

Demonstrature 6 - dodatni zadaci

Napomena 1. Ako u zadatku nije drugačije navedeno, onda nije potrebno provjeravati ispravnost inputa.

Zadatak 1. Implementirajte funkciju *listToInt* koja prima listu znamenaka te vraća prirodan broj s istim znamenkama. Npr. za listu $[1, 8, 2, 5]$ funkcija treba vratiti rezultat 1825. Ukoliko lista sadrži više od 18 elemenata ili je prazna, vratite -1.

Zadatak 2. Implementirajte funkciju *charToInt* koja prima znamenku u obliku podatka tipa `Char`, a vraća njenu brojčanu vrijednost.

Npr. `charToInt '4' = 4`.

Zadatak 3. Implementirajte funkciju *strToInt* koja prima nenegativan cijeli broj manji od 10^{19} reprezentiran stringom te vraća dani broj, ali kao `Int`. Npr. za input "1825" funkcija treba vratiti 1825. Ukoliko funkcija primi prazan string, vratite -1. Nije dozvoljeno korištenje funkcije `read`.

Zadatak 4. Implementirajte funkciju *update* koja prima listu tipa `a`, cijeli broj k te argument n tipa `a`. Funkcija postavlja k -ti element dane liste na vrijednost n , a ostatak liste ne dira. Pretpostavite da lista ima barem $k + 1$ elemenata.

Zadatak 5. Implementirajte funkciju *tabuStep* koja prima listu nenegativnih brojeva te nenegativne brojeve k i n . Funkcija radi sljedeće:

- k -ti element dane liste postavlja na vrijednost n ;
- sve pozitivne članove (osim eventualno k -tog) u listi smanjuje za 1;
- sve nule u listi (osim eventualno one na k -tom mjestu) ne dira.

Pretpostavite da lista ima barem $k + 1$ elemenata te da indeksi kreću od 0.

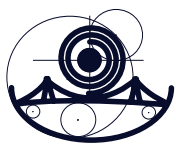
Zadatak 6. Implementirajte rekurzivnu funkciju *zbrojFoldl* koja prima listu cijelih brojeva i cijeli broj n te vraća zbroj svih članova dane liste i broja n , ali na način da se rezultat svakog poziva funkcije sprema u n .

Npr. `zbrojFoldl [7,5,1,2] 4 = zbrojFoldl [5,1,2] 11`

Zadatak 7. Implementirajte rekurzivnu funkciju *zbrojScanl* koja radi isto što i funkcija *zbrojFoldl*, ali sprema rezultat svakog poziva funkcije u listu.

Npr. za listu $[7, 5, 1, 2]$ i $n = 4$, funkcija treba vratiti $[4, 11, 16, 17, 19]$.

Zadatak 8. Implementirajte funkciju *decN* koja prima listu cijelih brojeva i nenegativan cijeli broj n , a smanjuje prvih n članova dane liste za 1. Pretpostavite da lista sadrži barem n članova. Npr. za listu $[3, 3, 3, 3, 2, 2, 2, 1]$ i $n = 5$, funkcija treba vratiti $[2, 2, 2, 2, 1, 2, 2, 1]$.



Zadatak 9. Implementirajte funkciju *is_graphical* koja prima konačnu, silazno sortiranu listu cijelih brojeva $[a_1, a_2, \dots, a_n]$ duljine n te provodi sljedeći algoritam:

- Ako je lista prazna, vrati *False*;
- Ako je lista neprazna i svi njeni članovi su jednaki 0, vrati *True*;
- Ako lista sadrži barem jedan negativan broj, vrati *False*;
- Ako je lista neprazna, nema negativnih članova i vrijedi $a_1 \geq n$, vrati *False*;
- Ukoliko niti jedan od prethodna 4 uvjeta nije ispunjen, izbaci a_1 iz liste, smanji sljedećih a_1 članova za 1, sortiraj rezultat silazno te pozovi funkciju *is_graphical* na dobivenom rezultatu.

Zadatak 10. Implementirajte funkciju *dotProd* koja računa skalarni produkt dvaju vektora koristeći funkciju *zip* u kombinaciji s generatorom liste ili funkcijom *map*.

Zadatak 11. Implementirajte funkciju *derive* koja prima polinom s cjelobrojnim koeficijentima te ga derivira. Ako je polinom zadan s

$$p(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n, \quad n \in \mathbb{N}_0,$$

onda ga zapišite kao nepraznu listu $[a_0, a_1, a_2, \dots, a_n]$.

Zadatak 12. Implementirajte funkciju *scale* koja prima realan broj α i vektor v te računa vektor $\alpha \cdot v$ koristeći funkciju *map*.

Zadatak 13. Implementirajte funkciju *is_zero* koja prima realan vektor v te vraća *True* akko je v nul-vektor. Koristite funkciju *filter*.

Zadatak 14. Implementirajte funkciju *checkOrth* koja prima dva realna vektora te vraća *True* akko su oni okomiti.

Zadatak 15. Implementirajte funkciju *norm* koja računa normu realnog vektora.

Zadatak 16. Implementirajte funkciju *normalize* koja normalizira realan vektor. Nul-vektor nije moguće normalizirati te u tom slučaju vratite nul-vektor.

Zadatak 17. Implementirajte funkciju *allNdim* koja prima listu realnih vektora i prirodan broj n te provjerava jesu li svi vektori u danoj listi dimenzije n .

Zadatak 18. Implementirajte funkciju *checkOrthBase* koja prima skup od n vektora iz \mathbb{R}^n te provjerava tvore li dani vektori ortogonalnu bazu za \mathbb{R}^n .

Napomena: skup od n vektora iz \mathbb{R}^n tvori ortogonalnu bazu za \mathbb{R}^n akko se među njima ne nalazi nul-vektor te su svi međusobno okomiti. Osim ova dva uvjeta, treba provjeriti i jesu li svi vektori dimenzije n , gdje je n broj vektora u danom skupu.