrati „ one hundred twenty-three thousand, four hundred fifty-six“.
2. Napišite funkciju merge koja prihvata dve sortirane liste (istih ili različitih dužina), i spaja (merdžuje) ih u jednu sortiranu listu.

* Uradite isto korišćenjem sort metode.
* Uradite ovo bez korišćenja sort metode.

1. U jednom od prethodnih poglavlja smo za proveru da li se neka reč nalazi u listi reči koristili naredbu

**if** rec **in** lista\_reci:

Ovo je nažalost veoma sporo, ali postoji brži način koji se zove binarno pretraživanje (binary search). Da implementiramo binarno pretraživanje u nekoj funkciji, počinjemo sa poređenjem varijable rec sa rečju iz sredine liste lista\_reci. Ako su iste završili smo traženje i možemo da vratimo True. S druge strane ako rec dolazi pre reči iz sredine, onda traženje treba da nastavimo u prvoj polovini liste. Ako rec dolazi posle sredine, tada traženje nastavljamo u drugoj polovini liste. Ponavljamo ovaj proces sve dok ili ne pronađemo traženu reč ili nemamo više gde da tražimo ,u kom slučaju vraćamo False. Operatori < i > mogu biti iskorišćeni za alfabetsko poređenje stringova.

1. (Tic-tac-toe), ili kako se to kod nas zove Iks-Oks, tabla može biti predstavljena jednom 3 × 3 dvodimenzionom listom, sa nulama za prazna polja, i X i O za popunjena polja.

* Napišite funkciju koja prihvata Iks-Oks listu i na slučajno odabranu poziciju (koja mora biti prazna) postavlja O.
* Napišite funkciju koja prihvata Iks-Oks listu i proverava da li je neko ostvario pobedu. Funkcija vraća True ako postoji pobednik, u suprotnom vraća False.

1. Napišite funkciju koja prihvata 9 × 9 potencijalno rešeni Sudoku i vraća True ako je rešenje korektno, a False ako ima grešku. Sudoku je tačno rešen ako nema ponovljenih brojeva ni u jednom redu ili koloni, kao ni u jednom od devet “blokova.”