Zadaci za praktikum
Programiranje II
Matematički fakultet u Beogradu

27. maj 2016

Sadržaj

1 I nedelja: Obnavljanje gradiva Programiranja 1

1.1 Zadaci

1. Napisati funkciju koji određuje dva najveća elementa niza a i program koji je testira. Elementi niza se unose sa standardnog ulaza sve dok se ne unese 0. Pretpostaviti da niz neće imati više od 32 elemenata.

Primer 1

```
Interakcija sa programom:
 1 2 3 4 5 6 0
 5 6
```

Primer 2

```
Interakcija sa programom:
 3 4 -1 2 3 0
 3 4
```

Primer 3

```
Interakcija sa programom:
 4 5 4 5 2 0
 5 5
```

2. Napisati program koji ispituje da li je niska koji se unosi sa standardnog ulaza . Pretpostaviti da se neće unositi niska duza od 256 karaktera.

Primer 1

```
Interakcija sa programom:
 abba
 da
```

Primer 2

```
Interakcija sa programom:
 programiranje
 ne
```

Primer 3

```
Interakcija sa programom:
 aa
 da
```

3. Napisati funkciju koja kvadrira one elemente niza koji su veći od aritmetičke sredine niza. Niz se unosi sa tastature, pod uslovima kao u zadatku 1. Na standardni izlaz ispisati promenjen niz.

Primer 1

```
Interakcija sa programom:
 1 2 3 4 5 6 0
 1 2 3 16 25 36
```

Primer 2

```
Interakcija sa programom:
 3 4 -1 2 3 0
 9 16 -1 2 9
```

Primer 3

```
Interakcija sa programom:
 4 5 4 5 2 0
 4 25 4 25 2
```

4. Napisati program koji utvrđuje koliko se puta data reč pojavljuje u datoj tekstualnoj datoteci. Naziv datoteke kao i reč zadaju se kao argumenti komandne linije.

Primer 2

```
Poziv: ./a.out test.txt programiranje

test.txt
 programiranje je jako vazno, i programiranje se lako uci.
 programiranje treba redovno vezbati.

Izlaz:
 3
```

5. U datoteci, čije se ime zadaje preko komandne linije, dat je neki tekst. Napisati program koji šifruje dati tekst tzv. Cezarovom šifrom, i rezultujući tekst upisuje u drugu datoteku, čije ime se takođe zadaje preko komandne linije. Ukoliko druga datoteka nije navedena ispisati rezultat na standardni izlaz. Cezarovom šifrom se slova kodiraju na sledeći način:

$$A \to D, B \to E, \ldots, X \to A, Y \to B, Z \to C,$$

slično za mala slova. Ostali karakteri ostaju nepromenjeni.

Primer 1

```
Poziv: ./a.out ulaz.txt izlaz.txt

ulaz.txt
 Abcde-Programiranje2

izlaz.txt
 Defgh-Surjudpludqmh2
```

Primer 2

```
Poziv: ./a.out ulaz.txt

ulaz.txt
 Ja sve umem!

Izlaz:
 Md vyh xphp!
```

6. Napisati program koji sa standardnog ulaza učitava cele brojeve, dok se ne unese 0 i smešta ih u niz. Potom, iz niza a izbacuje sve elemente $a[i]$ koji imaju sledeće svojstvo: $i > 0$ i $a[i]$ je manji od svih elemenata koji mu prethode u nizu a. Nakon izbacivanja, program ispisuje trenutni sadržaj niza. Pretpostaviti da niz neće imati više od 20 elemenata.

Primer 1

```
Interakcija sa programom:
 1 2 3 4 5 6 0
 1 2 3 4 5 6
```

Primer 2

```
Interakcija sa programom:
 3 4 -1 2 3 0
 3 4 3
```

Primer 3

```
Interakcija sa programom:
 4 5 3 7 4 2 0
 4 5 7 4
```

7. Napisati program koji ispituje da li data niska $s1$ nastaje cikličnim pomeranjim druge date niske $s2$ i ispisuje *da* ako nastaje, u suprotnom ispisuje *ne*. Niske se učitavaju sa standardnogg ulaza. Pretpostaviti da nijedna niska nije duža od 50 karaktera.

Primer 1

```
Interakcija sa programom:
 Unesite prvu nisku:    abcde
 Unesite prvu nisku:    cdeab
 da
```

2 II nedelja: Podela koda po datotekama

2.1 Zadatak za rad na času

1. Napisati biblioteku za rad sa polinomima.

   (a) Definisati strukturu `Polinom` koja opisuje polinom stepena najviše 20 koji je zadat nizom svojih koeficijenata tako da se na i-toj poziciji u nizu nalazi koeficijent uz i-ti stepen polinoma.

   (b) Napisati funkciju `void ispisi(const Polinom * p)` koja ispisuje polinom `p` na standardni izlaz, od najvišeg ka najnižem stepenu. Ipisati samo koeficijente koji su različiti od nule.

   (c) Napisati funkciju `Polinom ucitaj()` koja učitava polinom sa standardnog ulaza. Za polinom najpre uneti stepen, a zatim njegove koeficijente.

   (d) Napisati funkciju `double izracunaj(const Polinom * p, double x)` koja vraća vrednosti polinoma `p` u datoj tački `x` koristeći Hornerov algoritam.

   (e) Napisati funkciju `Polinom saberi(const Polinom * p, const Polinom * q)` koja vraća zbir dva polinoma `p` i `q`.

   (f) Napisati funkciju `Polinom pomnozi(const Polinom * p, const Polinom * q)` koja vraća proizvod dva polinoma `p` i `q`.

   (g) Napisati funkciju `Polinom izvod(const Polinom * p)` koja vraća izvod polinoma `p`.

   (h) Napisati funkciju `Polinom n_izvod(const Polinom * p, int n)` koja vraća `n`-ti izvod polinoma `p`.

   Napisati program koji testira prethodno napisane funkcije. Sa standardnog ulaza učitati polinome `p` i `q`, a zatim ih ispisati na standardni izlaz u odgovarajućem formatu. Izračunati i ispisati zbir `z` i proizvod `r` unetih polinoma `p` i `q`. Sa standardnog ulaza učitati realni broj `x`, a zatim na standardni izlaz ispisati vrednost polinoma `z` u tački `x` zaokruženu na dve decimale. Na kraju, sa standardnog ulaza učitati broj `n` i na izlaz ispisati `n`-ti izvod polinoma

   Sve vreme, paralelno sa razvojem funkcija, pisati i glavni program koji ih testira. Nakon toga, izdvojiti funkcije u zasebnu datoteku `polinom.c`, a program u `test-polinom.c`. Prikazati probleme u kompilaciji, pošto ne postoji više definicija strukture kao ni prototipovi funkcija u `test-polinom.c`. Disku- tovati da dupliranje ovoga u obe `C` datoteke dovodi do velikih problema prilikom održavanja programa. Uvesti `polinom.h` kao rešenje i uključiti ga u obe datoteke. Prikazati kreiranje objektne datoteke `polinom.o`.

```
gcc -o test-polinom polinom.c test-polinom.c
       i
gcc -c polinom.c
gcc -c test-polinom.c
gcc -o test-polinom polinom.o test-polinom.o
```

   Premestiti datoteke `polinom.h` i `polinom.o` (`ne polinom.c`) u novi direktorijum i napisati novi test program, prevesti ga i povezati.

Primer 1

```
Interakcija sa programom:
 Unesite polinom p (prvo stepen, pa zatim koeficijente od najveceg stepena do nultog):
 3 1.2 3.5 2.1 4.2
 Unesite polinom q (prvo stepen, pa zatim koeficijente od najveceg stepena do nultog):
 2 2.1 0 -3.9
 Zbir polinoma je polinom z:
 1.20x^3+5.60x^2+2.10x+0.30
 Prozvod polinoma je polinom r:
 2.52x^5+7.35x^4-0.27x^3-4.83x^2-8.19x-16.38
 Unesite tacku u kojoj racunate vrednost polinoma z:
 0
 Vrednost polinoma z u tacki 0.00 je 0.30
 Unesite izvod polinoma koji zelite:
 3
 3.  izvod polinoma r je:  151.20x^2+176.40x-1.62
```

2.2 Dodatni zadaci

1. Napisati biblioteku za rad sa razlomcima.

   (a) Definisati strukturu `Razlomak` koja opisuje razlomak.

(b) Napisati funkciju `Razlomak ucitaj()` za učitavanje razlomka.

(c) Napisati funkciju `void ispisi(const Razlomak * r)` koja ispisuje razlomak `r`.

(d) Napisati funkciju `int brojilac(const Razlomak * r)` koja vraćaja brojilac razlomka `r`.

(e) Napisati funkciju `int imenilac(const Razlomak * r)` koja vraćaja imenilac razlomka `r`.

(f) Napisati funkciju `double realna_vrednost(const Razlomak * r)` koja vraća odgovarajuću realnu vrednost razlomka `r`.

(g) Napisati funkciju `double reciprocna_vrednost(const Razlomak * r)` koja vraća recipročnu vrednost razlomka `r`.

(h) Napisati funkciju `Razlomak skrati(const Razlomak * r)` koja vraća skraćenu vrednost datog razlomka `r`.

(i) Napisati funkciju `Razlomak saberi(const Razlomak * r1, const Razlomak * r2)` koja vraća zbir dva razlomka `r1` i `r2`.

(j) Napisati funkciju `Razlomak oduzmi(const Razlomak * r1, const Razlomak * r2)` koja vraća razliku dva razlomka `r1` i `r2`.

(k) Napisati funkciju `Razlomak pomnozi(const Razlomak * r1, const Razlomak * r2)` koja vraća proizvod dva razlomka `r1` i `r2`.

(l) Napisati funkciju `Razlomak podeli(const Razlomak * r1, const Razlomak * r2)` koja vraća količnik dva razlomka `r1` i `r2`.

Napisati program koji testira prethodne funkcije. Sa standardnog ulaza učitati dva razlomka `r1` i `r2`. Na standardni izlaz ispisati skraćene vrednosti zbira, razlike, proizvoda i količnika razlomaka `r1` i recipročne vrednosti razlomka `r2`.

Primer 1

```
Interakcija sa programom:
 Unesite imenilac i brojilac prvog razlomka:  1 2
 Unesite imenilac i brojilac drugog razlomka:  2 3
 1/2 + 3/2 = 2
 1/2 - 3/2 = -1
 1/2 * 3/2 = 3/4
 1/2 / 3/2 = 1/3
```

3 III nedelja: Algoritmi za rad sa bitovima

3.1 Obavezni zadaci

1. Pretpostavimo da se broj `n` zapisuje pomoću `x` bitova. Napisati funkciju `int razlika(unsigned int n)` koja za dati broj `n` vraća razliku broja jedinica u x/2 bitova veće težine i broja jedinica u x/2 bitova manje težine. Napisati program koji tu funkciju testira za broj koji se zadaje sa standardnog ulaza.

Primer 1

```
Ulaz:
  10
Izlaz:
  Vise jedinica se nalazi na bitovima manje tezine.
```

Primer 2

```
Ulaz:
  2147377001
Izlaz:
  Vise jedinica se nalazi na bitovima vece tezine
```

Primer 3

```
Ulaz:
  0
Izlaz:
  Broj jedinica je jednak.
```

2. Napisati funkciju `int broj_parova(unsigned int x)` koja vraća broj pojava dve uzastopne nule u binarnom zapisu celog neoznačenog broja `x`. Napisati program koji tu funkciju testira za broj koji se zadaje sa standardnog ulaza. *Napomena: Tri uzastopne nule sadrže dve uzastopne nule dva puta.*

Primer 1

```
Ulaz:
  11
Izlaz:
  27
```

Primer 2

```
Ulaz:
  5656444
Izlaz:
  11
```

Primer 3

```
Ulaz:
  0
Izlaz:
  31
```

Primer 4

```
Ulaz:
  2147377146
Izlaz:
  1
```

3. Napisati program koji sa standardnog ulaza učitava pozitivan ceo broj, a na standardni izlaz ispisuje vrednost tog broja sa razmenjenim vrednostima bitova na pozicijama i i j. Pozicije i i j učitati kao parametre komandne linije. Pri rešavanju nije dozvoljeno koristiti ni pomoćni niz ni aritmetičke operatore +, -, /, *, %.

Primer 1

```
Poziv: ./a.out 1 2

Interakcija sa programom:
Ulaz:
  11
Izlaz:
  13
```

Primer 2

```
Poziv: ./a.out 1 2

Interakcija sa programom:
Ulaz:
  1024
Izlaz:
  1024
```

Primer 3

```
Poziv: ./a.out 12 12

Interakcija sa programom:
Ulaz:
  12345
Izlaz:
  12345
```

4. Napisati funkciju `unsigned int f1(unsigned int x)` koja u datom broju invertuje svaki treći bit. Prvi bit koji se invertuje je bit najmanje težine. Sa standardnog ulaza se unosi ceo pozitivan broj. Ispisati rezultat funkcije na standardni izlaz.

Primer 1

```
Ulaz:
  0
Izlaz:
  1227133513
```

Primer 2

```
Ulaz:
  345
Izlaz:
  1227133712
```

Primer 3

```
Ulaz:
  1024
Izlaz:
  1227134537
```

Primer 4

```
Ulaz:
  1
Izlaz:
  1227133512
```

3.2 Dodatni zadaci

1. Napisati program koji sa standardnog ulaza učitava ceo broj, a na standardni izlaz ispisuje vrednost tog broja sa razmenjenim vrednostima bajtova na pozicijama i i j, pri čemu važi $1 <= i,j <= 4$. Pozicije i i j učitati kao parametre komandne linije. Pri rešavanju nije dozvoljeno koristiti ni pomoćni niz ni aritmetičke operatore +, -, /, *, %. U slučaju pogrešnog unosa, na standardni izlaz za greške ispisati -1.

Primer 1

```
Poziv: ./a.out 1 2

Interakcija sa programom:
Ulaz:
  11
Izlaz:
  2816
```

Primer 2

```
Poziv: ./a.out 1 4

Interakcija sa programom:
Ulaz:
  11
Izlaz:
  184549376
```

Primer 3

```
Poziv: ./a.out 2 5

Interakcija sa programom:
Izlaz:
  -1
```

2. Napisati funkciju koja za data dva neoznačena broja x i y invertuje one bitove u broju x koji se poklapaju sa odgovarajućim bitovima u broju y. Ostali bitovi treba da ostanu nepromenjeni. Napisati program koji testira tu funkciju za brojeve koji se zadaju sa standardnog ulaza.

Primer 1

```
Ulaz:
  123 10
Izlaz:
  4294967285
```

Primer 2

```
Ulaz:
  3251 0
Izlaz:
  4294967295
```

Primer 3

```
Ulaz:
  12541 1024
Izlaz:
  4294966271
```

3. Napisati funkciju koja vraća broj petica u oktalnom zapisu neoznačenog celog broja x. Napisati program koji testira tu funkciju za broj koji se zadaje sa standardnog ulaza. *Napomena: Zadatak rešiti isključivo korišćenjem bitskih operatora.*

Primer 1

```
Ulaz:
  123
Izlaz:
  0
```

Primer 2

```
Ulaz:
  3245
Izlaz:
  2
```

Primer 3

```
Ulaz:
  100328
Izlaz:
  1
```

4 IV nedelja: Rekurzija

4.1 Obavezni zadaci

1. Napisati rekurzivnu funkciju koja odredjuje minimum niza celih brojeva. Napisati program koji testira ovu funkciju, za niz koji se učitava sa standardnog ulaza. Niz neće biti duži od 256 i njegovi elementi se unose sve do kraja ulaza.

Primer 1

```
Ulaz:
 1 5 9 8 4
Izlaz:
 1
```

Primer 2

```
Ulaz:
 6 -4 7 -9 3 3 7 8 1 3 0 7
Izlaz:
 -9
```

Primer 3

```
Ulaz:
 5
Izlaz:
 5
```

Primer 4

```
Ulaz:

Izlaz:
 Nije unet nijedan element.
```

2. Napisati rekurzivnu funkciju koja za nizove a i b računa sumu: $\sum_{i=1}^{n} {a_i}^2 * {b_i}^2$. Napisati program koji testira ovu funkciju za nizove koji se unose sa standardnog ulaza. Prvo treba uneti dimenziju nizova, a zatim i njihove elemente. Na standardni izlaz ispisati rezultat izvršavanja funkcije. Pretpostaviti da nizovi neće imati više od 256 elemenata.

Primer 1

```
Ulaz:
 Unesite n:
 3
 Unesite elemente niza a:
 1 2 3
 Unesite elemente niza b:
 1 2 3
Izlaz:
 98
```

Primer 2

```
Ulaz:
 Unesite n:
 2
 Unesite elemente niza a:
 0 -1
 Unesite elemente niza b:
 8 5
Izlaz:
 25
```

Primer 3

```
Ulaz:
 Unesite n:
 4
 Unesite elemente niza a:
 1 2 0 3
 Unesite elemente niza b:
 3 2 9 2
Izlaz:
 61
```

Primer 4

```
Ulaz:
 Unesite n:
 0
Izlaz:
 -1
```

Primer 5

```
Ulaz:
 Unesite n:
 300
Izlaz:
 -1
```

3. Napisati rekurzivnu funkciju koja sabira kubove dekadnih cifara celog broja x. Napisati program koji testira ovu funkciju, za broj koji se učitava sa standardnog ulaza.

Primer 1

```
Ulaz:
 Unesite x:
 123
Izlaz:
 36
```

Primer 2

```
Ulaz:
 Unesite x:
 3
Izlaz:
 27
```

Primer 3

```
Ulaz:
 Unesite x:
 -1005
Izlaz:
 126
```

4. Napisati rekurzivnu funkciju koja računa broj elemenata u nizu a, koji su strogo manji od zadatog broja k. Napisati program koji testira ovu funkciju, za k i niz koji se zadaju sa standardnog ulaza. Prvo se unosi k, a zatim elementi niza a, sve do kraja ulaza. Pretpostaviti da niz neće imati više od 256 elemenata.

Primer 1

```
Ulaz:
 Unesite ceo broj:
 5
 Unesite elemente niza:
 0 3 9 1 -8 19 34 5
Izlaz:
 4
```

Primer 2

```
Ulaz:
 Unesite ceo broj:
 -5
 Unesite elemente niza:
 5 6 7 8
Izlaz:
 0
```

Primer 3

```
Ulaz:
 Unesite ceo broj:
 7
 Unesite elemente niza:
 -1 -2 -6 2 3
Izlaz:
 5
```

5. Napisati rekurzivnu funkciju koja odredjuje broj pojavljivanja karaktera c u okviru niske s. Ne praviti razliku izmedju malih i velikih slova. Napisati program koji testira ovu funkciju za nisku i karakter koji se zadaju sa standardnog ulaza. Pretpostaviti da niska neće biti duža od 63 karaktera.

Primer 1

```
Ulaz:
  Unesite nisku:
  Programiranje
  Unesite karakter:
  a
Izlaz:
  2
```

Primer 2

```
Ulaz:
  Unesite nisku:
  Danas je lep dan.
  Unesite karakter:
  d
Izlaz:
  2
```

Primer 3

```
Ulaz:
  Unesite nisku:
  Jabuka, jelka, jelen
  Unesite karakter:
  J
Izlaz:
  3
```

Primer 4

```
Ulaz:
  Unesite nisku:
  Matematika
  Unesite karakter:
  R
Izlaz:
  0
```

6. Napisati rekurzivnu funkciju koja vrši modifikaciju niza pozitivnih celih brojeva na sledeći način: ukoliko se dva data pozitivna cela broja pojavljuju uzastopno u nizu a, postavlja ta dva elementa na -1. Sa standardnog ulaza učitati dva broja, a zatim elemente niza sve dok se ne unese 0. Na standardni izlaz ispisati novodobijeni niz. Pretpostaviti da neće biti uneto više od 256 brojeva.

Primer 1

```
Ulaz:
  Unesite dva cela broja:
  1 2
  Unesite elemente niza
  9 3 1 2 7 4 1 2 8 0
Izlaz:
  9 3 -1 -1 7 4 -1 -1 8
```

Primer 2

```
Ulaz:
  Unesite dva cela broja:
  3 5
  Unesite elemente niza
  3 4 5 6 7 8 0
Izlaz:
  3 4 5 6 7 8
```

Primer 3

```
Ulaz:
  Unesite dva cela broja:
  9 1
  Unesite elemente niza
  1 9 1 2 0
Izlaz:
  1 -1 -1 2
```

7. Napisati rekurzivnu funkciju koja računa broj pojava dve uzastopne jedinice u binarnom zapisu celog broja x. Napisati program koji tu funkciju testira za broj koji se zadaje sa standardnog ulaza.

Primer 1

```
Interakcija sa programom:
Ulaz:
  11
Izlaz:
  1
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  1024
Izlaz:
  0
```

Primer 3

```
Interakcija sa programom:
Ulaz:
  2147377146
Izlaz:
  22
```

4.2 Dodatni zadaci

1. Napisati rekurzivnu funkciju koja modifikuje broj x tako što mu svaku cifru, čija je vrednost neparan broj, uveća za 1. Napisati program koji testira ovu funkciju. Sa standardnog ulaza učitati jedan neoznačen broj. Na standardni izlaz ispisati novodobijeni broj.

Primer 1

```
Interakcija sa programom:
Ulaz:
  1234
Izlaz:
  2244
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  22367
Izlaz:
  22468
```

Primer 3

```
Interakcija sa programom:
Ulaz:
  333
Izlaz:
  444
```

Primer 4

```
Interakcija sa programom:
Ulaz:
  222
Izlaz:
  222
```

2. Modifikacija Hanojskih kula: Data su četiri vertikalna štapa. Na jednom se nalazi n diskova poluprečnika 1, 2, 3,... do n, tako da se najveći nalazi na dnu, a najmanji na vrhu. Ostala tri štapa su prazna. Potrebno je premestiti diskove na drugi štap tako da budu u istom redosledu, premeštajući jedan po jedan disk, pri čemu se ni u jednom trenutku ne sme staviti veći disk preko manjeg. Preostala dva štapa koristiti kao pomoćne štapove prilikom premeštanja. Napisati program koji za proizvoljnu vrednost n, koja se unosi sa standardnog ulaza, prikazuje proces premeštanja diskova.

5 V nedelja: Pretrage

5.1 Obavezni zadaci

1. U datoteci čije se ime zadaje kao prvi argument komandne linije nalaze se korisnicko ime studenta i ocene iz P1 i P2. U zavisnosti od prisustva opcija komandne linije -min ili -max ispisati korisničko ime studenta sa najvećim (-max), odnosno najmanjim (-min) prosekom ocena iz ova dva predmeta. Ako ima više studenata koji zadovoljavaju traženi uslov, ispisati prvog. U slučaju da se opcija ne navede ili da se navede neka treća opcija, na standardni izlaz za greške ispisati poruku "Morate zadati opciju -min ili -max". Pretpostaviti da u datoteci neće biti više od 1024 studenta.

Primer 1

```
Poziv: ./a.out test.txt -min

TEST.TXT
 mr15111 9 8
 mv14222 10 10
 mr14098 7 10
 mr15010 8 8
 mv15001 6 8

Izlaz:
 mv15001
```

Primer 2

```
Poziv: ./a.out dat.txt -max

DAT.TXT
 mr15111 9 8
 mv14222 10 10
 mr14098 7 10
 mr15010 8 8
 mv15001 6 8

Izlaz:
 mv14222
```

Primer 3

```
Poziv: ./a.out test.txt

Izlaz:
 Morate zadati opciju -min ili
-max
```

2. Napisati funkciju koja proverava da li u sortiranom nizu celih brojeva a, čiji su svi elementi različiti, postoji i za koje važi da je $a[i] = i$. Ukoliko postoji, funkcija treba da vrati i, u suprotnom treba da vrati -1. Napisati program koji testira ovu funkciju, za niz koji se zadaje sa standardnog ulaza. Elementi niza se unose u rastućem poretku. Učitavanje prekinuti kada se unese 0. Napomena: Nulu ne ubacivati kao poslednji član niza.

Primer 1

```
Interakcija sa programom:
Ulaz:
 -5 -1 1 3 6 7 9 0
Izlaz:
 3
```

Primer 2

```
Interakcija sa programom:
Ulaz:
 -9 -4 6 7 8 0
Izlaz:
 -1
```

Primer 3

```
Interakcija sa programom:
Ulaz:
 -100 -56 -3 -2 -1 4 5 6 7 9
19 34 0
Izlaz:
 9
```

Primer 4

```
Interakcija sa programom:
Ulaz:
 0
Izlaz:
 -1
```

3. Napisati funkciju koja određuje nulu funkcije $f(x) = sin(x) + cos(\sqrt{3}x)$ na intervalu [3, 5] metodom polovljenja intervala. Algoritam se završava kada se vrednost funkcije razlikuje za najviše 0.001 od nule.
Uputstvo: Korisiti metod polovljenja intervala (algoritam analogan algoritmu binarne pretrage).
Napomena: Ovaj metod se može primeniti na funkciju f(x) na intervalu [3, 5] zato što je ona na ovom intervalu neprekidna, i vrednosti funkcije na krajevima intervala su različitog znaka.

Primer 1

```
Izlaz:
 4.025
```

4. Napisati funkciju koja u sortiranom nizu nalazi prvi element veći od zadatog broja k (napomena: primeniti binarnu pretragu). U slučaju da takav element postoji, vratiti njegovu vrednost, u suprotnom vratiti -1. Napisati i program koji testira ovu funkciju za niz elemenata koji se zadaju kao argumenti komandne linije i broj k koji se zadaje sa standardnog ulaza. Niz neće imati više od 256 elemenata.

Primer 1

```
Poziv: ./a.out -5 -2 0 1 4 7 8 9

Interakcija sa programom:
Ulaz:
 Unesite k:
 6
Izlaz:
 7
```

Primer 2

```
Poziv: ./a.out 0 4 5 7 8

Interakcija sa programom:
Ulaz:
 Unesite k:
 -6
Izlaz:
 0
```

Primer 3

```
Poziv: ./a.out -4 5 7 9 56

Interakcija sa programom:
Ulaz:
 Unesite k:
 100
Izlaz:
 -1
```

5. Napisati funkciju koja rekurzivno implementira algoritam interpolacione pretrage i program koji ovu funkciju testira za brojeve koji se unose sa standardnog ulaza (pretpostaviti da niz brojeva koji se unosi nije duži od 1000 elemenata). Prvo se unosi broj koji se traži, a zatim sortirani elementi niza sve do kraja ulaza. Ukoliko se uneti broj nalazi u nizu, na standardni izlaz ispisati njegov indeks, u suprotnom, ispisati -1.

Primer 1

```
Ulaz:
  11 2 5 6 8 10 11 23
Izlaz:
  5
```

Primer 2

```
Ulaz:
  14 10 32 35 43 66 89 100 13
Izlaz:
  -1
```

5.2 Dodatni zadaci

1. U prvom kvadrantu dato je $1 \leq N \leq 10000$ duži svojim koordinatama (duži mogu da se seku, preklapaju, itd.). Napisati program koji pronalazi najmanji ugao $0 \leq \alpha \leq 90$, na dve decimale, takav da je suma dužina duži sa obe strane polupoluprave iz koordinatnog početka pod uglom α jednaka (neke duži bivaju presečene, a neke ne). (Uputstvo: vršiti binarnu pretragu intervala $[0, 90°]$).

Primer 1

```
Interakcija sa programom:
Ulaz:
  2
  2 0 2 1
  1 2 2 2
Izlaz:
  26.57
```

6 VI nedelja: Sortiranje

6.1 Obavezni zadaci

1. Napisati funkciju koja u datom nizu sortiranih brojeva pronalazi dva susedna elementa čije se vrednosti razlikuju za k i vraća njihov zbir. Ukoliko ima više takvih, izabrati dva najveća. U slučaju da takav par ne postoji, vratiti -1. Niz se zadaje sa standardnog ulaza, sve do kraja ulaza i neće imati više od 256 elemenata. Broj k se zadaje kao argument komandne linije. (Napomena: prvo sortirati niz, pa zatim pozvati funkciju.)

Primer 1

```
Poziv: ./a.out 5

Interakcija sa programom:
Ulaz:
  2 5 6 -1 11 4 98
Izlaz:
  17
```

Primer 2

```
Poziv: ./a.out 3

Interakcija sa programom:
Ulaz:
  4 7 1 9 8 92 93 89 -5 -6 -8
Izlaz:
  181
```

Primer 3

```
Poziv: ./a.out 17

Interakcija sa programom:
Ulaz:
  1 5 9 -5
Izlaz:
  -1
```

Primer 4

```
Poziv: ./a.out

Izlaz:
  Neispravan poziv programa!
```

2. Napisati funkciju koja sortira slova prosledjene niske s. Napisati program koji pronalazi karakter koji se najviše puta pojavljuje u okviru niske s. Ako ima više takvih, ispisati prvi. Razlikovati mala i velika slova. Niska s se zadaje sa standardnog ulaza i neće biti duža od 128 karaktera.

Primer 1

```
Interakcija sa programom:
Ulaz:
  danasJeLepDan
Izlaz:
  a
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  miVolimoDaProgramiramo
Izlaz:
  m
```

Primer 3

```
Interakcija sa programom:
Ulaz:
  dobroJutroSvima
Izlaz:
  o
```

Primer 4

```
Interakcija sa programom:
Ulaz:
  ababab
Izlaz:
  a
```

3. Napisati program koji ispisuje sve elemente niza koji se u njemu pojavljuju tačno 3 puta. (Napomena: prvo sortirati niz, a zatim pronaći tražene elemente). Niz se zadaje sa standardnog ulaza, sve do kraja ulaza i neće imati više od 256 elemenata. U slučaju da nema elemenata koji zadovoljavaju traženo svojstvo ispisati poruku "U nizu nema elemenata koji se pojavljuju tačno 3 puta."

Primer 1

```
Interakcija sa programom:
Ulaz:
  3 6 1 8 3 9 8 1 3 6 1
Izlaz:
  1 3
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  -6 -7 4 2 9
Izlaz:
  U nizu nema elemenata koji se pojavljuju tacno 3
puta.
```

Primer 3

```
Interakcija sa programom:
Ulaz:
  4 5 -1 3 9 -1 3 -1
Izlaz:
  -1
```

Primer 4

```
Interakcija sa programom:
Ulaz:
  9 9 9
Izlaz:
  9
```

Primer 4

```
Interakcija sa programom:
Ulaz:
  -6 4
Izlaz:
  U nizu nema elemenata koji se pojavljuju tacno 3
puta.
```

4. Napisati funkciju koja proverava da li u celobrojnom nizu a postoje dva elementa a_i i a_j za koja važi da je $p * a_i + q * a_j = w$, pri čemu se celi brojevi p,q i w zadaju kao argumenti komandne linije i važi $-1 \leq p,q \leq 1$. Niz učitavati sa standardnog ulaza, sve do unosa 0. Pretpostaviti da niz neće imati više od 256 elemenata. Napomena: Prvo sortirati niz, pa iskoristiti binarnu pretragu. Voditi računa da p i/ili q mogu imati vrednost 0!

Primer 1

```
Poziv: ./a.out -1 1 5

Interakcija sa programom:
Ulaz:
  2 3 5 6 9 1 4 5 0
Izlaz:
  da
```

Primer 2

```
Poziv: ./a.out 1 0 3

Interakcija sa programom:
Ulaz:
  4 1 9 -5 -6 0
Izlaz:
  ne
```

Primer 3

```
Poziv: ./a.out 0 -1 9

Interakcija sa programom:
Ulaz:
  5 -9 6 0
Izlaz:
  da
```

Primer 4

```
Poziv: ./a.out 0 0 3

Interakcija sa programom:
Ulaz:
  3 -3 1 0
Izlaz:
  ne
```

5. Napisati funkciju koja prima dva sortirana niza, i na osnovu njih pravi novi sortirani niz koji koji sadrži elemente oba niza $int\ merge(int\ *niz1,\ int\ dim1,\ int\ *niz2,\ int\ dim2,\ int\ *niz3,\ int\ dim3)$. Treća dimenzija predstavlja veličinu niza u koji se smešta rezultat. Ako je ona manja od potrebne dužine, funkcija vraća -1, kao indikator neuspeha, inače vraća 0. Napisati i program koji testira funkciju, u kome se nizovi unose sa standardnog ulaza, sve dok se ne unese 0.

Primer 1

```
Ulaz:
   3 6 7 11 14 35 0 3 5 8 0
Izlaz:
   3 3 5 6 7 8 11 14 35
```

Primer 2

```
Ulaz:
  1 4 7 0 9 11 23 54 75 0
Izlaz:
   1 4 7 9 11 23 54 75
```

6.2 Dodatni zadaci

1. ZADATAK SA TAKMIČENJA MATF 2016: Medijana niza sa neparnim brojem elemenata predstavlja onaj element koji ima srednji indeks u nizu koji se dobija sortiranjem polaznog. Na primer, za niz 5,1,17,107,999 medijana je 17, a za niz 1, 1, 1, 99, 2, medijana je 1.
Sa standardnog ulaza se u prvom redu učitava broj n, $1 \leq n \leq 100000$. U narednih n redova se redom učitavaju elementi niza, koji predstavljaju realne brojeve. Na standardni izlaz je potrebno, u svakom trenutku kada je broj trenutnih učitanih elemenata neparan, ispisati koja je medijana trenutnog niza (zapisano na 3 decimale).

Primer 1

```
Interakcija sa programom:
Ulaz:
  5
  17
  1
  1
  3
  5
Izlaz:
  17.000
  1.000
  3.000
```

7 VII nedelja: Sortiranje , Zadaci sa kolokvijuma

7.1 Obavezni zadaci

1. Napisati funkcije koje sortiraju niz duži po:

   (a) njihovoj dužini

   (b) koeficijentu pravca prave na kojoj leži data duž

   Napisati program koji učitava niz duži zadate dvema krajnjim tačkama čije se celobrojne x i y koordinate redom nalaze u datoteci čije se ime zadaje kao argument komandne linije, i u zavisnosti od prisutnih opcija u komandnoj liniji (*-d* označava sortiranje po dužini, a *-k* po koeficijentu pravca) sortira duži po jednom od prethodna dva kriterijum i rezultat upisuje u datoteku čije se ime zadaje kao drugi argument komandne linije. U ulaznoj datoteci nije zadato više od 100 duži. Pretpostaviti da će duži biti zadate tako da je njihov koeficijent pravca definisan. U slučaju da datoteka ne postoji ili nije zadata opcija ispravno na standardni izlaz ispisati poruku o grešci.

Primer 1

```
Poziv: ./a.out -d duzi.txt
sortduzi.txt

DUZI.TXT
 11 9 10 5
 0 0 5 5
 0 -4 3 -4

SORTDUZI.TXT
 0 -4 3 -4
 11 9 10 5
 0 0 5 5
```

Primer 2

```
Poziv: ./a.out -k duzi.txt
sortduzi.txt

DUZI.TXT
 11 9 11 5
 0 0 5 5
 0 -4 3 -4

SORTDUZI.TXT
 0 -4 3 -4
 0 0 5 5
 11 9 10 5
```

Primer 3

```
Poziv: ./a.out -k duzi1.txt
sortduzi.txt

Izlaz:
 Datoteka duzi1.txt ne
postoji.
```

Primer 3

```
Poziv: ./a.out -r duzi1.txt
sortduzi.txt

Izlaz:
 Morate zadati jednu od opcija
-k ili -d.
```

2. Definisati strukturu koja čuva imena, prezimena i godišta dece. Napisati funkciju koja sortira niz dece po godištu, a decu istog godišta sortira leksikografski po prezimenu i imenu. Napisati program koji učitava podatke o deci koji se nalaze u datoteci čije se ime zadaje kao prvi argument komandne linije, sortira ih i sortirani niz upisuje u datoteku čije se ime zadaje kao drugi argument komandne linije. Pretpostaviti da u ulaznoj datoteci nisu zadati podaci o više od 128 dece i da su imena i prezimena niske karaktera koje nisu duže od 30 karaktera. Ukoliko nije zadato dovoljno argumenata komandne linije ili jedna od datoteka ne postoji, na standardni izlaz ispisati poruku o grešci.

Primer 1

```
Poziv: ./a.out in.txt out.txt

IN.TXT
 Petar Petrovic 2007
 Milica Antonic 2008
 Ana Petrovic 2007
 Ivana Ivanovic 2009
 Dragana Markovic 2010
 Marija Antic 2007

OUT.TXT
 Marija Antic 2007
 Ana Petrovic 2007
 Petar Petrovic 2007
 Milica Antonic 2008
 Ivana Ivanovic 2009
 Dragana Markovic 2010
```

Primer 2

```
Poziv: ./a.out in.txt

Izlaz:
 Pogrešan broj argumenata komandne linije.
```

Primer 3

```
Poziv: ./a.out in1.txt out.txt

Izlaz:
 Datoteka in1.txt ne postoji.
```

3. Napisati funkciju koja sortira niz brojeva po broju jedinica u njihovom binarnom zapisu. Ukoliko imaju isti broj jedinica u binarnom zapisu tada ih sortirati po vrednosti. Napisati program koji testira ovu funkciju za brojeve koji se zadaju u datoteci *brojevi.txt* i na standardni izlaz ispisuje sortirane brojeve. Pretpostaviti da u nizu nema više od 100 elemenata.

Primer 1

```
Poziv: ./a.out

BROJEVI.TXT
 4831 -249561075 123 33489407 -55 16908287

Izlaz:
 123 -249561075 4831 16908287 33489407 -55
```

7.2 Dodatni zadaci

1. Razmatrajmo dve operacije: operacija U je unos novog broja x , a operacija N određivanje n -tog po veličini od unetih brojeva. Implementirati program koji izvršava ove operacije. Može postojati najviše 100000 operacija unosa, a uneti elementi se mogu ponavljati, pri čemu se i ponavljanja računaju prilikom brojanja. Optimizovati program, ukoliko se zna da neće biti više od 500 različitih unetih brojeva. *Napomena: Brojeve čuvati u sortiranom nizu i svaki naredni element umetati na svoje mesto.*

Primer 1

```
Interakcija sa programom:
Ulaz:
 U 2 U 0 U 6 U 4 N 1 U 8 N 2 N 5 U 2 N 3 N 5
Izlaz:
 0 2 8 2 6
```

2. Šef u restoranu je neuredan i palačinke koje ispeče ne slaže redom po veličini. Konobar pre serviranja mora da sortira palačinke po veličini, a jedina operacija koju sme da izvodi je da obrne deo palačinki. Na primer:

```
3     5     2     1
4     4     1__   2
5__   3     3     3
1     1     4     4
2     2__   5     5
```

Po kolonama su predstavljene naslagane palačinke posle svakog okretanja. Na početku, palačinka veličine 2 je na dnu, iznad nje se redom nalaze najmanja, najveća, itd... Na slici crtica predstavlja mesto iznad koga će konobar okrenuti palačinke. Prvi potez konobara je okretanje palačinki veličine 5, 4 i 3 (prva kolona), tada će veličine palačinki odozdo nagore biti 2, 1, 3, 4, 5 (druga kolona). Posle još dva okretanja, palačinke će biti složene. Napisati program koji u najviše $2n-3$ okretanja sortira učitani niz. *Uputstvo: imitirati selection sort i u svakom koraku dovesti jednu palačinku na svoje mesto korišćenjem najviše dva okretanja.)*

Primer 1

```
Interakcija sa programom:
Ulaz:
 23 64 123 76 22 7 34 123 54562 12 453 342 5342 42 542 1 3 432 1 32 43
Izlaz:
 1 1 3 7 12 22 23 32 34 42 43 64 76 123 123 342 432 453 542 5342 54562
```

7.3 Zadaci sa kolokvijuma

1. Napisati funkciju `int najmanje_k_1(int x, int k)` koja proverava da li se u binarnom zapisu broja x uzastopno pojavljuje najmanje k jedinica. Napisati program koji učitava ceo broj x u heksadekadnom formatu sa standardnog ulaza i testira napisanu funkciju. Pozitivan broj k se zadaje kao prvi argument komandne linije. Ukoliko u broju x postoji sekvenca od najmanje k jedinica, na standardni izlaz ispisati 1, inače ispisati 0. U slučaju greške na standardni izlaz za greške ispisati -1.

	Primer 1:	Primer 2:	Primer 3:	Primer 4:	Primer 5:	Primer 6:
Poziv:	./a.out 2	./a.out 4	./a.out 4	./a.out 16	./a.out -5	./a.out
Ulaz:	0x3	0x7	0x7c0	0xfffedc5a	0x3e0	0xfffffff2
Izlaz:	1	0	1	0	-1	-1

2. Napisati rekurzivnu funkciju `int broj_pluseva(char niska[])` koja određuje broj pojavljivanja karaktera + u datoj niski. Napisati program koji sa standardnog ulaza učitava nisku, testira napisanu funkciju i ispisuje rezultat na standardni izlaz.

   Napomena: Zadatak rešen iterativno, bez korišćenja rekurzije, nosi maksimalno 40% poena.

	Primer 1:	Primer 2:	Primer 3:	Primer 4:
Ulaz:	Pro+lecE	(b+c)*m*(n+1)	petAk	+++Antananarive+++
Rezultat:	1	2	0	6

3. Napisati strukturu `Drzava` sa podacima o nazivu države i broju glasova na Evroviziji. Pretpostaviti da je naziv države niska sa najviše 30 karaktera, a broj glasova neoznačen ceo broj.

   a) Napisati funkciju `void sort(Drzava niz[], int n)` koja sortira niz država po broju glasova na Evroviziji opadajuće, a ukoliko dve države imaju isti broj glasova, sortira ih leksikografski rastuće po nazivu države.

   b) Napisati funkciju `void pretraga(Drzava niz[], int n, unsigned broj)` koja binarnom pretragom pronalazi državu koja ima zadat broj glasova na Evroviziji. Ukoliko dve države imaju isti broj glasova, vratiti bilo koju.

   Napisati program koji učitava podatke o državama iz datoteke *drzave.txt* u kojoj se nalazi najviše 50 država i sortira ih funkcijom iz dela pod a). Nakon toga učitava broj glasova sa standardnog ulaza i korišćenjem funkcije iz dela pod b) pronalazi državu koja je imala zadat broj glasova i ispisuje njen naziv na standarni izlaz. U slučaju greške na standardni izlaz za greške ispisati -1.

Primer 1:

Sadržaj datoteke *drzave.txt* :
Norveska 102
Slovenija 39
Italija 292
Estonija 106
Rusija 303
Spanija 15
Australija 102

Ulaz: 106
Izlaz: Estonija

Primer 2:

Datoteka drzave.txt ne postoji

Izlaz: -1

Primer 3:

Sadržaj datoteke *drzave.txt* :
Norveska 102
Slovenija 39
Italija 292
Estonija 106
Rusija 303
Spanija 15
Australija 196

Ulaz: 100
Izlaz: -1

4. Napisati funkciju `int najmanje_k_0(int x, int k)` koja proverava da li se u binarnom zapisu broja x uzastopno pojavljuje najmanje k nula. Napisati program koji učitava ceo broj x u heksadekadnom formatu sa standardnog ulaza i testira napisanu funkciju. Pozitivan broj k se zadaje kao prvi argument komandne linije. Ukoliko u broju x postoji sekvenca od najmanje k nula, na standardni izlaz ispisati 1, inače ispisati 0. U slučaju greške na standardni izlaz za greške ispisati -1.

	Primer 1:	Primer 2:	Primer 3:	Primer 4:	Primer 5:	Primer 6:
Poziv:	./a.out 5	./a.out 31	./a.out 6	./a.out 11	./a.out -8	./a.out
Ulaz:	0x3	0x7	0xffffcc1	0xab4c	0xfffffdb	0x9fb
Izlaz:	1	0	0	1	-1	-1

5. Napisati rekurzivnu funkciju `int razliciti(char niska[])` koja određuje broj pojavljivanja karaktera različitih od karaktera $*$ u datoj niski. Napisati program koji sa standardnog ulaza učitava nisku, testira napisanu funkciju i ispisuje rezultat na standardni izlaz.

   Napomena: Zadatak rešen iterativno, bez korišćenja rekurzije, nosi maksimalno 40% poena.

	Primer 1:	Primer 2:	Primer 3:	Primer 4:
Ulaz:	a*a*b	(a+b)*c*d	*****	***kolokvijum***
Rezultat:	3	7	0	10

6. Napisati strukturu `Grad` sa podacima o nazivu grada i broju stanovnika. Pretpostaviti da je naziv grada niska sa najviše 30 karaktera, a broj stanovnika neoznačen ceo broj.

   a) Napisati funkciju `void sort(Grad niz[], int n)` koja sortira niz gradova po broju stanovnika rastuće, ukoliko više gradova ima isti broj stanovnika sortirati ih leksikografski rastuće po nazivu.

   b) Napisati funkciju `void pretraga(Grad niz[], int n, unsigned broj)` koja binarnom pretragom pronalazi grad sa zadatim brojem stanovnika. Ukoliko dva grada imaju isti broj stanovnika vratiti bilo koji.

   Napisati program koji učitava podatke o gradovima iz datoteke *gradovi.txt* u kojoj se nalazi najviše 50 gradova i sortira ih funkcijom iz dela pod a). Nakon toga učitava broj sa standardnog ulaza i korišćenjem funkcije iz dela pod b) pronalazi grad sa tim brojem stanovnika i ispisuje njegov naziv na standarni izlaz. U slučaju greške na standardni izlaz za greške ispisati -1.

Primer 1:

Sadržaj datoteke *gradovi.txt* :
Arandjelovac 34300
Valjevo 44000
Krusevac 39300
Raca 18440
Negotin 39300
Ub 14210
Smederevo 31300
Pancevo 26101

Ulaz: 31300
Izlaz: Smederevo

Primer 2:

Datoteka gradovi.txt ne postoji

Izlaz: -1

Primer 3:

Sadržaj datoteke *gradovi.txt* :
Arandjelovac 34300
Valjevo 44000
Krusevac 39300
Raca 18440
Negotin 39300
Ub 14210
Smederevo 31300
Pancevo 26101

Ulaz: 11000
Izlaz: -1

8 VIII nedelja - Pokazivači

8.1 Obavezni zadaci

1. Koristeći pokazivačku sintaksu, prebrojati koliko argumenata komandne linije sadrže bar jednu cifru.

Primer 1

```
Poziv: ./a.out programiranje2 nemacifre lala12lala jasamargumentkomandnelinije 0201 x*2

Izlaz:
  4
```

2. Napisati program koji na standardni izlaz ispisuje koliko se puta niska, koja se zadaje kao drugi argument komandne linije, pojavljuje u datoteci čije se ime zadaje kao prvi argument komandne linije. U zadatku ne koristiti ugradjene funkcije za rad sa stringovima, već implementirati svoje koristeći pokazivačku sintaksu.

Primer 1

```
Poziv: ./a.out datoteka.txt kinder

datoteka.txt
  Sva deca vole kinder jaja.  Baka uvek donese kinder cokoladice kada dolazi u posetu.  Omiljeni slatkis mog brata
je kinder bueno.  Ja sam bas slatka datoteka.  Za rodjendan se sprema kinder torta.

Izlaz:
  4
```

3. Napisati program koji kao prvi argument komandne linije prihvata putanju do datoteke u kojoj treba proveriti da li se karakter c pojavljuje uzastopno tačno k puta. Karakter c se zadaje kao drugi argument komandne linije a neoznačen ceo broj k kao treći argument komandne linije. Zadatak uraditi korišćenjem pokazivačke aritmetike.

Primer 1

```
Poziv: ./a.out datoteka.txt z 8

datoteka.txt
  Zzzzzzzzz, danas mi se bas spava.  Muva zuji
zzzzz.

Izlaz:
  Da.
```

Primer 2

```
Poziv: ./a.out datoteka.txt 1 3

datoteka.txt
  1*2+11/(42+x+log(45)) = y-1111.1/51.2

Izlaz:
  Ne.
```

Primer 3

```

Poziv: ./a.out datoteka.txt ?  1

datoteka.txt
  Ja sam datoteka u kojoj nema pitanja.

Izlaz:
  Ne.
```

8.2 Dodatni zadaci

1. Napisati program koji sa standardnog ulaza učitava ceo broj k a nakon toga pozitivan ceo broj n i matricu celih brojeva M dimenzija $n \times n$. Napisati funkciju koja množi matricu M skalarom k. Napisati program koji testira napisanu funkciju i ispisuje rezultujuću matricu na standardni izlaz.

Primer 1

```
Interakcija sa programom:
Ulaz:
  -1
  4
  1 2 3 4
  1 1 1 1
  4 3 2 1
  1 1 1 1
Izlaz:
  -1 -2 -3 -4
  -1 -1 -1 -1
  -4 -3 -2 -1
  -1 -1 -1 -1
```

9 IX nedelja - Matrice

9.1 Obavezni zadaci

1. (a) Napisati funkciju koja proverava da li je data matrica gornje trougaona.
   (b) Napisati funkciju koja proverava da li je data matrica donje trougaona.

(c) Napisati funkciju koja kao rezultat vraća transponovanu matricu date matrice.

Napisati program koji učitava matricu iz datoteke čije se ime zadaje kao prvi argument komandne linije, a zatim testira napisane funkcije. U prvoj liniji datoteke nalazi se ceo broj n koji predstavlja dimenziju matrice, a potom i sami elementi. Pretpostaviti da dimenzija neće biti veća od 100.

Primer 1

```
Poziv: ./a.out matrica.txt

matrica.txt
 4
 9 1 8 2
 0 1 4 3
 0 0 1 1
 0 0 0 -5

Izlaz:
 Matrica jeste gornje trougaona.
 Matrica nije donje trougaona.
 Transponovana matrica:
 9 0 0 0
 1 1 0 0
 8 4 1 0
 2 3 1 -5
```

2. Napisati program koji sa standardnog ulaza učiitava ceo broj n ne veći od 100 koji predstavlja dimenziju matrice, a nakon toga matricu dimenzija $n \times n$ i na standardni izlaz ispisuje:

   (a) Proizvod svih nenula elementa matrice na sporednoj dijagonali

   (b) Indeks vrste koja sadrži najviše negativnih elemenata

   (c) Indeks kolone koja sadrži najviše elemenata koji su jednaki nuli

   Smatrati da indeksiranje kreće od 0.

Primer 1

```
Interakcija sa programom:
Ulaz:
 4
  2 0 9 0
 -1 4 -7 0
 -4 0 1 0
  2 2 2 3
Izlaz:
 Proizvod nenula elemenata na sporednoj dijagonali:  -14.
 Indeks vrste koja sadrži najviše negativnih elemenata:  1
 Indeks kolone koja sadrži najviše 0:  3
```

3. Sa standardnog ulaza učiitava se dimenzija matrice, ceo broj n ne veći od 100, a nakon toga dve matrice A i B dimenzija $n \times n$. Napisati program koji proverava da li su matrice A i B komutativne. Matrice su komutativne ukoliko važi $A * B = B * A$.

Primer 1

```
Interakcija sa programom:
Ulaz:
 2
 1 2
 3 4
 1 4
 6 7
Izlaz:
 Jesu.
```

Primer 2

```
Interakcija sa programom:
Ulaz:
 2
 1 2
 3 4
 1 9
 8 3
Izlaz:
 Nisu.
```

4. Sa standardnog ulaza učitava se dimenzija matrice A, ceo broj n ne veći od 100 i matrica A, a nakon toga dimenzija matrice B, ceo broj m ne veći od 100 i matrica B. Napisati program koji proverava da li je matrica B podmatrica matrice A i ukoliko jeste na standardni izlaz ispisuje poziciju, uređen par indeksa (i,j), prvog elementa matrice B u matrici A. Smatrati da indeksiranje kreće od 0.

Primer 1

```
Interakcija sa programom:
Ulaz:
  6
  4 1 2 4 4 1
  0 3 1 2 9 0
  9 4 1 6 1 2
  5 0 1 9 3 4
  6 1 2 5 8 0


  2
  1 2
  3 4

Izlaz:
  Matrica B jeste podmatrica matrice A. Indeks
početnog elementa je (2,4).
```

9.2 Dodatni zadaci

1. Napisati funkciju koja izračunava k-ti stepen kvadratne matrice dimenzije $n \times n$ ($n <= 32$). Voditi računa da se prilikom stepenovanja izvrši što manji broj množenja. Napisati i program koji testira datu funkciju, za n, elemente matrice i k koji se unose sa standardnog ulaza.

Primer 1

```
Interakcija sa programom:
Ulaz:
  3
  1 2 3
  4 5 6
  7 8 9
  8
Izlaz:
  510008400 626654232 743300064
  1154967822 1419124617 1683281412
  1799927244 2211595002 2623262760
```

10 X nedelja - Dinamička alokacija memorije

10.1 Obavezni zadaci

1. Napisati funkciju `int palindrom(int *a, int n)` koja ispituje da li je dati niz celih brojeva palindrom. Napisati program koji sa standardnog ulaza učitava dimenziju niza celih brojeva, a zatim i njegove elemente. Ne praviti nikakve pretpostavke o dimenziji niza.

Primer 1

```
Interakcija sa programom:
Ulaz:
  Unesite broj elemenata:
  7
  Unesite elemente:
  1 2 3 4 3 2 1
Izlaz:
  Da.
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  Unesite broj elemenata:
  11
  Unesite elemente:
  1 2 3 15 4 42 3 2 1 -1 55
Izlaz:
  Ne.
```

2. Napisati funkciju `double *proizvod(double *a, double *b, int n)` koja množi n-dimenzione vektore a i b po koordinatama i vraća rezultujući vektor. Napisati program koji sa standardnog ulaza učitava broj n, a zatim i koordinate oba vektora. Vektore predstaviti nizovima realnih brojeva. Ne praviti nikakve pretpostavke o dimenziji nizova.

Primer 1

```
Interakcija sa programom:
Ulaz:
  Unesite n:
  4
  Unesite koordinate prvog vektora:
  -1.3 2 6 -1.2
  Unesite koordinate drugog vektora:
  0 1.5 -2.4 1
Izlaz:
  Rezultujući vektor:
  0.00 3.00 -14.40 -1.20
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  Unesite n:
  3
  Unesite koordinate prvog vektora:
  -5 4 8
  Unesite koordinate drugog vektora:
  5 -4 -8
Izlaz:
  Rezultujući vektor:
  -25.00 -16.00 -64.00
```

3. Napisati funkciju `char *ponovi_k(char *s, int k)` koja kao rezultat vraća nisku koja se dobija ponavljanjem niske $k \geq 1$ puta. Niska i broj ponavljanja zadaju se kao argumenti komandne linije. Na standardni izlaz ispisati nisku koja se dobija kao rezultat poziva funkcije. Za rezultujuću nisku dinamički alocirati memoriju.

Primer 1

```
Interakcija sa programom:
Poziv: ./a.out dan 3
Izlaz:
  dandandan
```

Primer 2

```
Interakcija sa programom:
Poziv: ./a.out lema 8
Izlaz:
  lemalemalemalemalemalemalemalema
```

Primer 3

```
Interakcija sa programom:
Poziv: ./a.out zzz -3
Izlaz:
  -1
```

Primer 4

```
Interakcija sa programom:
Poziv: ./a.out sunce 1
Izlaz:
  sunce
```

4. Za zadatu matricu celih brojeva dimenzije $n \times m$ napisati funkciju koja izračunava redni broj vrste matrice čiji je proizvod minimalan. Napisati program koji testira ovu funkciju. Sa standardnog ulaza učitati dimenzije matrice n i m (ne praviti nikakve pretpostavke o njihovoj veličini), a zatim elemente matrice. Na standardni izlaz ispisati redni broj vrste matrice sa minimalnim proizvodom. Ukoliko ima više takvih, ispisati redni broj poslednje.

Primer 1

```
Interakcija sa programom:
Ulaz:
  Unesite broj vrsta i broj kolona:
  3 4
  Unesite elemente matrice po vrstama:
  1 2 3 4
  4 -5 6 7
  7 3 8 9
Izlaz:
  2.  vrsta ima najmanji proizvod.
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  Unesite broj vrsta i broj kolona:
  4 2
  Unesite elemente matrice po vrstama:
  1 2
  4 -5
  7 3
  2 -10
Izlaz:
  4.  vrsta ima najmanji proizvod.
```

5. Napisati program koji na osnovu dve celobrojne matrice dimenzije $n \times m$ formira matricu dimenzije $n \times 2 * m$ tako što naizmenično kombinuje jednu kolonu prve matrice i jednu kolonu druge matrice. Sa standardnog ulaza učitati n i m (ne praviti nikakve pretpostavke o dimenziji matrica), a zatim elemente obe matrice. Rezultujuću matricu ispisati na standardni izlaz.

Primer 1

```
Interakcija sa programom:
Ulaz:
  Unesite broj vrsta i broj kolona:
  4 2
  Unesite elemente prve matrice po vrstama:
  1 2
  4 5
  7 3
  2 1
  Unesite elemente druge matrice po vrstama:
  7 4
  8 6
  2 1
  2 3
Izlaz:
  1 7 2 4
  4 8 5 6
  7 2 3 1
  2 2 1 3
```

10.2 Dodatni zadaci

1. Napisati funkciju `int ortogonalnost(int **matrica, int n)` koja ispituje da li je data kvadratna matrica ortogonalna. Sa standardnog ulaza učitati n (ne praviti nikakve pretpostavke o dimenziji matrice), a zatim elemente matrice po vrstama.
   Napomena: kvadratna matrica A je ortogonalna ukoliko važi da je $A \times A^T = I$

Primer 1

```
Interakcija sa programom:
Ulaz:
  Unesite dimenziju matrice:
  2
  Unesite elemente matrice po vrstama:
  0 1
  1 0
Izlaz:
  Da.
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  Unesite dimenziju matrice:
  3
  Unesite elemente matrice po vrstama:
  1 1 0
  3 -1 5
  4 -5 8
Izlaz:
  Ne.
```

11 XI nedelja - Pokazivači na funkcije. Bibliotečke funkcije pretrage i sortiranja.

11.1 Obavezni zadaci

1. Napisati program koji tabelarno štampa vrednosti proizvoljne realne funkcije sa jednim realnim argumentom, odnosno izračunava i ispisuje vrednosti date funkcije u n ekvidistantnih tačaka na intervalu $[a, b]$ sa dvostrukom preciznošću. Realni brojevi a i b ($a < b$), kao i ceo broj n ($n \geq 2$), učitavaju se sa standardnog ulaza.
   Funkcija se bira opcijom koja se navodi kao argument komandne linije:
   - Opcija `-exp`:
     $$f(x) = \frac{1}{1 + e^x}$$
   - Opcija `-esin`:
     $$f(x) = \frac{sin^2(x)}{1 + e^x}$$
   - Opcija `-ecos`:
     $$f(x) = \frac{cos(x) + x}{1 + e^x}$$

   Napomena: Konstanta sa vrednošću broja e definisana je u zaglavlju math.h pod nazivom M_E.

Primer 1

```
Poziv: ./a.out -exp

Interakcija sa programom:
Ulaz:
  Unesite krajeve intervala:
  -1 1
  Koliko tacaka ima na ekvidistantnoj
  mrezi (ukljucujuci krajeve intervala)?
  3
Izlaz:
  0.73105857863
  0.500000000000
  0.26894142137
```

Primer 2

```
Poziv: ./a.out -exp

Interakcija sa programom:
Ulaz:
  Unesite krajeve intervala:
  -5 -2
  Koliko tacaka ima na ekvidistantnoj
  mrezi (ukljucujuci krajeve intervala)?
  4
Izlaz:
  0.993307149076
  0.982013790038
  0.952574126822
  0.880797077978
```

Primer 3

```
Poziv: ./a.out -esin

Interakcija sa programom:
Ulaz:
  Unesite krajeve intervala:
  -1 1
  Koliko tacaka ima na ekvidistantnoj
  mrezi (ukljucujuci krajeve intervala)?
  3
Izlaz:
  0.517643146729
  0.000000000000
  0.190430271545
```

Primer 4

```
Poziv: ./a.out -ecos

Interakcija sa programom:
Ulaz:
  Unesite krajeve intervala:
  -1 2
  Koliko tacaka ima na ekvidistantnoj
  mrezi (ukljucujuci krajeve intervala)?
  4
Izlaz:
  -0.336065942872
  0.500000000000
  0.41425109148
  0.188799925138
```

2. Korišćenjem bibliotečke funkcije `qsort` napisati program koji sortira niz niski po sledećim kriterijumima:

   (a) po broju samoglasnika opadajuće,

   (b) po broju malih slova opadajuće,

   (c) po broju velikih slova rastuće.

   Niske se učitavaju iz datoteke `niske.txt`. Pretpostaviti da datoteka ne sadrži više od 1000 niski kao i da je svaka niska dužine najviše 30 karaktera. Rezultate svih sortiranja ispisivati na standardni izlaz.

Primer 1

```
NISKE.TXT
  jaFFa miLka keks CokOLadA VANILA nutEla kOkOs LeTo djumbir AnAnAs

Izlaz:
  Po broju samoglasnika opadajuce:
  CokOLadA VANILA nutEla AnAnAs jaFFa miLka kOkOs LeTo djumbir keks

  Po broju malih slova opadajuce:
  djumbir nutEla miLka keks CokOLadA jaFFa AnAnAs LeTo VANILA

  Po broju velikih slova rastuce:
  CokOLadA djumbir miLka nutEla jaFFa kOkOs LeTo AnAnAs CokOLadA VANILA
```

3. Uraditi prethodni zadatak sa dinamički alociranim niskama i sortiranjem niza pokazivača, umesto niza niski.

4. Napisati program koji korišćenjem bibliotečke funkcije `qsort` sortira studente prema broju poena osvojenih za aktivnost na praktikumu. Ukoliko više studenata ima isti broj poena za aktivnost, sortirati ih po imenu leksikografski opadajuće. Korisnik potom unosi broj poena i prikazuje mu se jedan od studenata sa tim brojem poena ili poruka ukoliko nema takvog. Potom, sa standardnog ulaza, unosi se ime traženog studenta i prikazuje se osoba sa tim imenom ili poruka da se nijedan student tako ne zove. Za pretraživanje koristiti odgovarajuće bibliotečke funkcije. Podaci o studentima čitaju se iz datoteke čije se ime zadaje preko argumenata komandne linije. Za svakog studenta u datoteci postoje podaci o imenu, prezimenu i osvojenim poenima za aktivnost. Pretpostaviti da neće biti više od 500 studenata i da je za ime i prezime svakog studenta dovoljno po 20 karaktera.

Primer 1

```
AKTIVNOST.TXT
  Sara Popadic 4.2
  Dragana Stankovic 4.3
  Nenad Stankov 2.0
  Ivona Stefanovic 4.3
  Aleksa Borovic 0.8
  Nevena Avramov 2.6
  Milan Vuletic 0.8
  Branislav Ilijin 3.5

Poziv: ./a.out aktivnost.txt

Interakcija sa programom:
  Unesite broj poena:
  4.3
  Student:  Ivona Stefanovic 4.3

  Unesite ime:
  Srecko
  Student sa tim imenom se ne nalazi na spisku.
```

Primer 2

```
STUDENTI.TXT
  Tamara Radovanovic 3.5
  Radomir Djokovic 2.8
  Nikola Cimbaljevic 3.5
  Jelena Losic 4.8

Poziv: ./a.out studenti.txt

Interakcija sa programom:
  Unesite broj poena:
  3.7
  Student sa tim brojem poena
  se ne nalazi na spisku.

  Unesite ime:
  Tamara
  Tamara Radovanovic 3.5
```

11.2 Dodatni zadaci

1. Napisati funkciju koja određuje integral funkcije $f(x)$ na intervalu $[a, b]$. Adresa funkcije f se prenosi kao parametar. Integral se računa prema formuli:

$$\int_b^a f(x) = h \cdot (\frac{f(a) + f(b)}{2} + \sum_{i=1}^{n} f(a + i \cdot h))$$

   Vrednost h se izračunava po formuli $h = \frac{b-a}{n}$, dok se vrednosti n, a i b unose sa standardnog ulaza kao i ime funkcije iz zaglavlja `math.h`. Na standardni izlaz ispisati vrednost integrala.

Primer 1

```
Interakcija sa programom:
Ulaz:
  Unesite ime funkcije, n, a i b:
  cos 6000 -1.5 3.5
Izlaz:
  Vrednost integrala je 0.645931.
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  Unesite ime funkcije, n, a i b:
  sin 10000 -5.2 2.1
Izlaz:
  Vrednost integrala je 0.973993.
```

12 XII nedelja - Liste

12.1 Obavezni zadaci

1. Napisati biblioteku za rad sa jednostruko povezanom listom čiji čvorovi sadrže cele brojeve:

   (a) Definisati strukturu *Cvor* kojom se predstavlja čvor liste, a koja sadrži ceo broj *vrednost* i pokazivač na sledeći čvor liste.

   (b) *Cvor *napravi_cvor(int broj)* koja kao argument dobija ceo broj, kreira nov čvor liste, inicijalizuje mu polja i vraća njegovu adresu.

   (c) *int dodaj_na_kraj_liste(Cvor ** adresa_glave, int broj)* koja dodaje novi čvor sa vrednošću *broj* na kraj liste. Funkcija treba da vrati 0, ako je sve bilo u redu, odnosno 1, ako se dogodila greška prilikom alokacije memorije za nov čvor.

   (d) *void oslobodi_listu(Cvor ** adresa_glave)* koja oslobađa dinamički zauzetu memoriju za čvorove liste.

   (e) *void ispisi_listu(Cvor * glava)* koja ispisuje vrednosti u čvorovima liste povezane strelicama.

   (f) *int velicina_liste(Cvor * glava)* koja vraća broj elemenata liste. Broj elemenata prazne liste je 0.

   (g) *int broj_negativnih(Cvor * glava)* koja vraća broj negativnih elemenata liste.

   (h) *int proizvod_liste(Cvor *glava)* koja računa proizvod svih elemenata liste.

   (i) *int zbir_liste(Cvor * glava)* koja računa zbir svih elemenata liste.

   (j) *float prosek_liste(Cvor * glava)* koja računa prosečnu vrednost svih elemenata liste.

   (k) *Cvor *minimum(Cvor * glava)* koja vraća adresu čvora koji sadrži najmanju vrednost. Ukoliko ih ima više, vratiti adresu prvog takvog čvora.

   (l) *int dva_najveca(Cvor * glava, int * max1, int * max2)* koja na adrese koje su prosleđene kao argumenti funkcije smešta dve najveće vrednosti liste. Ukoliko funkcija nema dovoljno elemenata, vraća -1, a inače 0.

   (m) *int uredjena_nerastuce(Cvor * glava)* koja proverava da li su elementi liste uređeni nerastuće. Ukoliko je to slučaj, funkcija treba da vrati 1, a inače 0.

   Napisati program koji koristi jednostruko povezanu listu za čuvanje celih brojeva koji se unose sa standardnog ulaza. Unošenje novih brojeva u listu prekida se učitavanjem kraja ulaza (EOF).

Primer 1

```
Interakcija sa programom:
Ulaz:
  Unesite elemente liste (Ctrl+D za kraj unosa):
  1 -2 3 8 1 9 7 -8 0
Izlaz:
  Lista:
  1 -> -2 -> 3 -> 8 -> 1 -> 9 -> 7 -> -8 -> 0
  Broj elemenata liste:  9
  Broj negativnih elemenata liste:  2
  Proizvod elemenata liste:  0
  Zbir elemenata liste:  19
  Prosečna vrednost:  2.11
  Minimum:  -8
  Dva najveca elementa:  8 9
  Lista nije uredjena nerastuce.
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  Unesite elemente liste (Ctrl+D za kraj unosa):
  4 3 2 1 -4
Izlaz:
  Lista:
  4 -> 3 -> 2 -> 1 -> -4
  Broj elemenata liste:  5
  Broj negativnih elemenata liste:  1
  Proizvod elemenata liste:  -96
  Zbir elemenata liste:  6
  Prosečna vrednost:  1.20
  Minimum:  -4
  Dva najveca elementa:  3 4
  Lista jeste uredjena nerastuce.
```

2. Definisati strukturu *Cvor* kojom se predstavlja čvor liste, a koja sadrži karakter *slovo* i pokazivač na sledeći čvor liste, a zatim implementirati sledeće funkcije:

   (a) *Cvor *napravi_cvor(char slovo)* koja kao argument dobija jedan karakter, kreira nov čvor liste, inicijalizuje mu polja i vraća njegovu adresu.

   (b) *void oslobodi_listu(Cvor ** adresa_glave)* koja oslobađa dinamički zauzetu memoriju za čvorove liste.

(c) *int dodaj_sortirano(Cvor ** adresa_glave, char slovo)* koja dodaje novi element u neopadajuće uređenu listu karaktera, tako da se očuva postojeće uređenje prema vrednosti ASCII koda. Funkcija treba da vrati 0, ako je sve bilo u redu, odnosno 1, ako se dogodila greška prilikom alokacije memorije za nov čvor.

(d) *void ispisi_listu(Cvor * glava)* koja ispisuje vrednosti u čvorovima liste, bez delimitera (kao jednu nisku).

(e) *void ispisi_listu_od_pocetka(Cvor * glava)* koja rekurzivno ispisuje vrednosti u čvorovima liste, bez delimitera (kao jednu nisku), počevši od prvog elementa liste.

(f) *void ispisi_listu_od_kraja(Cvor * glava)* koja rekurzivno ispisuje vrednosti u čvorovima liste, bez delimitera (kao jednu nisku), počevši od poslednjeg elementa liste.

Napisati program koji koristi jednostruko povezanu listu za čuvanje karaktera koji se unose sa standardnog ulaza. Unošenje novih brojeva u listu prekida se učitavanjem karaktera *!*.

Primer 1

```
Interakcija sa programom:
Ulaz:
 Unesite karaktere (!  za kraj unosa):
 j a z d c b k p !
Izlaz:
 Lista:  abcdjkpz
 Lista (rekurzivno, od pocetka):  abcdjkpz
 Lista (rekurzivno, od kraja):  zpkjdcba
```

Primer 2

```
Interakcija sa programom:
Ulaz:
 Unesite karaktere (!  za kraj unosa):
 b z Z A B c c !
Izlaz:
 Lista:  ABZbccz
 Lista (rekurzivno, od pocetka):  ABZbccz
 Lista (rekurzivno, od kraja):  zccbZBA
```

3. Definisati strukturu *Cvor* kojom se predstavlja čvor dvostruko povezane liste, a koja sadrži ceo broj *vrednost*, pokazivače na sledeći i prethodni čvor liste, a zatim implementirati sledeće funkcije:

   (a) *Cvor *napravi_cvor(int broj)* koja kao argument dobija ceo broj, kreira nov čvor liste, inicijalizuje mu polja i vraća njegovu adresu.

   (b) *void oslobodi_listu(Cvor ** adresa_glave, Cvor ** adresa_kraja)* koja oslobađa dinamički zauzetu memoriju za čvorove liste.

   (c) *int dodaj_na_pocetak(Cvor ** adresa_glave, Cvor ** adresa_kraja, int broj)* koja dodaje novi čvor sa vrednošću *broj* na početak liste. Funkcija treba da vrati 0, ako je sve bilo u redu, odnosno 1, ako se dogodila greška prilikom alokacije memorije za nov čvor.

   (d) *int palindrom(Cvor * glava, Cvor * kraj)* koja proverava da li elementi liste predstavljaju palindrom.

   (e) *int prvo_pojavljivanje(Cvor * glava, Cvor * kraj, int x)* koja vraća redni broj čvora u kom se prvi put, gledano s početka liste, pojavila vrednost x. Ukoliko se vrednost ne nalazi u listi, funkcija vraća -1.

   (f) *int poslednje_pojavljivanje(Cvor * glava, Cvor * kraj, int x)* koja vraća redni broj čvora u kom se poslednji put, gledano s početka liste, pojavila vrednost x. Ukoliko se vrednost ne nalazi u listi, funkcija vraća -1.

   Napisati program koji koristi dvostruko povezanu listu za čuvanje celih brojeva koji se unose sa standardnog ulaza. Unošenje novih brojeva u listu prekida se učitavanjem nule. Ukoliko je potrebno, implementirati funkciju za prebrojavanje elemenata liste.

Primer 1

```
Interakcija sa programom:
Ulaz:
 Unesite elemente liste (0 za kraj unosa):
 1 17 3 8 1 9 17 -8 0
Izlaz:
 Lista:
 -8 -> 17 -> 9 -> 1 -> 8 -> 3 -> 17 -> 1
 Elementi liste ne predstavljaju palindrom.
 Uneti element za pretragu:
 17
 Prvo pojavljivanje:  1
 Poslednje pojavljivanje:  6
```

Primer 2

```
Interakcija sa programom:
Ulaz:
 Unesite elemente liste (0 za kraj unosa):
 1 2 4 2 1 0
Izlaz:
 Lista:
 1 -> 2 -> 4 -> 2 -> 1
 Elementi liste predstavljaju palindrom.
 Uneti element za pretragu:
 2
 Prvo pojavljivanje:  1
 Poslednje pojavljivanje:  3
```

12.2 Dodatni zadaci

1. Biblioteku iz prvog zadatka obogatiti rekurzivnim implementacijama narednih funkcija:

   (a) *void ispisi_listu_rekurzivno(Cvor * glava)* koja ispisuje vrednosti u čvorovima liste povezane strelicama.

   (b) *int velicina_liste_rekurzivno(Cvor * glava)* koja vraća broj elemenata liste.

   (c) *int broj_neparnih_rekurzivno(Cvor * glava)* koja vraća broj neparnih elemenata liste.

(d) *int proizvod_liste_rekurzivno(Cvor * glava)* koja računa proizvod svih elemenata liste.

(e) *int zbir_liste_rekurzivno(Cvor * glava)* koja računa zbir svih elemenata liste.

(f) *int minimum_rekurzivno(Cvor * glava)* koja vraća najmanju vrednost koja se javlja u listi.

(g) *int maximum_rekurzivno(Cvor * glava)* koja vraća najveću vrednost koja se javlja u listi.

Primer 1

```
Interakcija sa programom:
Ulaz:
 Unesite elemente liste (Ctrl+D za kraj unosa):
 1 -2 3 8 1 9 7 -8 0
Izlaz:
 Lista:
 1 -> -2 -> 3 -> 8 -> 1 -> 9 -> 7 -> -8 -> 0
 Broj elemenata liste:  9
 Broj neparnih elemenata liste:  5
 Proizvod elemenata liste:  0
 Zbir elemenata liste:  19
 Minimum:  -8
 Maksimum:  9
```

Primer 2

```
Interakcija sa programom:
Ulaz:
 Unesite elemente liste (Ctrl+D za kraj unosa):
 -4 3 2 1 -4
Izlaz:
 Lista:
 -4 -> 3 -> 2 -> 1 -> -4
 Broj elemenata liste:  0
 Broj neparnih elemenata liste:  2
 Proizvod elemenata liste:  96
 Zbir elemenata liste:  -2
 Minimum:  -4
 Maksimum:  3
```

13 XII nedelja - Stabla

13.1 Obavezni zadaci

1. Napisati biblioteku za rad sa binarnim pretraživačkim stablima.

   (a) Definisati strukturu *Cvor* kojom se opisuje čvor stabla, a koja sadrži karakter *slovo* i pokazivače levo i desno redom na levo i desno podstablo.

   (b) Napisati funkciju *Cvor *napravi_cvor(char slovo)* koja alocira memoriju za novi čvor stabla i vrši njegovu inicijalizaciju zadatim karakterom *slovo*.

   (c) Napisati funkciju *int dodaj_u_stablo(Cvor ** adresa_korena, char slovo)* koja u stablo na koje pokazuje argument *adresa_korena* dodaje karakter *slovo*. Povratna vrednost funkcije je 0 ako je dodavanje uspešno, odnosno 1 ukoliko je došlo do greške.

   (d) Napisati funkciju *Cvor *pretrazi_stablo(Cvor * koren, char slovo)* koja proverava da li se karakter *slovo* nalazi u stablu sa korenom *koren*. Funkcija vraća pokazivač na čvor stabla koji sadrži traženu vrednost ili *NULL* ukoliko takav čvor ne postoji.

   (e) Napisati funkciju *void obrisi_element(Cvor ** adresa_korena, char slovo)* koja briše čvor koji sadrži karakter *slovo* iz stabla na koje pokazuje argument *adresa_korena*.

   (f) Napisati funkciju *void ispisi_stablo_infiksno(Cvor * koren)* koja infiksno ispisuje sadržaj stabla sa korenom *koren*. Infiksni ispis podrazumeva ispis levog podstabla, korena, a zatim i desnog podstabla.

   (g) Napisati funkciju *void ispisi_stablo_prefiksno(Cvor * koren)* koja prefiksno ispisuje sadržaj stabla sa korenom *koren*. Prefiksni ispis podrazumeva ispis korena, levog podstabla, a zatim i desnog podstabla.

   (h) Napisati funkciju *void ispisi_stablo_postfiksno(Cvor * koren)* koja postfiksno ispisuje sadržaj stabla sa korenom *koren*. Postfiksni ispis podrazumeva ispis levog podstabla, desnog podstabla, a zatim i korena.

   (i) Napisati funkciju *void oslobodi_stablo(Cvor ** adresa_korena)* koja oslobađa memoriju zauzetu stablom na koje pokazuje argument *adresa_korena*.

   Korišćenjem kreirane biblioteke, napisati program koji sa standardnog ulaza učitava karaktere sve dok se ne unese *!*, dodaje samo one koji predstavljaju malo slovo abecede (a-z) u binarno pretraživačko stablo i ispisuje stablo u svakoj od navedenih notacija. Zatim omogućiti unos još dva karaktera i demonstrirati rad funkcije za pretragu nad prvim unetim karakterom i rad funkcije za brisanje elemenata nad drugim unetim karakterom.

Primer 1

```
Interakcija sa programom:
Ulaz:
  Unesite elemente stabla (!  za kraj unosa):
  b z m k d h !
Izlaz:
  Infiksno:  b d h k m z
  Prefiksno:  b z m k d h
  Postfiksno:  h d k m z b
  Uneti karakter za pretragu:
  m
  Karakter se nalazi u stablu.
  Uneti karakter za brisanje:
  h
  Stablo nakon brisanja:  b d k m z
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  Unesite elemente stabla (!  za kraj unosa):
  p h u z a a m !
Izlaz:
  Infiksno:  a a h m p u z
  Prefiksno:  p h a a m u z
  Postfiksno:  a a m h z u p
  Uneti karakter za pretragu:
  q
  Karakter se ne nalazi u stablu.
  Uneti karakter za brisanje:
  l
  Stablo nakon brisanja:  a a m h z u p
```

2. Napisati sledeće funkcije za rad sa binarnim stablima, koja sadrže cele brojeve:

   (a) Funkcija koja izračunava broj čvorova stabla.

   (b) Funkcija koja izračunava broj listova stabla.

   (c) Funkcija koja izračunava proizvod čvorova stabla.

   (d) Funkcija koja izračunava broj prostih brojeva sadržanih u stablu.

   (e) Funkcija koja izračunava najmanji element u stablu.

   (f) Funkcija koja izračunava prosečnu vrednost čvorova.

   (g) Funkcija koja računa proizvod svih vrednosti u binarnom stablu pretrage koje su manje ili jednake od vrednosti sadržane u korenu.

   Napisati zatim i *main* funkciju koja testira ove funkcije. Cele brojeve unositi sa standardnog ulaza, sve dok se ne unese kraj ulaza (EOF). Ostale neophodne funkcije za rad sa stablima, kao i definiciju čvora stabla, iskoristiti iz priloženih fajlova *stabla.h* i *stabla.c*.

Primer 1

```
Interakcija sa programom:
Ulaz:
  Unesite elemente stabla (CTRL+D za kraj unosa):
  6 3 12 1 4 10 5
Izlaz:
  Broj cvorova:  7
  Broj listova:  3
  Proizvod cvorova:  43200
  Prostih brojeva:  3
  Najmanji element:  1
  Prosecna vrednost:  5.857
  Proizvod svih elemenata cija je vrednost manja
od 6:  360
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  Unesite elemente stabla (CTRL+D za kraj unosa):
  6 5 4 3 2 1
Izlaz:
  Broj cvorova:  6
  Broj listova:  1
  Proizvod cvorova:  720
  Prostih brojeva:  4
  Najmanji element:  1
  Prosecna vrednost:  3.500
  Proizvod svih elemenata cija je vrednost manja
od 6:  120
```

3. Napisati program koji izračunava i na standardni izlaz ispisuje broj pojavljivanja svakog od malih slova koja se javjaju u nizu niski. Niske se učitavaju iz datoteke čije se ime zadaje kao argument komandne linije. Program realizovati korišćenjem binarnog pretraživačkog stabla uređenog leksikografski po karakterima. Može se pretpostaviti da dužina reči neće biti veća od 50 karaktera, kao i da neće biti više od 100 niski. Napraviti svoju biblioteku modifikovanjem biblioteke iz prvog zadatka.

Primer 1

```
niske.txt
  antananarive ananas ana nana

Poziv:  ./a.out niske.txt

Interakcija sa programom:
  'a':  11
  'e':  1
  'i':  1
  'n':  8
  'r':  1
  's':  1
  't':  1
  'v':  1
```

Primer 2

```
karakteri.txt
  abbb acdbb bABC acd zzz

Poziv:  ./a.out karakteri.txt

Interakcija sa programom:
  'a':  3
  'b':  6
  'c':  1
  'd':  2
  'z':  3
```

13.2 Dodatni zadaci

1. Napisati funkciju koja proverava da li za svako podstablo binarnog stabla celih brojeva važi da je vrednost koja se nalazi u korenu podstabla jednaka razlici vrednosti korena desnog podstabla i vrednosti korena levog podstabla.

Primer 1

```
Interakcija sa programom:
Ulaz:
  Unesite elemente stabla (CTRL+D za kraj unosa):
  6 3 12 1 4 10 5
Izlaz:
  Ne vazi osobina.
```

Primer 2

```
Interakcija sa programom:
Ulaz:
  Unesite elemente stabla (CTRL+D za kraj unosa):
  8 3 11 2 5 10 21 20 41
Izlaz:
  Vazi osobina.
```