1.1 Teorijski zadaci

Zadatak 1.
Na predavanju ste se upoznali s algoritmom za sortiranje umetanjem. U ovom zadatku trebate analizirati algoritam sortiranja u engleskoj literaturi poznat kao *Bubble Sort* koji iterativno zamjenjuje mjesta susjednim elementima u polju. U nastavku slijedi pseudokod ovog algoritma.

BubbleSort(A)
1 **for** $i \leftarrow 1$ **to** $A.length - 1$
2 **do for** $j \leftarrow A.length$ **downto** $i + 1$
3 **do if** $A[j] < A[j-1]$
4 **do** zamijeni $A[j]$ i $A[j-1]$
5 **return** A

a) **[2 boda]** Neka je A' polje koje dobijemo izvršavanjem BubbleSort algoritma na polju A, tj. pozivom BubbleSort(A). Kako biste pokazali da je BubbleSort korektan algoritam, potrebno je dokazati da nakon izvršavanja vrijedi:

$$A'[1] \leq A'[1] \leq \ldots A'[n],$$

gdje je $n = A.length$. Iskažite i dokažite invarijantu for-petlje u linijama 2-4. Dokaz treba slijediti ideju s predavanja.

b) **[2 boda]** Koristeći terminaciju invarijante petlje koja je dokazana pod a), najprije iskažite, a zatim i dokažite invarijantu for-petlje u linijama 1-4 koja će omogućiti dokaz tražene nejednakosti iz koje slijedi korektnost algoritma. Dokaz treba slijediti ideju s predavanja.

c) [1 bod] Analizirajte vrijeme izvršavanja BubbleSort algoritma u najgorem slučaju? Kakav je BubbleSort u odnosu na sortiranje umetanjem (eng. InsertionSort)?

Zadatak 2.
Za sljedeće funkcije pokažite da vrijede navedene tvrdnje.

a) $f(n) = \frac{1}{2}n^2 - 3n$, $f(n) \in \Theta(n^2)$

b) $g(n) = 10n\lg(n) + 5n^2 + 3$, $g(n) \in \mathcal{O}(n^2)$

Zadatak 3.
Metodom supstitucije pokažite da za rekurziju $T(n) = 4T(\frac{n}{2}) + n$, $T(1) = \Theta(1)$ vrijede sljedeće tvrdnje

a) $T(n) \in \mathcal{O}(n^3)$

b) $T(b) \in \mathcal{O}(n^2)$

Zadatak 4.
Metodom supstitucije pokažite da je rješenje rekurzije $T(n) = T(\lceil \frac{n}{2} \rceil) + 1$ zadano s $\mathcal{O}(\lg(n))$.

Zadatak 5.
Metodom rekurzivnog stabla odredite rješenje rekurzije $T(n) = 4T(\lfloor \frac{n}{2} \rfloor) + cn$

Zadatak 6.
Master metodom odredite rješenja sljedećih rekurzija

a) $T(n) = 7T(\frac{n}{2}) + n^2$

b) $T(n) = 2T(\frac{n}{2}) + n^3$

c) $T(n) = 16T(\frac{n}{4}) + n^2$

d) * $T(n) = 8T(\frac{n}{2}) + n^2$

e) * $T(n) = 2T(\frac{n}{3}) + \sqrt[8]{n} \cdot 3^{\lg n}$

Zadatak 7. *
Na predavanju smo dokazali da je vrijeme izvršavanja algoritma sortiranja spajanjem $\Theta(n \lg n)$, dok je vrijeme algoritma sortiranja umetanjem $\mathcal{O}(n^2)$,što prvi algoritam čini brži u najgorem slučaju. Iako smo vidjeli da asimptotska notacija ignorira konstante u vremenu izvršavanja, kada je u pitanju algoritam sortiranja umetanjem, male konstante ga u praksi mogu učiniti brzim na poljima male duljine. Stoga svakako ima smisla kombinirati algoritam sortiranja spajanjem s algoritmom sortiranja umetanjem na način da se umetanjem sortira polje dovoljno male duljine u rekurzivnim pozivima algoritma sortiranja spajanjem. Razmotrite modificiranu verziju sortiranja spajanjem u kojem n/k potpolja duljine k soritramo koristeći sorto+iranje umetanjem. Tako sortirana polja spajamo na isti način kako smo to učinili kod sortiranja spajanjem.

a) Pokažite da algoritam sortiranja umetanjem može sortirati n/k potpolja duljine k u $\mathcal{O}(nk)$ vremenu.

b) Pokažite kako spojiti sortirana potpolja duljine k u vremenu $\Theta(n \lg \frac{n}{k})$.

c) Pokažite da se modificirani algoritam izvršava u $\Theta(nk + n \lg(n/k))$. Ukoliko k promatrate kao funkciju od n, za koju najveću vrijednost parametra k će modificirani algoritam imati isto vrijeme izvršavanja kao i standardno sortiranje spajanjem, u terminima Θ-notacije

Zadatak 8.
Analizirajte vremensku složenost Karatsubinog algoritma za množenje polinoma f i g koji su oblika $f(x) = a_n x^n + ... + a_1 x + a_0$ i $g(x) = b_n x^n + ... + b_1 x + b_0$. Kao ulazni podatak algoritam prima prirodni broj n i koeficijente $a_n, ..., a_0, b_n, ..., b_0$. Iskažite i primjenom odgovarajuće metode dokažite vrijeme izvršavanja ovog algoritma u ovisnosti o veličini ulaznog podatka.

Zadatak 9.
Za polje $A =< a_1, a_2, ..., a_n >$ kažemo da je unimodalno ako postoji $k \in \mathcal{N}$ takav da vrijedi $a_1 < a_2 < ... < a_k$ i $a_k > a_{k+1} > ... > a_n$. Napišite algoritam koji će u $\mathcal{O}(\lg n)$ vremenu pronaći najveći element u unimodalnom polju. Objasnite dobivenu vremensku složenost algoritma.

Zadatak 10.
Objasnite kako biste implementirali red koristeći dva stoga. Za tako zadani red analizirajte vremensku složenost operacija INSERT i DELETE.

Zadatak 11.
Objasnite kako biste implementirali stog koristeći dva reda. Za tako zadani stog analizirajte vremensku složenost operacija INSERT i DELETE.

Zadatak 12.
Koristeći jednostruko povezanu listu, objasnite implementaciju

1. stoga
2. reda,

te analizirajte vremenske složenosti operacija operacija INSERT i DELETE.

Zadatak 13.
Pokažite da za bilo koji čvor u binarnom stablu pretraživanja vrijedi da ako ima dva djeteta tada njegov sljedbenik nema lijevo dijete i njegov prethodnik nema desno dijete.

Zadatak 14.
Neka je T binarno stablo pretraživanja s međusobno različitim vrijednostima čvorova tj. ključevima. Neka je x list tog stabla i neka je y njegov roditelj. Pokažite da je tada ključ od y ili najmanji ključ u T veći od ključa od x ili najveći ključ u T manji od ključa od x.

Zadatak 15. Neka je zadano stabo T. Najniži zajednički predak (*engl.*Lowest Common Ancestor - LCA) od dva čvora u i v se definira kao najniži čvor u stablu T koji ima v i w kao nasljednike (dopušteno je da čvor bude sam sebi nasljednik). Dajte efikasan algoritam za računanje LCA.

Zadatak 16.
Na predavanju je objašnjena BUILD-MAX-HEAP operacija i pokazana je

Zadatak 17. *
Pokažite da binarna hrpa od n elemenata ima visinu $\lg(n)$.

Zadatak 18.
Pokažite da u binarnoj hrpi od$n$ elemenata na visi h može biti najviše $\lceil \frac{n}{2^{h+1}} \rceil$ elemenata.

Zadatak 19.
Napišite pseudokod operacije HEAP-DELETE(H, x) koja iz minimalne binarne hrpe H briše čvor x u vremenu $\mathcal{O}(\lg(n))$ Pretpostavite da je binarna hrpa implementirana pomoću polja.

Zadatak 20.
Do sada smo se susretali s binarnim hrpama čiji elementi mogu imati najviše dvoje djece. Umjesto binarne hrpe, sada ćemo promatrati $d-$arnu hrpu čiji elementi mogu imati najviše d djece

- Kako biste pomoću polja reprezentirali maksimalnu d-arnu hrpu? Za svaki element takve hrpe na poziciji k u polju odredite indekse njegove djeca i roditelja.
- Odredite visinu maksimalne d-arne hrpe koja ima n elemenata kao funkciju od n i d.

Zadatak 21.
Neka je T AVL stablo. Skicirajte T nakon izvršavanja sljedećih operacija. Vodite računa o AVL svojstvu stabla T i skicirajte izvršavanje rotacija kada je to potrebno.

- INSERT(15), INSERT(14), INSERT(17), INSERT(26), INSERT(20), DELETE(14), INSERT(23), INSERT(24), INSERT(1), INSERT(5), INSERT(3), DELETE(16), DELETE(1)

Zadatak 22.
Kupujete novo računalo i cijene onih računala koja vam se sviđaju spremate u AVL stablo. Dostupne su vam standardne operacije SEARCH i INSERT, no osim toga želite imati dostupnu operaciju IN_RANGE(A,B) koja vraća broj računala čija je cijena u rasponu od a do b.

a) Navedite vremensku složenost operacija SEARCH i INSERT

b) Potvrdite ili opovrgnite sljedeću tvrdnju: ubacivanje nove cijene u AVL stablo koje već ima n cijena zahtjeva barem $\Omega(\lg(n))$ rotacija. **Dodatno**, razmislite koji je **najveći** broj rotacija potreban pri operaciji INSERT i DELETE? Pomoć - prije dodavanja stablo je balansirano, slika

c) Napišite u pseudojeziku kako biste implementirali operaciju IN_RANGE(A,B) . Koja je njena vremenska složenost?

Proširimo AVL stablo tako da svaki čvor uz vrijednost (key) i pokazivače, dodatno sadrži informaciju o veličini podstabla ukorijenjenog u njemu.

d) Koristeći dodatno spremljenu informaciju za svaki čvor, opišite kako biste implementirali IN_RANGE(A,B) operaciju. Objasnite njenu vremensku složenost.

e) Objasnite kako biste implementirali INSERT algoritam tako da pri svakom dodavanju novog elementra ažurirate informaciju o veličini podstabla?

f) Objasnite kako biste modificirali rotacije tako da sačuvate informaciju o veličini podstabla?

Zadatak 23.
Napišite pseudokod algoritam koji će u $O(n)$ vremenu izračunati najdulji monotono rastući neprekinuti podniz niza $X = \langle x_1, x_2, \ldots, x_n \rangle$ cijelih brojeva. Primjerice, za $X = \langle -3, 5, 2, 1, 4, 8, 6, 1, 2, 3, 5, 9 \rangle$ vaš algoritam treba vratiti podniz $\langle 1, 2, 3, 5, 9 \rangle$.

Zadatak 24.
Napišite pseudokod algoritma koji će u $O(n^2)$ vremenu izračunati najdulji monotono rastući (moguće i prekinuti) podniz niza $X = \langle x_1, x_2, \ldots, x_n \rangle$ cijelih brojeva. Primjerice, za $X = \langle -3, 5, 2, 1, 4, 8, 6, 1, 2, 3, 5, 9 \rangle$ vaš algoritam treba vratiti podniz $\langle -3, 1, 1, 2, 3, 5, 9 \rangle$.

Zadatak 25.
Zadan je niz od n operacija na nekoj strukturi podataka. Trošak $i - te$ operacije je i ako je i potencija broja 2 te 1 u suprotnom. Koristeći agregatnu analizu odredite amortizirani trošak po operaciji.

Zadatak 26. *
Zadan je stog veličine najviše k. Na stogu se izvodi niz od n standardnih operacija, a nakon izvršenih k operacija se napravi kopija cijelog stoga (*backup*). Pridružite odgovarajući amortizirani trošak operacijama te na taj način pokažite da je trošak za n operacija, uključujući i kopiranje $\mathcal{O}(n)$.

Zadatak 27.
Zadan je niz od n operacija na nekoj strukturi podataka. Trošak $i - te$ operacije je i ako je i potencija broja 2 te 1 u suprotnom. Koristeći metodu kreditiranja odredite amortizirani trošak po operaciji.

Zadatak 28. *
Zadan je niz od n operacija na nekoj strukturi podataka. Trošak $i - te$ operacije je i ako je i potencija broja 2 te 1 u suprotnom. Koristeći metodu potencijala odredite amortizirani trošak po operaciji.

Zadatak 29.
Majmun tipka na tipkovnici koja ima samo 26 malih slova. Prilikom tipkanja svako je slovo odabrano nezavisno i uniformno iz abecede. Ako majmun natipka 1000000 slova, koji je očekivani broj pojavljivanja riječi *"dokaz"*?

Zadatak 30.
Koristeći indikator slučajnu varijablu rješite sljedeći problem pod nazivom **hat-check** problem. Svaki od n gostiju pri ulasku u restoran da svoj šešir osobi koja radi u garderobi. Na izlasku gostiju iz restorana osoba šešire podijeli u nasumičnom poretku. Koliki je očekivani broj gostiju koji su natrag dobili svoj šešir?

Zadatak 31.
U laboratoriju se vrši testiranje n uzoraka na prisutnost štetne tvari X. Svaki test može rezultirati negativno (uzorak nema štetanu tvar) ili pozitivno (uzorak ima štetnu tvar). Kada bi svaki uzorak posebno testirali to bi previše koštalo. Stoga je odlučeno napraviti sljedeće - nasumično podijeliti uzorke u grupe veličine k (pretpostavljamo da je k dijeli n), zatim pomiješati svih k uzoraka iz jedne grupe te ga testirati. Ako je rezultat tog testa negativan onda su svi uzorci u grupi negativni, a ako je rezultat pozitivan tada je potrebno napraviti dodanih k testiranja svakog uzorka iz grupe kako bi otkrili koji su uzorci pozitivni. Pretpostavimo da je vjerojatnost da uzorak bude pozitivan neovisna o drugim uzorcima i iznosi p.

a) Koja je vjerojatnost da test za grupu od k uzoraka bude pozitivan?

b) Koliki je očekivani broj testova potreban?

Zadatak 32.
Zadatak je pronaći element x u nesortiranom polju A veličine n. Predložen je sljedeći randomizirani algoritam. Odaberi slučajno jedan indeks i od A. Ako je $A[i] = x$ tada algoritam završava, inače, nastavlja dalje s pretragom birajući novi slučajni indeks od A. Postupak nastavlja sve dok ne pronađe indeks j od A za koje $A[j] = x$ ili sve dok nisu ispitani svi indeksi od A. Primijetite da se u ovom postupku indeksi biraju uvijek iz skupa $\{1, 2, ..., n\}$ pa neki indeksi mogu više puta biti odabrani.

a) Napišite pseudokod za proceduru Random-Search kako bi implementirali spomenutu strategiju. Vodite računa da algoritam terminira kad su svi indeksi ispitani.

b) Pretpostavimo da postoji točno jedan indeks i za koje je $A[i] = x$. Koji je očekivani broj indeksa od A koji moraju biti odabrani prije nego je x pronađen kad algoritam terminira?

Zadatak 33.
Promatrajmo proces koji se sastoji od ubacivanja jednakih loptica u b kutija, označenih s $1, 2, ..., b$. Bacanja su međusobno nezavisna i prilikom svakom bacanja jednaka je vjerojatnost pogotka bilo koje kutije. Odgovorite na sljedeća pitanja:

a) Koliki je očekivani broj loptica koje će upasti u neku kutiju?

b) Koliko bi, u prosjeku, loptica morali baciti da neka kutija sigurno sadrži jednu lopticu?

c) Koliko loptica očekujemo da moramo baciti kako bi svaka kutija sadržavala barem jednu lopticu?

Zadatak 34.
Na raspolaganju imamo generator pseudoslučajnih brojeva s uniformnom distribucijom nad intervalom $(0, 1)$.

a) Kako biste implementirali funkciju RANDOM(a,b) koja s jednakom vjerojatnošću vraća bilo koji broj iz intervala (a, b)?

b) Kako biste implementirali funkciju RANDOM(a,b) koja s jednakom vjerojatnošću vraća cijeli broj iz intervala RANDOM(a,b)?

Zadatak 35.
Jedan od načina za poboljšavanje algoritma RANDOMIZED-QUICKSORT je da particioniranje radimo pomoću pivota koji je odabran tako da prvo slučajnim odabirom uzmemo tri elementa iz zadanog polja A, a zatim od ta tri elementa odaberemo njihov medijan (srednji član). U ovom slučaju, pretpostavimo da polje A sadrži $ngeq3$ elemenata i označimo sortirano, izlazno polje s $A'[1...n]$.

a) Označimo sa x pivota, a sa $p_i = P(x = A'[i])$ vjerojatnost da pivot i-ti najmanji element, $i = 1, ..., n$.

b) Koliko se povećala vjerojatnost da za pivota odaberemo medijan podataka u usporedbi sa standardnim RANDOMIZED-QUICKSORT algoritmom? Pretpostavite da $n \to \infty$.

Zadatak 36.
Simulirajte COUNTING-SORT algoritam na polju $A = [6, 0, 2, 0, 1, 3, 4, 6, 1, 3, 2]$.

Zadatak 37. *
Sljedeći zadaci se odnose na algoritam za sortiranje u linearnom vremenu je RADIX-SORT (CLRS 8.3)

a) Demonstrajte RADIX-SORT na sljedećem primjeru gdje je potrebno sortirati riječi iz engleskog jezika: COW, DOG, SEA, RUG, ROW, MOB, BOX, TAB, BAR, EAR, TAR, DIG, BIG, TEA, NOW, FOX.

b) Koristeći indukciju pokažite da RADIX-SORT radi korektno. U kojem dijelu dokaza je potrebna pretpostavka da je algoritam za sortiranje u međukoraku stabilan?

Zadatak 38. *
Sljedeći zadaci se odnose na algoritam za sortiranje u linearnom vremenu je BUCKET-SORT (CLRS 8.4)

a) Demonstrirajte BUCKET-SORT na sljedećem primjeru
$A = [0.79, 0.13, 0.16, 0.64, 0.39, 0.20, 0.89, 0.53, 0.71, 0.42]$

b) Koje je najgore vrijeme izvršavanja BUCKET-SORT algoritma? Objasnite zašto.

1.2 Programerski zadaci

Zadatak 39.
Pokazni primjer kojem je cilj ponavljanje objektno orijentiranog programiranja u kojemu se razmatra klasa razlomak.

Zadatak 40. *
Implementirajte klase koje predstavljaju geometrijske likove sa sljedećim specifikacijama.

- Definirajte virtualnu klasu GeomLik koja reprezentira proizvoljni geometrijski lik s virtualnim funkcijama povrsine (GeomLik::povrsina), ispisa (GeomLik::print), te operatorom uspoređivanja (GeomLik::operator$<$) koji uspoređuje geometrijske likove prema njihovim površinama.
- Definirajte klase Trokut, Pravokutnik i Krug koje nasljeđuju klasu GeomLik, te implementirajte virtualne funkcije nasljeđene iz klase GeomLik.
  - Implementirajte klasu Trokut tako da su argumenti konstruktora duljine stranica a, b i c (float).
  - Implementirajte klasu Krug tako da je argument konstruktora duljina polumjera r (float).
  - Implementirajte klasu Pravokutnik tako da su argumenti konstruktora duljine stranica a i b (float).
  - Svim klasama implementirajte funkciju površina koja vraća površinu lika dobivenu pomoću argumenata konstruktora.
  - Svim klasama implementirajte funkciju ispis koja ispisuje o kojem se geometrijskom liku radi, te njegovu površinu (npr. Trokut 5).
- Klasama Trokut, Pravokutnik i Krug učinite podatke o duljinama stranica i polumjera privatnima. Usmeno obrazložite svrhu privatnih varijabli i funkcija unutar klase.
- Klasama Trokut, Pravokutnik i Krug onemogućite instanciranje bez unosa parametara danih u zadatku pod 2. Usmeno obrazložite zašto je to ponekad korisno.
- Pomoću operatora new instancirajte polje geometrijskih likova duljine 10 s elementima: Trokut(3, 4, 5), Pravokutnik(2, 4), Krug(10), Trokut(2, 2, 3), Krug(3), Krug(5), Pravokutnik(1, 10), Trokut(2, 4, 5), Krug(1), Pravokutnik(2, 2).
- Algoritmom Bubble Sort sortirajte polje geometrijskih likova iz zadatka prema njhovoj površini počevši od najmanje. Bubble Sort algoritam sortira polje (array) tako da prolazi poljem od zadnjeg elementa prema prvom. Ukoliko je lijevi susjed nekog elementa veći od njega samog, algoritam ih zamijeni. Ako je algoritam napravio zamjenu prilikom prolaska, procedura se ponavlja. U suprotnom algoritam terminira. Implementirajte funkciju BubbleSort koja će

po površini sortirati proizvoljno polje pokazivača na geometrijske likove od najvećeg ka najmanjem.

- Korisne poveznice:
  - `https://visualgo.net/sorting`
  - `https://en.wikipedia.org/wiki/Bubble_sort`

  Primjer polja geometrijskih likova:

```
GeomLik* A[10];
A[0] = new Trokut(3, 4, 5);
A[1] = new Pravokutnik(2, 4);
A[2] = new Krug(10);
A[3] = new Trokut(2, 2, 3);
A[4] = new Krug(3);
A[5] = new Krug(5);
A[6] = new Pravokutnik(1, 10);
A[7] = new Trokut(2, 4, 5);
A[8] = new Krug(1);
A[9] = new Pravokutnik(2, 2);
```

Zadatak 41. *

Ovaj zadatak se nastavlja na prethodni.

- Definirajte klasu Sesterokut koja reprezentira pravilni šesterokut stranice duljine a po uzoru na klase Krug, Trokut i Pravokutnik.
- Konstruktori klase Trokut mogu kao argumente primiti negativne duljine stranica ili vrijednosti stranica koje ne mogu činiti trokut. Koristeći funkciju assert iz biblioteke ¡assert.h¿ terminirajte izvršavanje programa ukoliko su konstruktoru klase Trokut proslijeđeni pogrešni argumenti. Funkcija assert terminira izvršavanje programa ukoliko kao argument primi varijablu tipa bool s vrijednosti 0.
- Po uzoru na prethodni zadatak, nadopunite konstruktore klasa Pravokutnik, Krug i Sesterokut.
- Implementirajte funkciju BubbleSort koja će po površini sortirati proizvoljno polje pokazivača na geometrijske likove od najvećeg ka najmanjem.
- Implementirajte funkciju Generator koja kao argument prima string oblika ”NAZIV_LIKA ARG1 ARG2 ARG3”, gdje je NAZIV_LIKA Krug, Pravokutnik, Trokut ili Sesterokut, dok su ARG1, ARG2 i ARG3 argumenti koje primaju konstruktori klasa Krug, Pravokutnik, Trokut i Sesterokut. Funkcija Generator treba vratiti pokazivač na geometrijski lik s odgovarajućim duljinama stranica ili polumjera (npr. koristite funkciju atof iz biblioteke ¡cstdlib¿ za konverziju stringa u float, npr. za string s = ”20.5”, float a = atof(s.c_str())), te mora prepoznati pogrešan unos. Primjer: Generator(”Pravokutnik 2 5”) treba vratiti pokazivac

na Pravokutnik s duljinama stranica 2 i 5. Generator("Trokut 1 2") treba vratiti nulu.

Zadatak 42. Napišite program koji računa izraze oblika b^e, gdje je $e \in \mathbb{N}_0$ i b cijeli ili racionalan broj. Za računanje rezultata koristite rekurzivni algoritam brzog potenciranja koji je demonstriran na idućem primjeru.

$$2^7 = 2 * 2^6 = 2 * (2^3)^2 = 2 * (2 * 2^2)^2$$

Program na ulazu prima parove cijelih brojeva b i e, te za svaki par vraća rezultat b^e.

Primjer 1.1. Input: 1 100 2 3
Output: 1 8

Zadatak 43.
Koristeći se kodom iz prethodna dva zadatka, napišite program koji efikasno računa n-ti Fibonaccijev broj F_n ako vrijedi:

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n = \begin{bmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{bmatrix}$$

Zadatak 44.
Dana je implementacija jednostruko povezane liste s podacima tipa int. Implementirajte funkciju reverse koja obrće unešenu jednostruko povezanu listu (npr. listu elemenata $1 \to 2 \to 3$ čiji je head 1 pretvara u listu $3 \to 2 \to 1$ čiji je head 3).

Zadatak 45.
Dana je implementacija jednostruko povezane liste s podacima tipa int. Implementirajte funkciju prune koja iz liste izbacuje svaki drugi element. Funkcija prune iz liste nikada neće izbaciti prvi element.

Primjer 1.2. Lista $4, 5, 6, 7, 8, 9$ će nakon poziva funkcije prune biti $4, 6, 8$.

Zadatak 46.
Dana je implementacija jednostruko povezane liste s podacima tipa int. Implementirajte funkciju sort koja sortira listu počevši od najmanjeg elementa.

Zadatak 47. *
U igri Snake (zmija) u cjelobrojnom koordinatnom sustavu postavljena je zmija duljine n tako da joj je "glava" na koordinati $(0, 0)$, a rep se proteže od glave sve do koordinate $(0, n)$. Zmija je reprezentirana pomoću liste uređenih parova dane u predlošku.

Napišite program koji unosi duljinu zmije n nakon koje slijedi niz naredbi koje označavaju smijer kretanja zmije. Tako naredba "L" pomiče zmiju jedno mjesto u lijevo (u negativnom smijeru x-osi), "R" u desno, "U" prema gore (pozitivan smjer na y-osi) i "D" prema dolje. Zmiju krećete tako da joj pomičete "glavu" za kojom slijedi "rep". Ukoliko se glava zmije sudari s repom (nađe na istoj koordinati kao jedan od segmenata repa) program terminira i ispisuje se "X". U suprotnom, nakon što su izvršene sve naredbe, ispišite "OK".

Primjer 1.3. Input: 10 U R D
Output: X

Primjer 1.4. Input: 20 U U R R D R U U L L L U U U L L L L L
Output: OK

Zadatak 48. *
Implementirajte funkciju insert koja u dvostruko povezanu listu ubacuje element (s ključem int) tako da je lista uvijek sortirana od najmanjeg do najvećeg elementa.

Primjer 1.5. Ulaz:
6
INS 1
INS 3
INS 2
INS 4
INS 5
DEL 2

Izlaz:
1
1 3
1 2 3
1 2 3 4
1 2 3 4 5
1 3 4 5

Zadatak 49.
Napišite program koji prati stanje maksimalnih elemenata stoga. Na ulazu je dan prirodan broj n, te nakon njega n cijelih brojeva koji su redom elementi stoga (od dna prema vrhu). Nakon toga, na ulazu je dodatno dan prirodan broj m. Ispišite maksimalni element u stogu nakon obavljenih m operacija "pop" (koje ukljanjaju prvi element sa stoga). Ukoliko nakon m operacija "pop" ne postoji maksimalni element u stogu, ispišite "X".

Primjer 1.6.

Input	*Output*
3 1 3 2 1	3
5 1 2 3 4 5 6	*X*
4 2 1 4 3 2	2

Zadatak 50. *
Učitajte string S duljine N. Provjerite je li S palindrom. Koristite strukturu Stack. Ukoliko dani niz znakova cini palindom ispisite DA, inace NE.

Primjer 1.7.

Input	*Output*
abba	*DA*
anavolimilovana	*DA*
danas	*NE*

Zadatak 51.
Zbog neočekivanog pada napona glavni računalni sustav jedne zračne luke izgubio je dio memorije u kojem je bio sadržan redoslijed ukrcavanja putnika na let L128. Na sreću, na kartama svakoga od putnika napisan je njihov prioritet (broj od 0 do 9) prilikom ukrcaja. Službenici zračne luke znaju kako grupe putnika proizvoljne veličine dolaze u razmacima od jedne minute. Također, u zrakoplov koji vrši let L128 moguće je ukrcati najviše 5 putnika u minuti. Napišite program koji će svake minute ukrcati do deset putnika počevši od onih s najvećim prioritetom.

Input se sastoji od blokova terminiranih znakom # u kojima su naznačeni prioriteti pojedinih putnika. Output se sastoji od niza prioriteta ukrcanih putnika.
Input
1 2 5 3 #
1 1 1 4 4 2 2 #
2 9 #

Output
5 3 2 1 4 4 2 2 1 9 2 1 1

Zadatak 52. *
Na avion se ukrcava N putnika od kojih svaki pripada ili 'business' (B) ili 'economy'

(E) klasi. Ukrcavanje putnika nadziru dva činovnika koja paze posebno na putnike iz klasa 'business' i 'economy' na način da za svaka dva propuštena putnika iz 'business' klase bude propušten jedan putnik iz 'economy' klase. Ukoliko je jedna od klasa ukrcana, putnici iz druge klase ukrcavaju se jedan po jedan. Simulirajte ukrcavanje na avion koristeći odgovarajuću strukturu podataka. Na ulazu očekujte dva prirodna broja koji će označavati broj putnika 'business' i 'economy' klase odvojene razmakom.
INPUT: 2 5
OUTPUT: B B E E E E E

INPUT: 5 7
OUTPUT: B B E B B E B E E E E E

Zadatak 53.
Pretpostavite da su telefonski brojevi duljine 7 znamenki od 0 do 9 sa znakom '-' između treće i četvrte znamenke (npr. 123-4567). Tipkovnica preko koje se unose telefonski brojevi daje mogućnost mapiranja slova na brojeve, i to na sljedeći način.

A, B, C mapira na 2,

D, E, F mapira na 3,

G, H, I mapira na 4,

J, K, L mapira na 5,

M, N, O mapira na 6,

P, R, S mapira na 7,

T, U, V mapira na 8,

W, X, Y mapira na 9.

Znakovi Q i Z nisu podržani, dok su svi znakovi '-' ignorirani prilikom poziva. Dakle, sljedeća tri broja su ekvivalentna: 111-8378, 111-TEST, 11-18–378 dok je standardni zapis 111-8378. Dva broja su ekvivalentna ako imaju jednake standardne zapise.
Pomozite u kreiranju telefonskog imenika tako da ukažete na ekvivalentne telefonske brojeve.
Ulaz: Prva linija inputa sadrži broj telefonskih brojeva koji se nalaze u imeniku, dok svaka sljedeća linija sadrži jedan telefonski broj (sedam znakova različitih od '-', bez znakova Q i Z).
Izlaz: Ukoliko su neki telefonski brojevi ekvivalentni, linija outputa treba sadržavati telefonski broj u standardnom obliku i broj ekvivalentnih telefonskih brojeva odvo-

jenih razmakom. Posebno, ukoliko su svi brojevi jedinstveni, program treba ispisati "svi razliciti".
Primjer:

Ulaz:	TUT-GLOP	**Izlaz:**
12	967-11-11	310-1010 2
4873279	310-GINO	487-3279 4
ITS-EASY	F101010	888-4567 3
888-4567	888-1200	
3-10-10-10	-4-8-7-3-2-7-9-	
888-GLOP	487-3279	

Zadatak 54.
Implementirajte klasu BST koja predstavlja binarno stablo pretraživanja.

Zadatak 55. *
Implementirajte algoritam za traženje najnižeg zajedničkog pretka definiran u teorijskom zadatku 3.

Zadatak 56. *
Koristeći kod iz prethodnih zadataka (binarno stablo pretraživanja), pomoću funkcije fold izračunajte produkt svih parnih elemenata binarnog stabla pretraživanja čiji su elementi cijeli brojevi.
Stablo unosite tako da ga inicijalno instancirate praznim, a zatim unosite proizvoljan broj elemenata koje u stablo dodajete pomoću metode INSERT.

ULAZ
1 2 3 4 5 6 7 8

IZLAZ
384

Zadatak 57.
Davno ste naučili da je zbrajanje prirodnih brojeva komutativna operacija. Uistinu, $10 + 1 = 1 + 10 = 11$. No sada nam neće biti svejedno kojim redoslijedom zbrajamo tri ili više prirodnih brojeva jer će svaki međurezultat (međuzbroj) imati cijenu. Svaki međuzbroj je ujedno i cijena koja se dodaje sljedećem pribrojniku. Cijena zabavnog zbrajanja je zbroj cijena međuzbrojeva.

U sljedećoj tablici prikazana su tri načina zbrajanja brojeva 1, 2 i 3. Ukoliko prvo zbrojimo 1 i 2, cijena tog zbrajanja je $3 = 1 + 2$. Nadalje, sljedeći pribrojnik je 3 i njemu se dodaje cijena prethodnog zbrajanja što ukupno iznosi $6 = 3 + 3$. Dakle, imali smo dva koraka. Prvi je koštao 3, a drugi 6, što je ukupno 9. Ostale načine interpretiramo analogno.

Primjer 1.8.
$1 + 2 = 3$, cijena = 3
$3 + 3 = 6$, cijena = 6
Ukupno = 9

Primjer 1.9.
$1 + 3 = 4$, cijena = 4
$2 + 4 = 6$, cijena = 6
Ukupno = 10

Primjer 1.10.
$2 + 3 = 5$, cijena = 5
$1 + 5 = 6$, cijena = 6
Ukupno = 11

Za dani niz prirodnih brojeva potrebno je odrediti minimalnu cijenu zbrajanja. U prethodnom primjeru minimalna cijena je 9.

Upute:

- ulaz se sastoji od dva redka gdje u prvom piše prirodan broj $2 \leq n \leq 5000$, a u drugom se nalazi n prirodnih brojeva od kojih su svi manji od 100000.
- izlaz se sastoji od jednog broja koji predstavlja minimalnu cijenu zbrajanja n prirodnih brojeva na ulazu

Napomena: Preporuča se korištenje priority_queue strukture podataka u queueSTL biblioteci: `http://www.cplusplus.com/reference/queue/priority_queue/`.

Primjer 1.11.

Ulaz	*Izlaz*
3 1 2 3	9
4 1 2 3 4	19

Zadatak 58.
Na nepoznatom se računalu nalazi struktura podataka za koju znamo da podržava operacije push i pop. Temeljom uparenog inputa i outputa poslanog i primljenog s nepoznatog računala odredite o kojoj se strukturi podataka radi. Mogućnosti su red, stog i prioritetni red. Operacije na nepoznatom računalu se izvršavaju na sljedeći način.

- ”1 x” označava push elementa x na nepoznatu strukturu.

- "2 x" označava pop koji vraća vrijednost x i uklanja ju sa strukture.

Input započinje linijom u kojoj se nalazi prirodan broj (od 0 do 1000) koji označava broj operacija nad nepoznatom strukturom koje dolaze u sljedećim linijama. Ovakvi blokovi inputa mogu se ponavljati više puta. Na outputu se, za svaki blok u inputu, nalazi odluka o vrsti nepoznate strukture podataka koja može biti jedna od sljedećih.

- stog
- red
- prioritetni red
- neodlucno (nepoznata struktura nije jednoznačno određena)
- nemoguce (nepoznata struktura nije niti stog, niti red, niti prioritetni red)

Primjer 1.12.

Ulaz	*Izlaz*
6 1 1 1 2 1 3 2 1 2 2 2 3	*red*
6 1 1 1 2 1 3 1 3 2 2 2 1	*neodlucno*
2 1 1 2 2	*nemoguce*
4 1 2 1 1 2 1 2 2	*stog*
7 1 2 1 5 1 1 1 3 2 5 1 4 2 4	*prioritetni red*

Zadatak 59. *

a) Implementirajte klasu koja predstavlja AVL stablo sa sljedećim specifikacijama
   - svaki čvor ima pokazivač na lijevo i desno dijete, roditelja, ključ i visinu na kojoj se nalazi
   - operacija INSERT za dodavanje novog čvora u stablo, operacija DELETE za brisanje čvora iz stabla
   - operacije za traženje sljedbenika i prethodnika
   - opracije LEFTROTATE i RIGHTROTATE za rotiranje podstabla oko zadanog čvora
   - metoda HEIGHT koja za zadani čvor vraća njegovu visinu
   - metoda BALANCE koja za proizvoljni čvor vraća razliku visina njegovog desnog i lijevog djeteta
   - NUMBEROFNODESLESS, NUMBEROFNODESGREATER, NUMBEROFNODESINRANGE koje za proslijeđeni čvor vraćaju broj čvorova čiji su ključevi manji od zadanog, veći od zadanog, odnosno u zadanom intervalu.

b) Proširite vaše AVL stablo tako da metoda NUMBEROFNODESINRANGE bude efikasnija.

`https://www.cs.usfca.edu/~galles/visualization/AVLtree.html`
`https://visualgo.net/en`

Zadatak 60.
Na temelju ankete provedene među putnicima jednog broda na kojem je organizirano krstarenje Kornatima, svakoj od pojedinih etapa krstarenja pridružen je cijeli broj koji označava estetsku ocjenu etape. Etapom smatramo prostor između dvije uzastopne plutače. Napišite program koji će pronaći rutu za krstarenje s najvećom sumom ocjena. Rutom smatramo sve etape između dvije plutače.
Na prvom mjestu u inputu se nalazi prirodan broj n koji označava broj krstarenja kojima valja maksimizirati sumu estetskih ocjena.
Svako krstarenje započinje prirodnim brojem m koji označava broj plutača.
Sljedećih m brojeva su estetske ocjene (m - 1) etapa koje se nalaze između plutača.
Za svako i od n krstarenja definiramo maksimalnu estetsku sumu s.
Na outputu treba stajati:

- ako je s pozitivno ispiši "Najbolja ruta na krstarenju i se nalazi između plutača a i b."
- inače ispiši "Krstarenje i nije pogodno."

Ukoliko unutar krstarenja postoji više ruta s maksimalnom estetskom sumom preferirajte one koje prolaze kraj više plutača i s manjim indeksom početne plutače.

Input
3
3
1 5
3
-1 -2
5
3 -2 3 -1

Output
The nicest part of route 1 is between stops
1 and 3
Bad cruising
The nicest part of route 1 is between stops
1 and 4

Input
3
5
2 1 -3 3
5
-1 -2 5 2
5
1 4 -1 1

The nicest part of route 1 is between stops
1 and 3
The nicest part of route 2 is between stops
3 and 5
The nicest part of route 3 is between stops
1 and 5

Zadatak 61. Unosite retke terminirane brojem -999999 u kojima su zapisani vektori cijelih brojeva.
Odredite najveći uzastopni produkt brojeva u vektoru.

Input
2 3 -999999
-2 2 -3 4 -5 -999999
1 2 3 - 999999
-2 2 -3 4 -5 -999999
-5 -3 4 1 -8 -999999
20 -3 4 1 -8 -999999

Output
6
120
6
120
160 640

Zadatak 62.
Unesite kvadratnu matricu cijelih brojeva dimenzije n (po recima), te u njoj odredite najveću sumu elemenata svih podmatrica u zadanoj matrici.
Input
4
0 -2 -7 0
9 2 -6 2
-4 1 -4 1

-1 8 0 -2

Output
15

Zadatak 63.
Na raspolaganju imate neograničen broj kovanica od 1, 2, 5, 10, 20 i 50 lipa. Za unešeni broj n odredite najmanji mogući broj kovanica čija je ukupna vrijednost n lipa.

Input
36
Output
4

Zadatak 64.
U nizu cijelih brojeva $X = \langle x_1, x_2, \ldots, x_n \rangle$ potrebno je pronaći neprekinuti podniz takav da mu je suma članova najveća.
Primjer:

Input: n (duljina niza)	$x_1, ..., x_n$	Output: suma najvećeg niza
Input: 9	-2 1 -3 4 -1 2 1 -5 4	Output: 6
Input: 6	2 -1 2 3 4 -5	Output: 10
Input: 5	-2 -3 -1 -4 -6	Output: -1

Zadatak 65.
Napišite algoritam koji će u $O(n)$ vremenu pronaći najdulji monotono rastući neprekinuti podniz niza $X = \langle x_1, x_2, \ldots, x_n \rangle$ cijelih brojeva.
Primjer:

Input: n (duljina niza)	$x_1, ..., x_n$	Output: $x_i, ...x_j$
Input: 12	-3 5 2 1 4 8 6 1 2 3 5 9	Output: 1 2 3 5 9
Input: 9	6 3 4 8 10 5 7 1 9	Output: 3 4 8 10

Zadatak 66.
Napišite algoritma koji će u $O(n^2)$ vremenu pronaći najdulji monotono rastući (moguće i prekinuti) podniz niza $X = \langle x_1, x_2, \ldots, x_n \rangle$ cijelih brojeva te ispisati njegovu duljinu.
Primjer:

Input: n (duljina niza)	$x_1, ..., x_n$	Output: duljina najduljeg niza
Input: 9	2 5 1 3 4 8 3 6 7	Output: 5
Input: 6	2 4 3 7 4 5	Output: 4

Zadatak 67.
Palindrom je neprazni string koji se čita isto unaprijed i unatrag. Primjeri palindroma su: *kisik*, *rotor*, *noon*. Vaš zadatak je u zadanom nizu $\langle x_1, ..., x_n \rangle$ pronaći najduži palindrom. **Primjer:**

Input:$x_1, ..., x_n$	Output: duljina najduljeg palindroma
Input: CHARACTER	Output: 5

Zadatak 68.
Na raspolaganju imate neograničeni broj kovanica različitih vrijednosti $m_1, m_2, ..., m_m$. Za uneseni broj n treba odrediti broj načina na koji se može vratiti n kuna koristeći kovanice koje su vam na raspolaganju.
Primjer: Na raspolaganju su vam kovanice u vrijednosti $8, 3, 1, 2$ te je potrebno razmijeniti 3 kune. To možete učiniti na tri načina, kao $1+1+1, 3, 1+2$.

- ULAZ: n m $m_1, \ldots, m_m$
- IZLAZ: broj načina na koji se može vratiti n
- ULAZ: 4 3 1 2 3
- IZLAZ: 4
- ULAZ:10 4 2 5 3 6
- IZLAZ: 5
- ULAZ: 18 23 49 9 40 17 46 24 42 26 43 41 35 1 47 28 20 38 2 44 32 22 18 45 25
- IZLAZ: 18

Zadatak 69.
Neka je dan štap duljine L, te skup točaka $C = \{c_1, \ldots, c_m\}$ čiji su elementi između 0 i L. Točke iz C označavaju mjesta na štapu počevši od njegovog lijevog ruba na kojima trebaju biti izvršeni rezovi.Cijena reza određena je trenutnom duljinom štapa koji režemo.

Primjer 1.13. Režemo štap duljine 100, redom, na mjestima 25 i 75. Prvo ćemo platiti 100 kako bismo dobili jedan štap duljine 25 i drugi duljine 75. Zatim radimo drugi rez na štapu duljine 75 kojega dijelimo na štapove duljina 50 i 25 s cijenom 75.

Napišite program koji će tehnikom dinamičkog programiranja odrediti minimalnu cijenu rezanja štapa u točkama iz C. Na ulazu se nalazi duljina štapa L nakon koje slijedi proizvoljno mnogo rezova. Ispišite samo minimalnu traženu cijenu.

- ULAZ: L m $c_1, \ldots, c_m$
- IZLAZ: minimalna tražena cijena
- ULAZ: 100 3 25 50 75
- IZLAZ: 200
- ULAZ: 20 3 3 8 10
- IZLAZ: 37

- ULAZ: 20 4 2 3 8 10
- IZLAZ: 40

Zadatak 70.
Aladin više od svega voli jesti kolače. Zbog toga je odlučio posjetiti veliki sajam slastica na kojem se nada pronaći svoje omiljene vrste kolača (svi su od banane). Na sajmu se domogao ukupno n kolača od kojih svaki ima c_i kalorija. Budući da Aladin želi sačuvati svoju besprijekornu liniju, za svaki kolač koji pojede mora pretrčati $2^i \cdot c_j$ kilometara, gdje je i broj kolača koje je Aladin pojeo prije kolača j. Odredite minimalan broj kilometara koje Aladin mora pretrčati ukoliko želi pojesti sve kolače.

- INPUT: 3 1 3 2
- OUTPUT: 11

Zadatak 71.
Za dan skup $A \subset \mathbb{Z}$ elemenata a_i, $i = 1, \ldots n$ odredite podskup A' kardinalnosti k koji minimizira izraz

$$\max_{a \in A'} a - \min_{a \in A'} a.$$

Na inputu se nalazi broj k nakon kojega dolaze elementi skupa A. Na outputu ispišite minimum gornjeg izraza.

- INPUT: 3 10 100 300 200 1000 20 30
- OUTPUT: 20

Zadatak 72.
Učiteljica želi podijeliti bombone svojim učenicima u razredu. Učenici su se poslagali u vrstu i učiteljica im želi podijeliti bombone tako da svatko dobije barem jedan bombon, a kada dva učenika sjede jedan do drugoga više bombona mora dobiti onaj koji ima bolje ocjene*. Učiteljica želi minimizirati broj bombona koje mora kupiti.

* Bolje ocjene znači sljedeće: svakom učeniku je izračunat prosjek ocjena.

Primjer:
Svakom učeniku je pridružena ocjena i poslagali su se na sljedeći način: $4, 6, 4, 5, 6, 2$. Bombone tada treba podijeliti: $1, 2, 1, 2, 3, 1$ i minimalni broj bombona je 10.

- ULAZ: broj učenika n ocjene $o_1, ..., o_n$
- IZLAZ: minimalni broj bombona koje treba kupiti
- ULAZ: 3 1 2 2
- IZLAZ: 4

- ULAZ: 10 2 4 2 6 1 7 8 9 2 1
- IZLAZ: 19
- ULAZ: 8 2 4 3 5 2 6 4 5
- IZLAZ: 12

Zadatak 73.
Kažemo da je niz znakova *dobar* ako se sastoji od samo dvije vrste znakova - 'a' i 'b' te su svaka dva uzastopna znaka različita. Npr. 'aba' je dobar, dok 'abb' nije dobar.

Na ulazu je zadano a znakova 'a', b znakova 'b' i c znakova 'ab'. Koristeći zadane znakove koja dužina najduljeg 'dobrog' niza znakova?

- ULAZ: a b c
- IZLAZ: duljina najduljeg dobrog niza.
- ULAZ: 1 1 1
- IZLAZ: 4
- ULAZ: 2 1 2
- IZLAZ: 7
- ULAZ: 3 5 2
- IZLAZ: 11
- ULAZ: 2 2 1
- IZLAZ: 6

Zadatak 74. *
Djevojčica je za Sv.Nikolu dobila n slatkiša označenih sa brojevima $1, ..., n$ i svaki slatkiš ima a_i šećera. Djevojčica voli jesti slatkiše, ali svaki dan smije pojesti najviše m slatkiša. Ako dane d označimo s brojevima $d = 1, 2, ...$ tada jedenje slatkiša i na dan d uzrokuje unos $d \cdot a_i$ šećera jer slatkiši postaju sve slađi kako dani prolaze.
Ukupan unos šećera je zbroj unos šećera za svaki slatkiš posebno.
Pretpostavite da djevojčica odabere k slatkiša i pojede ih u bilo kojem poretku, koliki je najmanji unos šećera koji može postići?

ULAZ:
n - broj slatkiša, m - dozvoljni broj slatkiša u jednom danu
$a_1, ..., a_n$

IZLAZ: n brojeva $x_1, ...x_n$ gdje x_k označava najmanju ukupnu količinu šećera koju može unijeti ako odabere k slatkiša

ULAZ:
9 2
6 19 3 4 4 2 6 7 8

IZLAZ: 2 5 11 18 30 43 62 83 121

ULAZ:
1 1
7

IZLAZ: 7

Zadatak 75. *

Grupa prijatelja kupuje buket cvijeća za poklon, troškove će podijeliti tako da svaki od prijatelja plati za neki cvijet u buketu. No cvijećar želi iskoristiti priliku i što više naplatiti taj buket pa je odlučio cijenu svakog cvijeta kojeg će staviti u buket uvećati i to na sljedeći način: cijena cvijeta će biti njegova originalna cijena pomnožena s brojem cvijetova koje je taj prijatelj već kupio $(0+1) \times cijena, (1+1) \times cijena, ...$
Npr. 3 prijatelja žele kupiti buket od 4 cvijeta kojima su cijene $1, 2, 3, 4$. Na koji način se trebaju rasporediti pri kupnji tako da prođu što jeftinije?

- ULAZ: n- broj cvijetova u buketu m - broj prijatelja $c_1, ...c_n$ -cijene cvijetova
- IZLAZ: najmanja cijena koju mogu platiti
- ULAZ: 3 3 2 5 6
- IZLAZ: 13
- ULAZ: 5 3 1 3 5 7 9
- IZLAZ: 29
- ULAZ: 10 4 3 2 1 6 7 4 8 11 14 5
- IZLAZ: 85

Zadaci označeni sa * su za zadaću te na njima možete skupiti dodatne bodove s kojma je moguće utjecati na konačnu ocjenu. Rok za slanje rješenja će biti dodatno definiran (oko trećeg kolokvija).